

Mario V. Wüthrich  
Michael Merz

# Statistical Foundations of Actuarial Learning and its Applications



Swiss Re



SAV  
ASA  
ASA

Schweizerische  
Aktuarvereinigung  
Association Suisse  
des Actuaire  
Associazione Svizzera  
degli Attuari

OPEN ACCESS



Springer

# Springer Actuarial

## **Editors-in-Chief**

Hansjoerg Albrecher, University of Lausanne, Lausanne, Switzerland

Michael Sherris, UNSW, Sydney, NSW, Australia

## **Series Editors**

Daniel Bauer, University of Wisconsin-Madison, Madison, WI, USA

Stéphane Loisel, ISFA, Université Lyon 1, Lyon, France

Alexander J. McNeil, University of York, York, UK

Antoon Pelsser, Maastricht University, Maastricht, The Netherlands

Ermanno Pitacco, Università di Trieste, Trieste, Italy

Gordon Willmot, University of Waterloo, Waterloo, ON, Canada

Hailiang Yang, The University of Hong Kong, Hong Kong, Hong Kong

This is a series on actuarial topics in a broad and interdisciplinary sense, aimed at students, academics and practitioners in the fields of insurance and finance.

Springer Actuarial informs timely on theoretical and practical aspects of topics like risk management, internal models, solvency, asset-liability management, market-consistent valuation, the actuarial control cycle, insurance and financial mathematics, and other related interdisciplinary areas.

The series aims to serve as a primary scientific reference for education, research, development and model validation.

The type of material considered for publication includes lecture notes, monographs and textbooks. All submissions will be peer-reviewed.

Mario V. Wüthrich • Michael Merz

# Statistical Foundations of Actuarial Learning and its Applications



Mario V. Wüthrich  
Department of Mathematics, RiskLab  
Switzerland  
ETH Zürich  
Zürich, Switzerland

Michael Merz  
Faculty of Business Administration  
University of Hamburg  
Hamburg, Germany



This work was supported by Schweizerische Aktuarvereinigung SAV and Swiss Re.

ISSN 2523-3262

ISSN 2523-3270 (electronic)

Springer Actuarial

ISBN 978-3-031-12408-2

ISBN 978-3-031-12409-9 (eBook)

<https://doi.org/10.1007/978-3-031-12409-9>

Mathematics Subject Classification: C13, C21/31, C24/34, G22, 62F10, 62F12, 62J07, 62J12, 62M45, 62P05, 68T01, 68T50

© The Authors 2023. This book is an open access publication.

**Open Access** This book is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this book are included in the book's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the book's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG  
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

# Acknowledgments

We kindly thank our very generous sponsors, the Swiss Association of Actuaries (SAA) and Swiss Re, for financing the open access option of the electronic version of this book. Our special thanks go to Sabine Betz (President of SAA), Adrian Kolly (Swiss Re), and Holger Walz (SAA) who were very positive and interested in this book project from the very beginning, and who made this open access funding possible within their institutions.

A very special thank you goes to Hans Bühlmann who has been supporting us over the last 30 years. We have had so many inspiring discussions over these years, and we have greatly benefited and learned from Hans' incredible knowledge and intuition.

Jointly with Christoph Buser, we have started to teach the lecture “Data Analytics for Non-Life Insurance Pricing” at ETH Zurich in 2018. Our data analytics lecture focuses (only) on the Poisson claim counts case, but its lecture notes have provided a first draft for this book project. This draft has been developed and extended to the general case of the exponential family. Since our first lecture, we have greatly benefited from interactions with many colleagues and students. In particular, we would like to mention the data science initiative “Actuarial Data Science” of the Swiss Association of Actuaries (chaired by Jürg Schelldorfer), whose tutorials provided a great stimulus for this book. Moreover, we mention the annual Insurance Data Science Conference (chaired by Markus Gesmann and Andreas Tsanakas) and the ASTIN Reading Club (chaired by Ronald Richman and Dimitri Semenovich). Furthermore, we would like to kindly thank Ronald Richman who has always been a driving force behind learning and adapting new machine learning techniques, and we also kindly thank Simon Rentzmann for many interesting discussions on how to apply these techniques on real insurance problems.

We thank the following colleagues by name (in alphabetical order). We collaborated and had inspiring discussions in the field of statistical learning with the following colleagues: Johannes Abegglen, Hansjörg Albrecher, Davide Apolloni, Peter Bühlmann, Christoph Buser, Patrick Cheridito, Łukasz Delong, Paul Embrechts, Andrea Ferrario, Tobias Fissler, Luca Fontana, Daisuke Frei, Tsz Chai Fung, Guangyuan Gao, Yan-Xing Lan, Gee Lee, Mathias Lindholm, Christian

Lorentzen, Friedrich Loser, Michael Mayer, Daniel Meier, Alexander Noll, Gareth Peters, Jan Rabenseifner, Peter Reinhard, Simon Rentzmann, Ronald Richman, Ludger Rüschendorf, Robert Salzmänn, Marc Sarbach, Jürg Schelldorfer, Pavel Shevchenko, Joël Thomann, Andreas Tsanakas, George Tzougas, Emiliano Valdez, Tim Verdonck, and Patrick Zöchbauer.

# Contents

- 1 Introduction** ..... 1
  - 1.1 The Statistical Modeling Cycle ..... 1
  - 1.2 Preliminaries on Probability Theory ..... 3
  - 1.3 Lab: Exploratory Data Analysis ..... 7
  - 1.4 Outline of This Book ..... 9
- 2 Exponential Dispersion Family** ..... 13
  - 2.1 Exponential Family ..... 13
    - 2.1.1 Definition and Properties ..... 13
    - 2.1.2 Single-Parameter Linear EF: Count Variable Examples ..... 18
    - 2.1.3 Vector-Valued Parameter EF: Absolutely Continuous Examples ..... 20
    - 2.1.4 Vector-Valued Parameter EF: Count Variable Example ..... 27
  - 2.2 Exponential Dispersion Family ..... 28
    - 2.2.1 Definition and Properties ..... 28
    - 2.2.2 Exponential Dispersion Family Examples ..... 31
    - 2.2.3 Tweedie’s Distributions ..... 34
    - 2.2.4 Steepness of the Cumulant Function ..... 37
    - 2.2.5 Lab: Large Claims Modeling ..... 38
  - 2.3 Information Geometry in Exponential Families ..... 40
    - 2.3.1 Kullback–Leibler Divergence ..... 40
    - 2.3.2 Unit Deviance and Bregman Divergence ..... 42
- 3 Estimation Theory** ..... 49
  - 3.1 Introduction to Decision Theory ..... 49
  - 3.2 Parameter Estimation ..... 51
  - 3.3 Unbiased Estimators ..... 56
    - 3.3.1 Cramér–Rao Information Bound ..... 56
    - 3.3.2 Information Bound in the Exponential Family Case ..... 62



3.4	Asymptotic Behavior of Estimators	67
3.4.1	Consistency	67
3.4.2	Asymptotic Normality	69
<b>4</b>	<b>Predictive Modeling and Forecast Evaluation</b>	<b>75</b>
4.1	Generalization Loss	75
4.1.1	Mean Squared Error of Prediction	76
4.1.2	Unit Deviances and Deviance Generalization Loss	79
4.1.3	A Decision-Theoretic Approach to Forecast Evaluation	88
4.2	Cross-Validation	95
4.2.1	In-Sample and Out-of-Sample Losses	95
4.2.2	Cross-Validation Techniques	98
4.2.3	Akaike's Information Criterion	103
4.3	Bootstrap	106
4.3.1	Non-parametric Bootstrap Simulation	106
4.3.2	Parametric Bootstrap Simulation	109
<b>5</b>	<b>Generalized Linear Models</b>	<b>111</b>
5.1	Generalized Linear Models and Log-Likelihoods	112
5.1.1	Regression Modeling	112
5.1.2	Definition of Generalized Linear Models	113
5.1.3	Link Functions and Feature Engineering	115
5.1.4	Log-Likelihood Function and Maximum Likelihood Estimation	116
5.1.5	Balance Property Under the Canonical Link Choice	122
5.1.6	Asymptotic Normality	123
5.1.7	Maximum Likelihood Estimation and Unit Deviances	124
5.2	Actuarial Applications of Generalized Linear Models	126
5.2.1	Selection of a Generalized Linear Model	126
5.2.2	Feature Engineering	127
5.2.3	Offsets	132
5.2.4	Lab: Poisson GLM for Car Insurance Frequencies	133
5.3	Model Validation	141
5.3.1	Residuals and Dispersion	141
5.3.2	Hypothesis Testing	145
5.3.3	Analysis of Variance	147
5.3.4	Lab: Poisson GLM for Car Insurance Frequencies, Revisited	150
5.3.5	Over-Dispersion in Claim Counts Modeling	155
5.3.6	Zero-Inflated Poisson Model	162
5.3.7	Lab: Gamma GLM for Claim Sizes	167

5.3.8	Lab: Inverse Gaussian GLM for Claim Sizes .....	173
5.3.9	Log-Normal Model for Claim Sizes: A Short Discussion .....	176
5.4	Quasi-Likelihoods .....	180
5.5	Double Generalized Linear Model .....	182
5.5.1	The Dispersion Submodel .....	182
5.5.2	Saddlepoint Approximation .....	183
5.5.3	Residual Maximum Likelihood Estimation .....	186
5.5.4	Lab: Double GLM Algorithm for Gamma Claim Sizes .....	187
5.5.5	Tweedie's Compound Poisson GLM .....	189
5.6	Diagnostic Tools .....	190
5.6.1	The Hat Matrix .....	190
5.6.2	Case Deletion and Generalized Cross-Validation .....	192
5.7	Generalized Linear Models with Categorical Responses .....	195
5.7.1	Logistic Categorical Generalized Linear Model .....	195
5.7.2	Maximum Likelihood Estimation in Categorical Models .....	196
5.8	Further Topics of Regression Modeling .....	198
5.8.1	Longitudinal Data and Random Effects .....	198
5.8.2	Regression Models Beyond the GLM Framework .....	199
5.8.3	Quantile Regression .....	202
<b>6</b>	<b>Bayesian Methods, Regularization and Expectation-Maximization .....</b>	<b>207</b>
6.1	Bayesian Parameter Estimation .....	207
6.2	Regularization .....	210
6.2.1	Maximal a Posterior Estimator .....	210
6.2.2	Ridge vs. LASSO Regularization .....	212
6.2.3	Ridge Regression .....	215
6.2.4	LASSO Regularization .....	217
6.2.5	Group LASSO Regularization .....	226
6.3	Expectation-Maximization Algorithm .....	230
6.3.1	Mixture Distributions .....	230
6.3.2	Incomplete and Complete Log-Likelihoods .....	232
6.3.3	Expectation-Maximization Algorithm for Mixtures .....	233
6.3.4	Lab: Mixture Distribution Applications .....	240
6.4	Truncated and Censored Data .....	248
6.4.1	Lower-Truncation and Right-Censoring .....	248
6.4.2	Parameter Estimation Under Right-Censoring .....	250
6.4.3	Parameter Estimation Under Lower-Truncation .....	254
6.4.4	Composite Models .....	264

<b>7</b>	<b>Deep Learning</b>	267
7.1	Deep Learning and Representation Learning	267
7.2	Generic Feed-Forward Neural Networks	269
7.2.1	Construction of Feed-Forward Neural Networks	269
7.2.2	Universality Theorems	274
7.2.3	Gradient Descent Methods	278
7.3	Feed-Forward Neural Network Examples	293
7.3.1	Feature Pre-processing	293
7.3.2	Lab: Poisson FN Network for Car Insurance Frequencies	295
7.4	Special Features in Networks	298
7.4.1	Special Purpose Layers	298
7.4.2	The Balance Property in Neural Networks	305
7.4.3	Boosting Regression Models with Network Features	315
7.4.4	Network Ensemble Learning	319
7.4.5	Identifiability in Feed-Forward Neural Networks	340
7.5	Auto-encoders	342
7.5.1	Standardization of the Data Matrix	343
7.5.2	Introduction to Auto-encoders	343
7.5.3	Principal Components Analysis	344
7.5.4	Lab: Lee–Carter Mortality Model	347
7.5.5	Bottleneck Neural Network	351
7.6	Model-Agnostic Tools	357
7.6.1	Variable Permutation Importance	357
7.6.2	Partial Dependence Plots	359
7.6.3	Interaction Strength	365
7.6.4	Local Model-Agnostic Methods	366
7.6.5	Marginal Attribution by Conditioning on Quantiles	366
7.7	Lab: Analysis of the Fitted Networks	376
<b>8</b>	<b>Recurrent Neural Networks</b>	381
8.1	Motivation for Recurrent Neural Networks	381
8.2	Plain-Vanilla Recurrent Neural Network	383
8.2.1	Recurrent Neural Network Layer	383
8.2.2	Deep Recurrent Neural Network Architectures	385
8.2.3	Designing the Network Output	387
8.2.4	Time-Distributed Layer	388
8.3	Special Recurrent Neural Networks	390
8.3.1	Long Short-Term Memory Network	390
8.3.2	Gated Recurrent Unit Network	392
8.4	Lab: Mortality Forecasting with RN Networks	394
8.4.1	Lee–Carter Model, Revisited	394
8.4.2	Direct LSTM Mortality Forecasting	402

<b>9</b>	<b>Convolutional Neural Networks</b>	407
9.1	Plain-Vanilla Convolutional Neural Network Layer	407
9.1.1	Input Tensors and Channels	408
9.1.2	Generic Convolutional Neural Network Layer	408
9.1.3	Example: Time-Series Analysis and Image Recognition	411
9.2	Special Purpose Tools for Convolutional Neural Networks	413
9.2.1	Padding with Zeros	413
9.2.2	Stride	414
9.2.3	Dilation	414
9.2.4	Pooling Layer	415
9.2.5	Flatten Layer	416
9.3	Convolutional Neural Network Architectures	416
9.3.1	Illustrative Example of a CN Network Architecture	416
9.3.2	Lab: Telematics Data	418
9.3.3	Lab: Mortality Surface Modeling	422
<b>10</b>	<b>Natural Language Processing</b>	425
10.1	Feature Pre-processing and Bag-of-Words	425
10.2	Word Embeddings	429
10.2.1	Word to Vector Algorithms	430
10.2.2	Global Vectors Algorithm	436
10.3	Lab: Predictive Modeling Using Word Embeddings	440
10.4	Lab: Deep Word Representation Learning	445
10.5	Outlook: Creating Attention	448
<b>11</b>	<b>Selected Topics in Deep Learning</b>	453
11.1	Deep Learning Under Model Uncertainty	453
11.1.1	Recap: Tweedie's Family	454
11.1.2	Lab: Claim Size Modeling Under Model Uncertainty	458
11.1.3	Lab: Deep Dispersion Modeling	466
11.1.4	Pseudo Maximum Likelihood Estimator	472
11.2	Deep Quantile Regression	476
11.2.1	Deep Quantile Regression: Single Quantile	477
11.2.2	Deep Quantile Regression: Multiple Quantiles	478
11.2.3	Lab: Deep Quantile Regression	479
11.3	Deep Composite Model Regression	483
11.3.1	Joint Elicitability of Quantiles and Expected Shortfalls	483
11.3.2	Lab: Deep Composite Model Regression	487
11.4	Model Uncertainty: A Bootstrap Approach	492
11.5	LocalGLMnet: An Interpretable Network Architecture	495
11.5.1	Definition of the LocalGLMnet	495
11.5.2	Variable Selection in LocalGLMnets	497

11.5.3	Lab: LocalGLMnet for Claim Frequency Modeling .....	499
11.5.4	Variable Selection Through Regularization of the LocalGLMnet .....	507
11.5.5	Lab: LASSO Regularization of LocalGLMnet .....	509
11.6	Selected Applications .....	513
11.6.1	Mixture Density Networks .....	513
11.6.2	Estimation of Conditional Expectations .....	521
11.6.3	Bayesian Networks: An Outlook .....	530
<b>12</b>	<b>Appendix A: Technical Results on Networks</b> .....	<b>537</b>
12.1	Universality Theorems .....	537
12.2	Consistency and Asymptotic Normality .....	540
12.3	Functional Limit Theorem .....	546
12.4	Hypothesis Testing .....	549
<b>13</b>	<b>Appendix B: Data and Examples</b> .....	<b>553</b>
13.1	French Motor Third Party Liability Data .....	553
13.2	Swedish Motorcycle Data .....	564
13.3	Wisconsin Local Government Property Insurance Fund .....	570
13.4	Swiss Accident Insurance Data .....	573
	<b>Bibliography</b> .....	<b>577</b>
	<b>Index</b> .....	<b>595</b>

# Chapter 1

## Introduction



### 1.1 The Statistical Modeling Cycle

We consider statistical modeling of insurance problems. This comprises the process of data collection, data analysis and statistical model building to forecast insured events that (may) happen in the future. This problem is at the very heart of statistics and statistical modeling. Our goal here is to present and provide the statistical tools that are useful in daily actuarial practice, in particular, we aim at describing the mathematical foundation behind these statistical concepts and how they can be applied. Statistical modeling has a wide range of applications, and, depending on the application, the theoretical aspects may be weighted differently. In insurance pricing we are mainly interested in optimal predictions, whereas economists often use statistical tools to explain observations, and in medical fields one is interested in causal effects that medications have on patients. Therefore, statistical theory is wide ranging, and one should always keep the corresponding application in mind. Shmueli [338] nicely discusses the difference between prediction and explanation; our focus here is mainly on prediction.

Box–Jenkins [49] and McCullagh–Nelder [265] distinguish three processes in statistical modeling: (i) model identification/selection, (ii) estimation, and (iii) prediction. In our statistical modeling cycle these three points are slightly modified and extended:

(1) Data collection, cleaning and pre-processing:

This item takes at least 80% of the total time in statistical modeling. It includes exploratory data analysis, data visualization and data pre-processing. This part of the modeling cycle does not seem to be very scientific, however, it is a highly important step because only extended data analysis allows the modeler to fully understand the data. Based on this knowledge the modeler can formulate her/his research question, her/his model, etc.

(2) Selection of a model class:

Based on the knowledge collected in the first item, the modeler has to select a suitable model class that is able to answer her/his research question. This model class can be in the sense of a data model (proper stochastic model), but it can also be an algorithmic model; we refer to the discussion on the “two modeling cultures” by Breiman [53].

(3) Choice of an objective function:

Once the modeler has specified a model class, she/he needs to define a decision rule how a particular member of the model class is selected for the collected data. Often this is in terms of an objective function, e.g., a scoring rule or a loss function that quantifies misspecification.

(4) Solving a (non-convex) optimization problem:

Once the first three items are completed, one is left with an optimization problem that tries to find the best model within the selected model class w.r.t. the given objective function and the collected data. In simple cases this optimization problem is a convex minimization problem for which numerical tools are in place. In more complex cases the optimization problem is neither convex nor concave, and the ‘best’ solution can often not be found explicitly. In that case, also the meaning of solution needs to be discussed.

(5) Model validation:

In the final/next step, the selected and fitted model needs to be validated. That is, does the model fit to the data, does it serve at predicting new data, does it answer the research question adequately, is there any better model/process choice, etc.?

(6) Possibly go back to (1):

If the answers in item (5) are not satisfactory, one typically goes back to (1). For instance, data pre-processing needs to be done differently, etc.

Especially, the two modeling cultures discussion of Breiman [53], after the turn of the millennium, has shaken up the statistical community. Having predictive performance as the main criterion, the data modeling culture has gradually shifted to the algorithmic culture, where the model itself plays a secondary role as long as the prediction is accurate. The latter is often in the form of a point predictor which can come from an algorithm. Lifting this discussion to a more scientific level, providing prediction uncertainty will slowly merge the two modeling cultures. There is another interesting discussion by Efron [116] on prediction, estimation (of model parameters) and attribution (predictor selection), that is very much at the core of statistical modeling. In these notes we want to especially emphasize the one modeling culture view of Yu–Barter [397] who expect the two modeling cultures of Breiman [53] to merge much closer than one would expect. Our goal is to demonstrate how all these different techniques and views can be seen as a unified modeling framework.

Concluding, the purpose of these notes is to discuss and illustrate how the different statistical techniques from the data modeling culture and the algorithmic modeling culture can be combined to solve actuarial questions in the best possible way. The main emphasis in this discussion lies on the statistical modeling tools,

and we present these tools along with actuarial examples. In actuarial practice one often distinguishes between life and general insurance. This distinction is done for good reasons. There are legislative reasons that require to legally separate life from general insurance business, but there are also modeling reasons, because insurance products in life and general insurance can have rather different features. In this book, we do not make this distinction because the statistical methods presented here can be useful in both branches of insurance, and we are going to consider life and general insurance examples, e.g., the former considering mortality forecasting and the latter aiming at insurance claims prediction for pricing.

## 1.2 Preliminaries on Probability Theory

The modern axiomatic foundation of probability theory was introduced in 1933 by the famous mathematician Kolmogoroff [221] in his book called “Grundbegriffe der Wahrscheinlichkeitsrechnung”. We give a brief introduction to probability theory and random variables; this introduction follows the lecture notes [387]. Throughout we assume to work on a sufficiently rich probability space  $(\Omega, \mathcal{A}, \mathbb{P})$ , meaning that this probability space should be able to carry all objects that we study. We denote (real-valued) random variables on this probability space by capital letters  $Y, Z, \dots$ , and random vectors use boldface capital letters, e.g., we have a random vector  $\mathbf{Y} = (Y_1, \dots, Y_q)^\top$  of dimension  $q \in \mathbb{N}$ , where each component  $Y_k$ ,  $1 \leq k \leq q$ , is a random variable. Random variables  $Y$  are characterized by (cumulative) distribution functions<sup>1</sup>  $F : \mathbb{R} \rightarrow [0, 1]$ , for  $y \in \mathbb{R}$

$$F(y) = \mathbb{P}[Y \leq y],$$

being the probability of the event that  $Y$  has a realization of less or equal to  $y$ . We write  $Y \sim F$  for  $Y$  having distribution function  $F$ . Similarly random vectors  $\mathbf{Y} \sim F$  are characterized by (cumulative) distribution functions  $F : \mathbb{R}^q \rightarrow [0, 1]$  with

$$F(\mathbf{y}) = \mathbb{P}[Y_1 \leq y_1, \dots, Y_q \leq y_q] \quad \text{for } \mathbf{y} = (y_1, \dots, y_q)^\top \in \mathbb{R}^q.$$

In insurance modeling, there are two important types of random variables, namely, discrete random variables and absolutely continuous random variables:

- The distribution function  $F$  of a discrete random variable  $Y$  is a step function with countably many steps in discrete points  $k \in \mathfrak{N} \subset \mathbb{R}$ . A discrete random variable has probability weights in these discrete points

$$f(k) = \mathbb{P}[Y = k] > 0 \quad \text{for } k \in \mathfrak{N},$$

---

<sup>1</sup> Cumulative distribution functions  $F$  are right-continuous, non-decreasing with  $\lim_{x \rightarrow -\infty} F(x) = 0$  and  $\lim_{x \rightarrow \infty} F(x) = 1$ .



satisfying  $\sum_{k \in \mathfrak{N}} f(k) = 1$ . If  $\mathfrak{N} \subseteq \mathbb{N}_0$ , the integer-valued random variable  $Y$  is called count random variable. Count random variables are used to model the number of claims in insurance. A similar situation occurs if  $Y$  models nominal outcomes, for instance, if  $Y$  models gender with female being encoded by 0 and male being encoded by 1, then  $f(0)$  is the probability weight of having a female and  $f(1) = 1 - f(0)$  the probability weight of having a male; in this case we identify the finite set  $\mathfrak{N} = \{0, 1\} = \{\text{female}, \text{male}\}$ .

- A random variable  $Y \sim F$  is said to be absolutely continuous<sup>2</sup> if there exists a non-negative (measurable) function  $f$ , called density of  $Y$ , such that

$$F(y) = \int_{-\infty}^y f(x) dx \quad \text{for all } y \in \mathbb{R}.$$

In that case we equivalently write  $Y \sim f$  and  $Y \sim F$ . Absolutely continuous random variables are often used to model claim sizes in insurance.

More generally speaking, discrete and absolutely continuous random variables have densities  $f(\cdot)$  w.r.t. a  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$ . In the former case, this  $\sigma$ -finite measure  $\nu$  is the counting measure on  $\mathfrak{N} \subset \mathbb{R}$ , and in the latter case it is the Lebesgue measure on  $\mathbb{R}$ . In actuarial science we also consider mixed cases, for instance, Tweedie's compound Poisson random variable is absolutely continuous on  $(0, \infty)$  having an additional point mass in 0; this model will be studied in Sect. 2.2.3, below.

Choose a random variable  $Y \sim F$  and a measurable function  $h : \mathbb{R} \rightarrow \mathbb{R}$ . The expected value of  $h(Y)$  is defined by (upon existence)

$$\mathbb{E}[h(Y)] = \int_{\mathbb{R}} h(y) dF(y).$$

We mainly focus on the following important examples of function  $h$ :

- expected value, mean or first moment of  $Y \sim F$ : for  $h(y) = y$

$$\mu = \mathbb{E}[Y] = \int_{\mathbb{R}} y dF(y);$$

- $k$ -th moment of  $Y \sim F$  for  $k \in \mathbb{N}$ : for  $h(y) = y^k$

$$\mathbb{E}[Y^k] = \int_{\mathbb{R}} y^k dF(y);$$

---

<sup>2</sup> Absolutely continuous is a stronger property than continuous.

- moment generating function of  $Y \sim F$  in  $r \in \mathbb{R}$ : for  $h(y) = e^{ry}$

$$M_Y(r) = \mathbb{E} \left[ e^{rY} \right] = \int_{\mathbb{R}} e^{ry} dF(y);$$

always subject to existence.

The moment generating function  $M_Y(\cdot)$  is sufficient for identifying distribution functions of random variables  $Y$ . The following statements are elementary and their proofs are based on Section 30 of Billingsley [34], for more details we also refer to Chapter 1 in the lecture notes [387]. Assume that the moment generating function of  $Y \sim F$  has a strictly positive radius of convergence  $\rho_0 > 0$  around the origin implying that  $M_Y(r) < \infty$  for all  $r \in (-\rho_0, \rho_0)$ . In this case we can write  $M_Y(r)$  as a power series expansion

$$M_Y(r) = \sum_{k=0}^{\infty} \frac{r^k}{k!} \mathbb{E} \left[ Y^k \right] \quad \text{for all } r \in (-\rho_0, \rho_0).$$

As a consequence we can differentiate  $M_Y(\cdot)$  in the open interval  $(-\rho_0, \rho_0)$  arbitrarily often, term by term under the sum. The derivatives in  $r = 0$  provide the  $k$ -th moments (which all exist and are finite)

$$\frac{d^k}{dr^k} M_Y(r)|_{r=0} = \mathbb{E} \left[ Y^k \right] \quad \text{for all } k \in \mathbb{N}_0. \quad (1.1)$$

In particular, in this case we immediately know that all moments of  $Y$  exist, and these moments completely determine the moment generating function  $M_Y$  of  $Y$ . Another consequence is that for a random variable  $Y$ , whose moment generating function  $M_Y$  has a strictly positive radius of convergence around the origin, the distribution function  $F$  is fully determined by this moment generating function. That is, if we have two such random variables  $Y_1$  and  $Y_2$  with  $M_{Y_1}(r) = M_{Y_2}(r)$  for all  $r \in (-r_0, r_0)$ , for some  $r_0 > 0$ , then  $Y_1 \stackrel{(d)}{=} Y_2$ .<sup>3</sup> Thus, these two random variables have the same distribution function. This statement carries over to the limit, i.e., if we have a sequence of random variables  $(Y_n)_n$  whose moment generating functions converge on a common interval  $(-r_0, r_0)$ , for some  $r_0 > 0$ , to the moment generating function of  $Y$ , also being finite on  $(-r_0, r_0)$ , then  $(Y_n)_n$  converges in distribution to  $Y$ ; such an argument is used to prove the central limit theorem (CLT).

---

<sup>3</sup> The notation  $Y_1 \stackrel{(d)}{=} Y_2$  is generally used for equality in distribution meaning that  $Y_1$  and  $Y_2$  have the same distribution function.

In insurance, we often deal with so-called positive random variables  $Y$ , meaning that  $Y \geq 0$ , almost surely (a.s.). In that case, the statements about moment generating functions and distributions hold true without the assumption of having a positive radius of convergence around the origin, see Theorem 22.2 in Billingsley [34]. Note that for positive random variables the moment generating function  $M_Y(r)$  exists for all  $r \leq 0$ .

Existence of the moment generating function  $M_Y(r)$  for some positive  $r > 0$  can also be interpreted as having a light-tailed distribution function. Observe that if  $M_Y(r)$  exists for some positive  $r > 0$ , then we can choose  $s \in (0, r)$  and Chebychev's inequality gives us (we assume  $Y \geq 0$ , a.s., here)

$$\mathbb{P}[Y > y] = \mathbb{P}[\exp\{sY\} > \exp\{sy\}] \leq \exp\{-sy\} M_Y(s). \quad (1.2)$$

The latter tells us that the survival function  $1 - F(y) = \mathbb{P}[Y > y]$  decays exponentially for  $y \rightarrow \infty$ . Heavy-tailed distribution functions do not have this property, but the survival function decays slower than exponentially as  $y \rightarrow \infty$ . This slower decay of the survival function is the case for so-called subexponential distribution functions (an example is the log-normal distribution, we refer to Rolski et al. [320]) and for regularly varying survival functions (an example is the Pareto distribution). Regularly varying survival functions  $1 - F$  have the property

$$\lim_{y \rightarrow \infty} \frac{1 - F(ty)}{1 - F(y)} = t^{-\beta} \quad \text{for all } t > 0 \text{ and some } \beta > 0. \quad (1.3)$$

These distribution functions have a polynomial tail (power tail) with tail index  $\beta > 0$ . In particular, if a positively supported distribution function  $F$  has a regularly varying survival function with tail index  $\beta > 0$ , then this distribution function is also subexponential, see Theorem 2.5.5 in Rolski et al. [320].

We are not going to specifically focus on heavy-tailed distribution functions, here, but we will explain how light-tailed random variables can be transformed to enjoy heavy-tailed properties. In these notes, we are mainly interested in studying different aspects of regression modeling. Regression modeling requires numerous observations to be able to successfully fit these models to the data. By definition, large claims are scarce, as they live in the tail of the distribution function and, thus, correspond to rare events. Therefore, it is often not possible to employ a regression model for scarce tail events. For this reason, extreme value analysis only plays a marginal role in these notes, though, it has a significant impact on insurance prices. For more on extreme value theory we refer to the relevant literature, see, e.g., Embrechts et al. [121], Rolski et al. [320], Mikosch [277] and Albrecher et al. [7].

### 1.3 Lab: Exploratory Data Analysis

Our theory is going to be supported by several data examples. These examples are mostly based on publicly available data. The different data sets are described in detail in Chap. 13. We highly recommend the reader to use these data sets to gain her/his own modeling experience.

We describe some tools here that allow for a descriptive and exploratory analysis of the available data; exploratory data analysis has been introduced and promoted by Tukey [357]. We consider the observed claim sizes of the Swedish motorcycle data set described in Sect. 13.2. This data set consists of 656 (positive) claim amounts  $y_i$ ,  $1 \leq i \leq n = 656$ . These claim amounts are illustrated in the boxplots of Fig. 1.1.

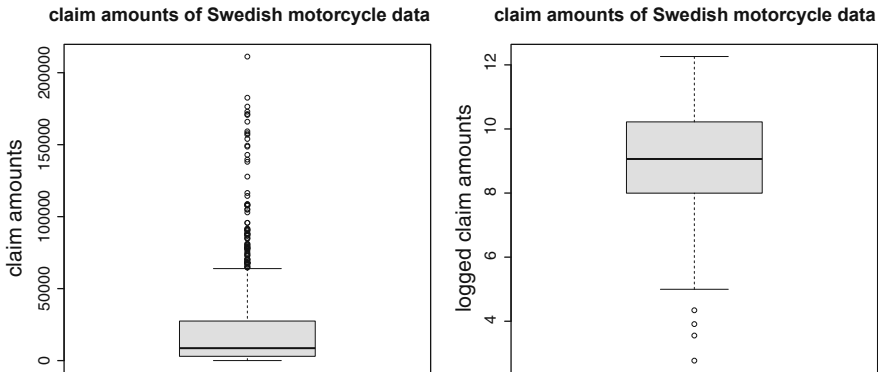
Typically in insurance, there are large claims that dominate the picture, see Fig. 1.1 (lhs). This results in right-skewed distribution functions, and such data is better illustrated on the log scale, see Fig. 1.1 (rhs). The latter, of course, assumes that all claims are strictly positive.

Figure 1.2 (lhs) shows the empirical distribution function of the observations  $y_i$ ,  $1 \leq i \leq n$ , which is obtained by

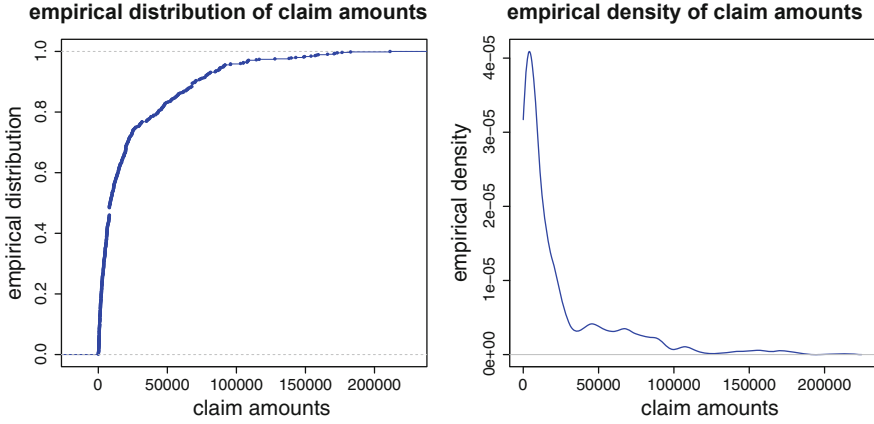
$$\hat{F}_n(y) = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{\{y_i \leq y\}} \quad \text{for } y \in \mathbb{R}.$$

If this data set has been generated by i.i.d. random variables, then the Glivenko–Cantelli theorem [64, 159] tells us that this empirical distribution function  $\hat{F}_n$  converges uniformly to the (true) data generating distribution function, a.s., as the number  $n$  of observations converges to infinity, see Theorem 20.6 in Billingsley [34].

Figure 1.2 (rhs) shows the empirical density of the observations  $y_i$ ,  $1 \leq i \leq n$ . This empirical density is obtained by considering a kernel smoother of a given



**Fig. 1.1** Boxplot of the claim amounts of the Swedish motorcycle data set: (lhs) on the original scale and (rhs) on the log scale



**Fig. 1.2** (lhs) Empirical distribution and (rhs) empirical density of the observed claim amounts  $y_i$ ,  $1 \leq i \leq n$

bandwidth around each observation  $y_i$ . The standard choice is the Gaussian kernel, with the bandwidth determining the variance parameter  $\sigma^2 > 0$  of the Gaussian density,

$$y \mapsto \hat{f}_n(y) = \frac{1}{n} \sum_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left\{ -\frac{1}{2} \frac{(y - y_i)^2}{\sigma^2} \right\}.$$

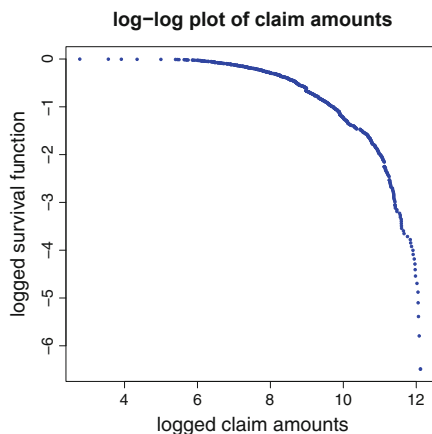
From the graph in Fig. 1.2 (rhs) we observe that the main body of the claim sizes is below an amount of 50'000, but the biggest claim exceeds 200'000. The latter motivates to study heavy-tailedness of the claim size data. Therefore, one usually benchmarks with a distribution function  $F$  that has a regularly varying survival function with a tail index  $\beta > 0$ , see (1.3). Asymptotically a regularly varying survival function behaves as  $y^{-\beta}$ ; for this reason the log-log plot is a popular tool to identify regularly varying tails. The log-log plot of a distribution function  $F$  is obtained by considering

$$y > 0 \mapsto (\log y, \log(1 - F(y))) \in \mathbb{R}^2.$$

Figure 1.3 gives the log-log plot of the empirical distribution function  $\hat{F}_n$ . If this plot looks asymptotically (for  $y \rightarrow \infty$ ) like a straight line with a negative slope  $-\beta$ , then the data shows heavy-tailedness in the sense of regular variation. Such data cannot be modeled by a distribution function for which the moment generating function  $M_Y(r)$  exists for some positive  $r > 0$ , see (1.2). Figure 1.3 does not suggest a regularly varying tail as we do not see an obvious asymptotic straight line for increasing claim sizes.

These graphs give us a first indication what the claim size data is about. Later on we are going to introduce explanatory variables that describe the insurance

**Fig. 1.3** Log-log plot of the empirical distribution function  $\hat{F}_n$



policyholders behind these claims. These explanatory variables characterize the policyholder and the general goal is to get a better description of the claim sizes as a function of these explanatory variables, e.g., older policyholders may cause larger claims than younger ones, etc. Such patterns are called *systematic effects* that can be explained by explanatory variables.

## 1.4 Outline of This Book

This book has eleven chapters (including the present one), and it has two appendices. We briefly describe the contents of these chapters and appendices.

In Chap. 2 we introduce and discuss the exponential family (EF) and the exponential dispersion family (EDF). The EF and the EDF are by far the most important classes of distribution functions for regression modeling. They include, among others, the Gaussian, the binomial, the Poisson, the gamma, the inverse Gaussian and Tweedie's models. We introduce these families of distribution functions, discuss their properties and provide several examples. Moreover, we introduce the Kullback–Leibler (KL) divergence and the Bregman divergence, which are important tools in model evaluation.

Chapter 3 is on classical statistical decision theory. This chapter is important for historical reasons, but it also provides the right mathematical grounding and intuition for more modern tools from data science and machine learning. In particular, we discuss maximum likelihood estimation (MLE), unbiasedness, consistency and asymptotic normality of MLEs in this chapter.

Chapter 4 is the core theoretical chapter on predictive modeling and forecast evaluation. The main problem in actuarial modeling is to forecast and price future claims. For this, we build predictive models, and this chapter deals with assessing and ranking these predictive models. We therefore introduce the mean squared

error of prediction (MSEP) and, more generally, the generalization loss (GL) to assess predictive models. This chapter is complemented by a more decision-theoretic approach to forecast evaluation, it discusses deviance losses, proper scoring, elicibility, forecast dominance, cross-validation, Akaike's information criterion (AIC) and we give an introduction to the bootstrap simulation method.

Chapter 5 discusses state-of-the-art statistical modeling in insurance which is the generalized linear model (GLM). We discuss GLMs in the light of claim count and claim size modeling, we present feature engineering, model fitting, model selection, over-dispersion, zero-inflated claim counts problems, double GLMs, and insurance-specific issues such as the balance property for having unbiasedness.

Chapter 6 summarizes some techniques that use Bayes' theorem. These are classical Bayesian statistical models, e.g., using the Markov chain Monte Carlo (MCMC) method for model fitting. This chapter discusses regularization of regression models such as ridge and LASSO regularization, which has a Bayesian interpretation, and it concerns the Expectation-Maximization (EM) algorithm. The EM algorithm is a general purpose tool that can handle incomplete data settings. We illustrate this for different examples coming from mixture distributions, censored and truncated claims data.

The core of this book are deep learning methods and neural networks. Chapter 7 considers deep feed-forward neural (FN) networks. We introduce the generic architecture of deep FN networks, and we discuss universality theorems of FN networks. We present network fitting, back-propagation, embedding layers for categorical variables and insurance-specific issues such as the balance property in network fitting and network ensembling to reduce model uncertainty. This chapter is complemented by many examples on non-life insurance pricing, but also on mortality modeling, as well as tools that help to explain deep FN network regression results.

Chapters 8 and 9 consider recurrent neural (RN) networks and convolutional neural (CN) networks. These are special network architectures that are useful for time-series and spatial data modeling, e.g., applied to image recognition problems. Time-series and images have a natural topology, and RN and CN networks try to benefit from this additional structure (over tabular data). We introduce these network architectures and provide insurance-relevant examples.

Chapter 10 discusses natural language processing (NLP) which deals with regression modeling of non-tabular or unstructured text data. We explain how words can be embedded into low-dimension spaces that serve as numerical word encodings. These can then be used for text recognition, either using RN networks or attention layers. We give an example where we aim at predicting claim perils from claim descriptions.

Chapter 11 is a selection of different topics. We mention forecasting under model uncertainty, deep quantile regression, deep composite regression or the LocalGLMnet which is an interpretable FN network architecture. Moreover, we provide a bootstrap example to assess prediction uncertainty, and we discuss mixture density networks.

Chapter 12 (Appendix A) is a technical chapter that discusses universality theorems for networks and sieve estimators, which are useful for studying asymptotic normality within a network framework. Chapter 13 (Appendix B) illustrates the data used in this book.

Finally, we remark that the book is written in a typical mathematical style using the structure of Lemmas, Theorems, etc. Results and statements which are particularly important for applications are highlighted with gray boxes.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.





## Chapter 2

# Exponential Dispersion Family



We introduce the exponential family (EF) and the exponential dispersion family (EDF) in this chapter. The single-parameter EF has been introduced in 1934 by the British statistician Sir Fisher [128], and it has been extended to vector-valued parameters by Darrois [88], Koopman [223] and Pitman [306] between 1935 and 1936. It is the most commonly used family of distribution functions in statistical modeling; among others, it contains the Gaussian distribution, the gamma distribution, the binomial distribution and the Poisson distribution. Its parametrization is taken in a special form that is convenient for statistical modeling. The EF can be introduced in a constructive way providing the main properties of this family of distribution functions. In this chapter we follow Jørgensen [201–203] and Barndorff-Nielsen [23], and we state the most important results based on this constructive introduction. This gives us a unified notation which is going to be useful for our purposes.

## 2.1 Exponential Family

### 2.1.1 Definition and Properties

We define the EF w.r.t. a  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$ . The results in this section can be generalized to  $\sigma$ -finite measures on  $\mathbb{R}^m$ , but such an extension is not necessary for our purposes. Select an integer  $k \in \mathbb{N}$ , and choose measurable functions  $a : \mathbb{R} \rightarrow \mathbb{R}$  and  $T : \mathbb{R} \rightarrow \mathbb{R}^k$ .<sup>1</sup> Consider for a *canonical parameter*  $\theta \in \mathbb{R}^k$  the Laplace

---

<sup>1</sup> We could also use boldface notation for  $T$  because  $T(y) \in \mathbb{R}^k$  is vector-valued, but we prefer to not use boldface notation for (vector-valued) functions.

transform

$$\mathfrak{L}(\boldsymbol{\theta}) = \int_{\mathbb{R}} \exp \left\{ \boldsymbol{\theta}^\top T(y) + a(y) \right\} d\nu(y).$$

Assume that this Laplace transform is not identically equal to  $+\infty$ . The *effective domain* is defined by

$$\boldsymbol{\Theta} = \left\{ \boldsymbol{\theta} \in \mathbb{R}^k; \mathfrak{L}(\boldsymbol{\theta}) < \infty \right\} \subseteq \mathbb{R}^k. \quad (2.1)$$

**Lemma 2.1** *The effective domain  $\boldsymbol{\Theta} \subseteq \mathbb{R}^k$  is a convex set.*

The effective domain  $\boldsymbol{\Theta}$  is not necessarily an open set, but in many applications it is open. Counterexamples are given in Problem 4.1 of Chapter 1 in Lehmann [244], and in the inverse Gaussian example in Sect. 2.1.3, below.

**Proof of Lemma 2.1** Choose  $\boldsymbol{\theta}_i \in \mathbb{R}^k$ ,  $i = 1, 2$ , with  $\mathfrak{L}(\boldsymbol{\theta}_i) < \infty$ . Set  $\boldsymbol{\theta} = c\boldsymbol{\theta}_1 + (1 - c)\boldsymbol{\theta}_2$  for  $c \in (0, 1)$ . We use Hölder's inequality, applied to the norms  $p = 1/c$  and  $q = 1/(1 - c)$ ,

$$\begin{aligned} \mathfrak{L}(\boldsymbol{\theta}) &= \int_{\mathbb{R}} \exp \left\{ (c\boldsymbol{\theta}_1 + (1 - c)\boldsymbol{\theta}_2)^\top T(y) + a(y) \right\} d\nu(y) \\ &= \int_{\mathbb{R}} \exp \left\{ \boldsymbol{\theta}_1^\top T(y) + a(y) \right\}^c \exp \left\{ \boldsymbol{\theta}_2^\top T(y) + a(y) \right\}^{1-c} d\nu(y) \\ &\leq \mathfrak{L}(\boldsymbol{\theta}_1)^c \mathfrak{L}(\boldsymbol{\theta}_2)^{1-c} < \infty. \end{aligned}$$

This implies  $\boldsymbol{\theta} \in \boldsymbol{\Theta}$  and proves the claim. □

We define *the cumulant function* on the effective domain  $\boldsymbol{\Theta}$

$$\kappa : \boldsymbol{\Theta} \rightarrow \mathbb{R}, \quad \boldsymbol{\theta} \mapsto \kappa(\boldsymbol{\theta}) = \log \mathfrak{L}(\boldsymbol{\theta}).$$

**Definition 2.2** The EF with  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$  and cumulant function  $\kappa : \boldsymbol{\Theta} \rightarrow \mathbb{R}$  is given by the distribution functions  $F$  on  $\mathbb{R}$  with

$$dF(y; \boldsymbol{\theta}) = f(y; \boldsymbol{\theta}) d\nu(y) = \exp \left\{ \boldsymbol{\theta}^\top T(y) - \kappa(\boldsymbol{\theta}) + a(y) \right\} d\nu(y), \quad (2.2)$$

for canonical parameters  $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subseteq \mathbb{R}^k$ .

## Remarks 2.3

- The definition of the EF (2.2) assumes that the effective domain  $\Theta \subseteq \mathbb{R}^k$  has been constructed from the choices  $a : \mathbb{R} \rightarrow \mathbb{R}$  and  $T : \mathbb{R} \rightarrow \mathbb{R}^k$  as described in (2.1). This is not explicitly stated in the surrounding text of (2.2).
- The support of any random variable  $Y \sim F(\cdot; \theta)$  of this EF does *not* depend on the explicit choice of the canonical parameter  $\theta \in \Theta$ , but solely on the choice of the  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$ , and the distribution functions  $F(\cdot; \theta)$  are mutually absolutely continuous (equivalent) w.r.t.  $\nu$ .
- In statistics, the main object of interest is the canonical parameter  $\theta$ . Importantly for parameter estimation, the function  $a(\cdot)$  does not involve the canonical parameter. Therefore, it is irrelevant for parameter estimation and (only) serves as a normalization so that  $F$  in (2.2) is a proper distribution function. In fact, this is the way how the EF is often introduced in the statistical and actuarial literature, but in this latter introduction we lose the deeper interpretation of the cumulant function  $\kappa$ , nor is it immediately clear what properties it possesses.
- The case  $k \geq 2$  gives a *vector-valued* canonical parameter  $\theta$ . The case  $k = 1$  gives a *single-parameter* EF, and, if additionally  $T(y) = y$ , it is called a *single-parameter linear EF*.

**Theorem 2.4** Assume the effective domain  $\Theta$  has a non-empty interior  $\overset{\circ}{\Theta}$ . Choose  $Y \sim F(\cdot; \theta)$  for fixed  $\theta \in \overset{\circ}{\Theta}$ . The moment generating function of  $T(Y)$  for sufficiently small  $\mathbf{r} \in \mathbb{R}^k$  is given by

$$M_{T(Y)}(\mathbf{r}) = \mathbb{E}_{\theta} \left[ \exp \left\{ \mathbf{r}^{\top} T(Y) \right\} \right] = \exp \{ \kappa(\theta + \mathbf{r}) - \kappa(\theta) \},$$

where the expectation operator  $\mathbb{E}_{\theta}$  illustrates the selected canonical parameter  $\theta$  for  $Y$ .

**Proof** Choose  $\theta \in \overset{\circ}{\Theta}$  and  $\mathbf{r} \in \mathbb{R}^k$  so small that  $\theta + \mathbf{r} \in \overset{\circ}{\Theta}$ . We receive

$$\begin{aligned} M_{T(Y)}(\mathbf{r}) &= \int_{\mathbb{R}} \exp \left\{ (\theta + \mathbf{r})^{\top} T(y) - \kappa(\theta) + a(y) \right\} d\nu(y) \\ &= \exp \{ \kappa(\theta + \mathbf{r}) - \kappa(\theta) \} \int_{\mathbb{R}} \exp \left\{ (\theta + \mathbf{r})^{\top} T(y) - \kappa(\theta + \mathbf{r}) + a(y) \right\} d\nu(y) \\ &= \exp \{ \kappa(\theta + \mathbf{r}) - \kappa(\theta) \}, \end{aligned}$$

where the last identity follows from the fact that the support of the EF does not depend on the explicit choice of the canonical parameter.  $\square$

Theorem 2.4 has a couple of immediate implications. First, in any interior point  $\theta \in \overset{\circ}{\Theta}$  both the moment generating function  $\mathbf{r} \mapsto M_{T(Y)}(\mathbf{r})$  (in the neighborhood of the origin) and the cumulant function  $\theta \mapsto \kappa(\theta)$  have derivatives of all orders, and, similarly to Sect. 1.2, moments of all orders of  $T(Y)$  exist, see also (1.1). Existence

of moments of all orders implies that the distribution function of  $T(Y)$  cannot have a regularly varying tails.

**Corollary 2.5** *Assume  $\overset{\circ}{\Theta}$  is non-empty. The cumulant function  $\theta \mapsto \kappa(\theta)$  is convex, and for  $Y \sim F(\cdot; \theta)$  with  $\theta \in \overset{\circ}{\Theta}$*

$$\mu = \mathbb{E}_{\theta} [T(Y)] = \nabla_{\theta} \kappa(\theta) \quad \text{and} \quad \text{Var}_{\theta} (T(Y)) = \nabla_{\theta}^2 \kappa(\theta),$$

where  $\nabla_{\theta}$  is the gradient and  $\nabla_{\theta}^2$  the Hessian w.r.t. vector  $\theta$ .

Similarly to  $T : \mathbb{R} \rightarrow \mathbb{R}^k$ , we will not use boldface notation for the (multi-dimensional) mean because later on we will understand the mean  $\mu = \mu(\theta) \in \mathbb{R}^k$  as a function of the canonical parameter  $\theta$ ; see Footnote 1 on page 13 on boldface notation.

**Proof** Existence of the moment generating function for all sufficiently small  $\mathbf{r} \in \mathbb{R}^k$  (around the origin) implies that we have first and second moments. For the first moment we receive

$$\mu = \mathbb{E}_{\theta} [T(Y)] = \nabla_{\mathbf{r}} M_{T(Y)}(\mathbf{r})|_{\mathbf{r}=0} = \exp \{ \kappa(\theta + \mathbf{r}) - \kappa(\theta) \} \nabla_{\mathbf{r}} \kappa(\theta + \mathbf{r})|_{\mathbf{r}=0} = \nabla_{\theta} \kappa(\theta).$$

Denote component  $j$  of  $T(Y) \in \mathbb{R}^k$  by  $T_j(Y)$ . We have for  $1 \leq j, l \leq k$

$$\begin{aligned} \mathbb{E}_{\theta} [T_j(Y) T_l(Y)] &= \frac{\partial^2}{\partial r_j \partial r_l} M_{T(Y)}(\mathbf{r}) \Big|_{\mathbf{r}=0} \\ &= \exp \{ \kappa(\theta + \mathbf{r}) - \kappa(\theta) \} \left( \frac{\partial^2}{\partial r_j \partial r_l} \kappa(\theta + \mathbf{r}) + \frac{\partial}{\partial r_j} \kappa(\theta + \mathbf{r}) \frac{\partial}{\partial r_l} \kappa(\theta + \mathbf{r}) \right) \Big|_{\mathbf{r}=0} \\ &= \left( \frac{\partial^2}{\partial \theta_j \partial \theta_l} \kappa(\theta) + \frac{\partial}{\partial \theta_j} \kappa(\theta) \frac{\partial}{\partial \theta_l} \kappa(\theta) \right). \end{aligned}$$

This implies for the covariance

$$\text{Cov}_{\theta}(T_j(Y), T_l(Y)) = \frac{\partial^2}{\partial \theta_j \partial \theta_l} \kappa(\theta).$$

The convexity of  $\kappa$  follows because  $\nabla_{\theta}^2 \kappa(\theta)$  is the positive semi-definite covariance matrix of  $T(Y)$ , for all  $\theta \in \overset{\circ}{\Theta}$ . This finishes the proof.  $\square$

**Assumption 2.6 (Minimal Representation)** *We assume that the interior  $\overset{\circ}{\Theta}$  of the effective domain  $\Theta$  is non-empty and that the cumulant function  $\kappa$  is strictly convex on this interior  $\overset{\circ}{\Theta}$ .*

*Remarks 2.7*

- Throughout these notes we will work under Assumption 2.6 without making explicit reference. This assumption strengthens the properties of the cumulant function  $\kappa$  from being convex, see Corollary 2.5, to being strictly convex. This strengthening implies that the mean function  $\theta \mapsto \mu = \mu(\theta) = \nabla_{\theta} \kappa(\theta)$  can be inverted; this is needed for the canonical link, see Definition 2.8, below.
- The strict convexity of  $\kappa$  means that the covariance matrix  $\nabla_{\theta}^2 \kappa(\theta)$  of  $T(Y)$  is positive definite and has full rank  $k$  for all  $\theta \in \mathring{\Theta}$ , see Corollary 2.5. This property is important, otherwise we do not have identifiability in the canonical parameter  $\theta$  because we have a linear dependence between the components of  $T(Y)$ .
- Mathematically, this strict convexity is not a restriction because it can be obtained by working under a so-called *minimal representation*. If the covariance matrix  $\nabla_{\theta}^2 \kappa(\theta)$  does not have full rank  $k$ , the choice  $k$  is “non-optimal” because the problem lives in a smaller dimension. Thus, w.l.o.g., we may and will assume to work in this smaller dimension, called minimal representation; for a rigorous derivation of a minimal representation we refer to Section 8.1 in Barndorff-Nielsen [23].

**Definition 2.8** The canonical link is defined by  $h = (\nabla_{\theta} \kappa)^{-1}$ .

The application of the canonical link  $h$  to the mean implies under Assumption 2.6

$$h(\mu) = h(\mathbb{E}_{\theta}[T(Y)]) = \theta,$$

for mean  $\mu = \mathbb{E}_{\theta}[T(Y)]$  of  $Y \sim F(\cdot; \theta)$  with  $\theta \in \mathring{\Theta}$ .

*Remarks 2.9 (Dual Parameter Space)* Assumption 2.6 provides that the canonical link  $h$  is well-defined, and we can either work with the canonical parameter representation  $\theta \in \mathring{\Theta} \subseteq \mathbb{R}^k$  or with its dual (mean) parameter representation  $\mu = \mathbb{E}_{\theta}[T(Y)] \in \mathcal{M}$  with

$$\mathcal{M} \stackrel{\text{def.}}{=} \nabla_{\theta} \kappa(\mathring{\Theta}) = \{\nabla_{\theta} \kappa(\theta); \theta \in \mathring{\Theta}\} \subseteq \mathbb{R}^k. \quad (2.3)$$

Strict convexity of  $\kappa$  implies that there is a one-to-one correspondence between these two parametrizations.  $\mathring{\Theta}$  is called the *effective domain* and  $\mathcal{M}$  is called the *dual parameter space* or the *mean parameter space*.

In Sect. 2.2.4, below, we introduce one more property called *steepness* that the cumulant function  $\kappa$  should satisfy. This additional property gives a relationship between the support  $\mathfrak{T}$  of the random variables  $T(Y)$  of the given EF and the boundary of the dual parameter space  $\mathcal{M}$ . This steepness property is important for parameter estimation.

### 2.1.2 Single-Parameter Linear EF: Count Variable Examples

We start by giving single-parameter discrete linear EF examples based on counting measures on  $\mathbb{N}_0$ . Since we work in one dimension  $k = 1$ , we replace boldface  $\theta$  by scalar  $\theta \in \Theta \subseteq \mathbb{R}$  in this section.

#### Bernoulli Distribution as a Single-Parameter Linear EF

For the Bernoulli distribution with parameter  $p \in (0, 1)$  we choose as  $\nu$  the counting measure on  $\{0, 1\}$ . We make the following choices:  $T(y) = y$ ,

$$a(y) = 0, \quad \kappa(\theta) = \log(1 + e^\theta), \quad p = \kappa'(\theta) = \frac{e^\theta}{1 + e^\theta}, \quad \theta = h(p) = \log\left(\frac{p}{1-p}\right),$$

for effective domain  $\Theta = \mathbb{R}$ , dual parameter space  $\mathcal{M} = (0, 1)$  and support  $\mathfrak{T} = \{0, 1\}$  of  $Y = T(Y)$ . With these choices we have

$$dF(y; \theta) = \exp\{\theta y - \log(1 + e^\theta)\} d\nu(y) = \left(\frac{e^\theta}{1 + e^\theta}\right)^y \left(\frac{1}{1 + e^\theta}\right)^{1-y} d\nu(y).$$

$\theta \mapsto \kappa'(\theta)$  is the logistic or sigmoid function, and the canonical link  $p \mapsto h(p)$  is the logit function. Mean and variance are given by

$$\mu = \mathbb{E}_\theta[Y] = \kappa'(\theta) = p \quad \text{and} \quad \text{Var}_\theta(Y) = \kappa''(\theta) = \frac{e^\theta}{(1 + e^\theta)^2} = p(1 - p),$$

and the probability weights satisfy for  $y \in \mathfrak{T} = \{0, 1\}$

$$\mathbb{P}_\theta[Y = y] = p^y (1 - p)^{1-y}.$$

#### Binomial Distribution as a Single-Parameter Linear EF

For the binomial distribution with parameters  $n \in \mathbb{N}$  and  $p \in (0, 1)$  we choose as  $\nu$  the counting measure on  $\{0, \dots, n\}$ . We make the following choices:  $T(y) = y$ ,

$$a(y) = \log\binom{n}{y}, \quad \kappa(\theta) = n \log(1 + e^\theta), \quad \mu = \kappa'(\theta) = \frac{ne^\theta}{1 + e^\theta}, \quad \theta = h(\mu) = \log\left(\frac{\mu}{n - \mu}\right),$$

for effective domain  $\Theta = \mathbb{R}$ , dual parameter space  $\mathcal{M} = (0, n)$  and support  $\mathfrak{T} = \{0, \dots, n\}$  of  $Y = T(Y)$ . With these choices we have

$$dF(y; \theta) = \binom{n}{y} \exp\{\theta y - n \log(1 + e^\theta)\} d\nu(y) = \binom{n}{y} \left(\frac{e^\theta}{1 + e^\theta}\right)^y \left(\frac{1}{1 + e^\theta}\right)^{n-y} d\nu(y).$$

Mean and variance are given by

$$\mu = \mathbb{E}_\theta [Y] = \kappa'(\theta) = np \quad \text{and} \quad \text{Var}_\theta (Y) = \kappa''(\theta) = n \frac{e^\theta}{(1 + e^\theta)^2} = np(1 - p),$$

where we set  $p = e^\theta / (1 + e^\theta)$ . The probability weights satisfy for  $y \in \mathfrak{T} = \{0, \dots, n\}$

$$\mathbb{P}_\theta[Y = y] = \binom{n}{y} p^y (1 - p)^{n-y}.$$

### Poisson Distribution as a Single-Parameter Linear EF

For the Poisson distribution with parameter  $\lambda > 0$  we choose as  $\nu$  the counting measure on  $\mathbb{N}_0$ . We make the following choices:  $T(y) = y$ ,

$$a(y) = \log\left(\frac{1}{y!}\right), \quad \kappa(\theta) = e^\theta, \quad \mu = \kappa'(\theta) = e^\theta, \quad \theta = h(\mu) = \log(\mu),$$

for effective domain  $\Theta = \mathbb{R}$ , dual parameter space  $\mathcal{M} = (0, \infty)$  and support  $\mathfrak{T} = \mathbb{N}_0$  of  $Y = T(Y)$ . With these choices we have

$$dF(y; \theta) = \frac{1}{y!} \exp\{\theta y - e^\theta\} d\nu(y) = e^{-\mu} \frac{\mu^y}{y!} d\nu(y). \quad (2.4)$$

The canonical link  $\mu \mapsto h(\mu)$  is the log-link. Mean and variance are given by

$$\mu = \mathbb{E}_\theta [Y] = \kappa'(\theta) = \lambda \quad \text{and} \quad \text{Var}_\theta (Y) = \kappa''(\theta) = \lambda = \mu = \mathbb{E}_\theta [Y],$$

where we set  $\lambda = e^\theta$ . The probability weights in the Poisson case satisfy for  $y \in \mathfrak{T} = \mathbb{N}_0$

$$\mathbb{P}_\theta[Y = y] = e^{-\lambda} \frac{\lambda^y}{y!}.$$

### Negative-Binomial (Pólya) Distribution as a Single-Parameter Linear EF

For the negative-binomial distribution with  $\alpha > 0$  and  $p \in (0, 1)$  we choose as  $\nu$  the counting measure on  $\mathbb{N}_0$ ;  $\alpha$  plays the role of a nuisance parameter or hyperparameter. We make the following choices:  $T(y) = y$ ,

$$a(y) = \log\left(\frac{y + \alpha - 1}{y}\right), \quad \kappa(\theta) = -\alpha \log(1 - e^\theta),$$

$$\mu = \kappa'(\theta) = \alpha \frac{e^\theta}{1 - e^\theta}, \quad \theta = h(\mu) = \log \left( \frac{\mu}{\mu + \alpha} \right),$$

for effective domain  $\Theta = (-\infty, 0)$ , dual parameter space  $\mathcal{M} = (0, \infty)$  and support  $\mathfrak{T} = \mathbb{N}_0$  of  $Y = T(Y)$ . With these choices we have

$$\begin{aligned} dF(y; \theta) &= \binom{y + \alpha - 1}{y} \exp \{ \theta y + \alpha \log(1 - e^\theta) \} dv(y) \\ &= \binom{y + \alpha - 1}{y} p^y (1 - p)^\alpha dv(y), \end{aligned}$$

with  $p = e^\theta$ . Parameter  $\alpha > 0$  is treated as nuisance parameter, otherwise we drop out of the EF framework. We have first the two moments

$$\mu = \mathbb{E}_\theta[Y] = \alpha \frac{e^\theta}{1 - e^\theta} = \alpha \frac{p}{1 - p} \quad \text{and} \quad \text{Var}_\theta(Y) = \mathbb{E}_\theta[Y] \left( 1 + \frac{e^\theta}{1 - e^\theta} \right) > \mathbb{E}_\theta[Y].$$

This model allows us to model over-dispersion, in contrast to the Poisson model. In fact, the negative-binomial model is a mixed Poisson model with a gamma mixing distribution, for details see Sect. 5.3.5, below. Typically, one uses a different parametrization. Set  $e^\theta = \lambda/(\alpha + \lambda)$ , for  $\lambda > 0$ . This implies

$$\mu = \mathbb{E}_\theta[Y] = \lambda \quad \text{and} \quad \text{Var}_\theta(Y) = \lambda \left( 1 + \frac{\lambda}{\alpha} \right) > \lambda.$$

For  $\alpha \in \mathbb{N}$  this model can also be interpreted as the waiting time until we observe  $\alpha$  successful trials among i.i.d. trials, for instance, for  $\alpha = 1$  we have the geometric distribution (with a small reparametrization).

The probability weights of the negative-binomial model satisfy for  $y \in \mathfrak{T} = \mathbb{N}_0$

$$\mathbb{P}_\theta[Y = y] = \binom{y + \alpha - 1}{y} p^y (1 - p)^\alpha. \quad (2.5)$$

### 2.1.3 Vector-Valued Parameter EF: Absolutely Continuous Examples

We give vector-valued parameter absolutely continuous EF examples with  $k = 2$ , and being based on the Lebesgue measure on (subsets of)  $\mathbb{R}$ , in this section.



### Gaussian Distribution as a Vector-Valued Parameter EF

For the Gaussian distribution with parameters  $\mu \in \mathbb{R}$  and  $\sigma^2 > 0$  we choose as  $\nu$  the Lebesgue measure on  $\mathbb{R}$ , and we make the following choices:  $T(y) = (y, y^2)^\top$ ,

$$a(y) = -\frac{1}{2}\log(2\pi), \quad \kappa(\theta) = -\frac{\theta_1^2}{4\theta_2} - \frac{1}{2}\log(-2\theta_2),$$

$$(\mu, \sigma^2 + \mu^2)^\top = \nabla_\theta \kappa(\theta) = \left( \frac{\theta_1}{-2\theta_2}, (-2\theta_2)^{-1} + \frac{\theta_1^2}{4\theta_2^2} \right)^\top,$$

for effective domain  $\Theta = \mathbb{R} \times (-\infty, 0)$ , dual parameter space  $\mathcal{M} = \mathbb{R} \times (0, \infty)$  and support  $\mathfrak{T} = \mathbb{R} \times [0, \infty)$  of  $T(Y) = (Y, Y^2)^\top$ . With these choices we have

$$dF(y; \theta) = \frac{1}{\sqrt{2\pi}} \exp \left\{ \theta^\top T(y) + \frac{\theta_1^2}{4\theta_2} + \frac{1}{2}\log(-2\theta_2) \right\} d\nu(y)$$

$$= \frac{1}{\sqrt{2\pi}(-2\theta_2)^{-1/2}} \exp \left\{ -\frac{1}{2} \frac{1}{(-2\theta_2)^{-1}} \left( y - \frac{\theta_1}{-2\theta_2} \right)^2 \right\} d\nu(y).$$

This is the Gaussian model with mean  $\mu = \theta_1/(-2\theta_2)$  and variance  $\sigma^2 = (-2\theta_2)^{-1}$ .

If we treat  $\sigma > 0$  as a nuisance parameter, we obtain the Gaussian model as a single-parameter EF. This is the most common example of an EF. Set  $T(y) = y/\sigma$  and

$$a(y) = -\frac{1}{2}\log(2\pi\sigma^2) - y^2/(2\sigma^2), \quad \kappa(\theta) = \theta^2/2, \quad \mu = \kappa'(\theta) = \theta, \quad \theta = h(\mu) = \mu,$$

for effective domain  $\Theta = \mathbb{R}$ , dual parameter space  $\mathcal{M} = \mathbb{R}$  and support  $\mathfrak{T} = \mathbb{R}$  of  $T(Y) = Y/\sigma$ . With these choices we have

$$dF(y; \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp \left\{ \theta y/\sigma - y^2/(2\sigma^2) - \theta^2/2 \right\} d\nu(y)$$

$$= \frac{1}{\sqrt{2\pi}\sigma} \exp \left\{ -\frac{1}{2\sigma^2} (y - \sigma\theta)^2 \right\} d\nu(y),$$

and, in particular, the canonical link is the *identity link*  $\mu \mapsto \theta = h(\mu) = \mu$  in this single-parameter EF example.

### Gamma Distribution as a Vector-Valued Parameter EF

For the gamma distribution with parameters  $\alpha, \beta > 0$  we choose as  $\nu$  the Lebesgue measure on  $\mathbb{R}_+$ . Then we make the following choices:  $T(y) = (y, \log y)^\top$ ,

$$a(y) = -\log y, \quad \kappa(\theta) = \log \Gamma(\theta_2) - \theta_2 \log(-\theta_1),$$

$$\left( \alpha/\beta, \frac{\Gamma'(\alpha)}{\Gamma(\alpha)} - \log(\beta) \right)^\top = \nabla_\theta \kappa(\theta) = \left( \frac{\theta_2}{-\theta_1}, \frac{\Gamma'(\theta_2)}{\Gamma(\theta_2)} - \log(-\theta_1) \right)^\top,$$

for effective domain  $\Theta = (-\infty, 0) \times (0, \infty)$ , and setting  $\beta = -\theta_1 > 0$  and  $\alpha = \theta_2 > 0$ . The dual parameter space is  $\mathcal{M} = (0, \infty) \times \mathbb{R}$ , and we have support  $\mathfrak{T} = (0, \infty) \times \mathbb{R}$  of  $T(Y) = (Y, \log Y)^\top$ . With these choices we obtain

$$\begin{aligned} dF(y; \theta) &= \exp \left\{ \theta^\top T(y) - \log \Gamma(\theta_2) + \theta_2 \log(-\theta_1) - \log y \right\} d\nu(y) \\ &= \frac{(-\theta_1)^{\theta_2}}{\Gamma(\theta_2)} y^{\theta_2-1} \exp \{ -(-\theta_1)y \} d\nu(y) \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} \exp \{ -\beta y \} d\nu(y). \end{aligned}$$

This is a vector-valued parameter EF with  $k = 2$ , and the first moment is given by

$$\mathbb{E}_\theta \left[ (Y, \log Y)^\top \right] = \nabla_\theta \kappa(\theta) = \left( \alpha/\beta, \frac{\Gamma'(\alpha)}{\Gamma(\alpha)} - \log(\beta) \right)^\top.$$

Parameter  $\alpha$  is called *shape parameter* and parameter  $\beta$  is called *scale parameter*.<sup>2</sup>

If we treat the shape parameter  $\alpha > 0$  as a nuisance parameter we can turn the gamma distribution into a single-parameter linear EF. Set  $T(y) = y$  and

$$a(y) = (\alpha - 1)\log y - \log \Gamma(\alpha), \quad \kappa(\theta) = -\alpha \log(-\theta), \quad \mu = \kappa'(\theta) = \frac{\alpha}{-\theta}, \quad \theta = h(\mu) = -\frac{\alpha}{\mu},$$

for effective domain  $\Theta = (-\infty, 0)$ , dual parameter space  $\mathcal{M} = (0, \infty)$  and support  $\mathfrak{T} = (0, \infty)$ . With these choices we have for  $\beta = -\theta > 0$

$$dF(y; \theta) = \frac{(-\theta)^\alpha}{\Gamma(\alpha)} y^{\alpha-1} \exp \{ -(-\theta)y \} d\nu(y). \quad (2.6)$$

This provides us with mean and variance

$$\mu = \mathbb{E}_\theta[Y] = \frac{\alpha}{\beta} \quad \text{and} \quad \sigma^2 = \text{Var}_\theta(Y) = \frac{\alpha}{\beta^2} = \frac{1}{\alpha} \mu^2.$$

<sup>2</sup> The function  $\Psi(x) = \frac{d}{dx} \log \Gamma(x) = \Gamma'(x)/\Gamma(x)$  is called digamma function.

For parameter estimation one often needs to invert these identities which gives us

$$\alpha = \frac{\mu^2}{\sigma^2} \quad \text{and} \quad \beta = \frac{\mu}{\sigma^2}.$$

*Remarks 2.10*

- The gamma distribution contains as special cases the exponential distribution for  $\alpha = \theta_2 = 1$  and  $\beta = -\theta_1 > 0$ , and the  $\chi_r^2$ -distribution with  $r$  degrees of freedom for  $\alpha = \theta_2 = r/2$  and  $\beta = -\theta_1 = 1/2$ .
- The distributions of the EF are all light-tailed in the sense that all moments of  $T(Y)$  exist. Therefore, the EF does not allow for regularly varying survival functions, see (1.3). If  $Y$  is gamma distributed, then  $Z = \exp\{Y\}$  is log-gamma distributed (with the special case of the Pareto distribution for the exponential case  $\alpha = \theta_2 = 1$ ). For an example we refer to Sect. 2.2.5. However, this log-transformation is not always recommended because it may provide accurate models on the transformed log-scale, but back-transformation to the original scale may not necessarily provide a good predictive model on that original scale.
- The gamma density (2.6) may be a bit tricky in applications because the effective domain  $\Theta = (-\infty, 0)$  is one-sided bounded (we come back to this below). For this reason, in practice, one often uses links different from the canonical link  $h(\mu) = -\alpha/\mu$ . For instance, a parametrization  $\theta = -\exp\{-\vartheta\}$  for  $\vartheta \in \mathbb{R}$ , see Ohlsson–Johansson [290], leads to the following model

$$dF(y; \vartheta) = \frac{y^{\alpha-1}}{\Gamma(\alpha)} \exp\{-e^{-\vartheta}y - \alpha\vartheta\} dv(y). \quad (2.7)$$

We will study the gamma model in more depth below, and parametrization (2.7) will correspond to the log-link choice, see Example 5.5, below.

Figure 2.1 gives examples of gamma densities for shape parameters  $\alpha \in \{1/2, 1, 3/2, 2\}$  and scale parameters  $\beta \in \{1/2, 1, 3/2, 2\}$  with  $\alpha = \beta$  all providing the same mean  $\mu = \mathbb{E}_\theta[Y] = \alpha/\beta = 1$ . The crucial observation is that these gamma densities can have two different shapes, for  $\alpha \leq 1$  we have a strictly decreasing shape and for  $\alpha > 1$  we have a unimodal density with mode in  $(\alpha - 1)/\beta$ .

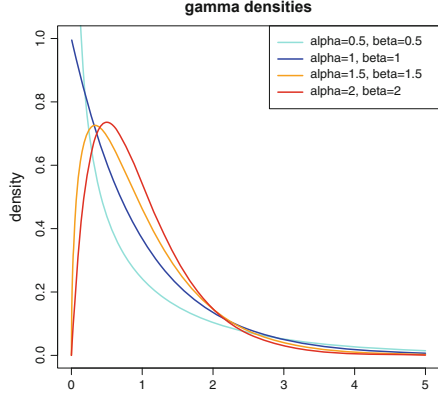
### Inverse Gaussian Distribution as a Vector-Valued Parameter EF

For the inverse Gaussian distribution with parameters  $\alpha, \beta > 0$  we choose as  $\nu$  the Lebesgue measure on  $\mathbb{R}_+$ . Then we make the following choices:  $T(y) = (y, 1/y)^\top$ ,

$$a(y) = -\frac{1}{2}\log(2\pi y^3), \quad \kappa(\theta) = -2(\theta_1\theta_2)^{1/2} - \frac{1}{2}\log(-2\theta_2),$$

$$\left(\alpha/\beta, \beta/\alpha + 1/\alpha^2\right)^\top = \nabla_\theta \kappa(\theta) = \left(\left(\frac{-2\theta_2}{-2\theta_1}\right)^{1/2}, \left(\frac{-2\theta_1}{-2\theta_2}\right)^{1/2} + \frac{1}{-2\theta_2}\right)^\top,$$

**Fig. 2.1** Gamma densities  
for shape parameters  
 $\alpha \in \{1/2, 1, 3/2, 2\}$  and scale  
parameters  
 $\beta \in \{1/2, 1, 3/2, 2\}$  all  
providing the same mean  
 $\mu = \alpha/\beta = 1$



for  $\theta = (\theta_1, \theta_2)^\top \in (-\infty, 0)^2$ , and setting  $\beta = (-2\theta_1)^{1/2}$  and  $\alpha = (-2\theta_2)^{1/2}$ . The dual parameter space is  $\mathcal{M} = (0, \infty)^2$ , and we have support  $\mathfrak{T} = (0, \infty)^2$  of  $T(Y) = (Y, 1/Y)^\top$ . With these choices we obtain

$$\begin{aligned}
 dF(y; \theta) &= \exp \left\{ \theta^\top T(y) + 2(\theta_1 \theta_2)^{1/2} + \frac{1}{2} \log(-2\theta_2) - \frac{1}{2} \log(2\pi y^3) \right\} dv(y) \\
 &= \frac{1}{(2\pi y^3)^{1/2}} (-2\theta_2)^{1/2} \exp \left\{ -\frac{1}{2y} \left( (-2\theta_1)y^2 + (-2\theta_2) - 4(\theta_1 \theta_2)^{1/2} y \right) \right\} dv(y) \\
 &= \frac{\alpha}{(2\pi y^3)^{1/2}} \exp \left\{ -\frac{\alpha^2}{2y} \left( 1 - \frac{\beta}{\alpha} y \right)^2 \right\} dv(y). \tag{2.8}
 \end{aligned}$$

This is a vector-valued parameter EF with  $k = 2$  and with first moment

$$\mathbb{E}_\theta \left[ (Y, 1/Y)^\top \right] = \nabla_\theta \kappa(\theta) = \left( \alpha/\beta, \beta/\alpha + 1/\alpha^2 \right)^\top.$$

For receiving (2.8) we have chosen canonical parameter  $\theta = (\theta_1, \theta_2)^\top \in (-\infty, 0)^2$ . Interestingly, we can close this parameter space for  $\theta_1 = 0$ , i.e., the effective domain  $\Theta$  is not open in this example. The choice  $\theta_1 = 0$  gives us cumulant function  $\kappa(\theta) = -\frac{1}{2} \log(-2\theta_2)$  and boundary case

$$\begin{aligned}
 dF(y; \theta) &= \exp \left\{ \theta^\top T(y) + \frac{1}{2} \log(-2\theta_2) - \frac{1}{2} \log(2\pi y^3) \right\} dv(y) \\
 &= \frac{1}{(2\pi y^3)^{1/2}} (-2\theta_2)^{1/2} \exp \left\{ -\frac{-2\theta_2}{2y} \right\} dv(y) \\
 &= \frac{\alpha}{(2\pi y^3)^{1/2}} \exp \left\{ -\frac{\alpha^2}{2y} \right\} dv(y). \tag{2.9}
 \end{aligned}$$

This is the distribution of the first-passage time of level  $\alpha > 0$  of a standard Brownian motion, see Bachelier [20]; this distribution is also known as Lévy distribution.

If we treat  $\alpha > 0$  as a nuisance parameter, we can turn the inverse Gaussian distribution into a single-parameter linear EF by setting  $T(y) = y$ ,

$$a(y) = \log \left( \frac{\alpha}{(2\pi y^3)^{1/2}} \right) - \frac{\alpha^2}{2y}, \quad \kappa(\theta) = -\alpha(-2\theta)^{1/2},$$

$$\mu = \kappa'(\theta) = \frac{\alpha}{(-2\theta)^{1/2}}, \quad \theta = h(\mu) = -\frac{1}{2} \frac{\alpha^2}{\mu^2},$$

for  $\theta \in (-\infty, 0)$ , dual parameter space  $\mathcal{M} = (0, \infty)$  and support  $\mathfrak{T} = (0, \infty)$ . With these choices we have the inverse Gaussian model for  $\beta = (-2\theta)^{1/2} > 0$

$$dF(y; \theta) = \exp\{a(y)\} \exp \left\{ -\frac{1}{2y} \left( (-2\theta)y^2 - 2\alpha(-2\theta)^{1/2}y \right) \right\} dv(y)$$

$$= \frac{\alpha}{(2\pi y^3)^{1/2}} \exp \left\{ -\frac{\alpha^2}{2y} \left( 1 - \frac{\beta}{\alpha} y \right)^2 \right\} dv(y).$$

This provides us with mean and variance

$$\mu = \mathbb{E}_\theta[Y] = \frac{\alpha}{\beta} \quad \text{and} \quad \sigma^2 = \text{Var}_\theta(Y) = \frac{\alpha}{\beta^3} = \frac{1}{\alpha^2} \mu^3.$$

For parameter estimation one often needs to invert these identities, which gives us

$$\alpha = \frac{\mu^{3/2}}{\sigma} \quad \text{and} \quad \beta = \frac{\mu^{1/2}}{\sigma}.$$

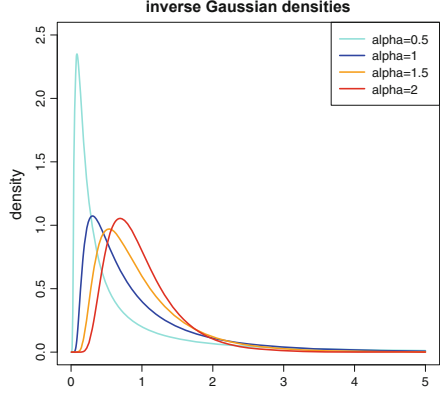
Figure 2.2 gives examples of inverse Gaussian densities for parameter choices  $\alpha = \beta \in \{1/2, 1, 3/2, 2\}$  all providing the same mean  $\mu = \mathbb{E}_\theta[Y] = \alpha/\beta = 1$ .

### Generalized Inverse Gaussian Distribution as a Vector-Valued Parameter EF

For the generalized inverse Gaussian distribution with parameters  $\alpha, \beta > 0$  and  $\gamma \in \mathbb{R}$  we choose as  $\nu$  the Lebesgue measure on  $\mathbb{R}_+$ . We combine the terms of the gamma and the inverse Gaussian models to the vector-valued choice:  $T(y) = (y, \log y, 1/y)^\top$  with  $k = 3$ . Moreover, we choose  $a(y) = -\log y$  and cumulant function

$$\kappa(\theta) = \log \left( 2K_{\theta_2}(2\sqrt{\theta_1\theta_3}) \right) - \frac{\theta_2}{2} \log(\theta_1/\theta_3),$$

**Fig. 2.2** Inverse Gaussian densities for parameters  $\alpha = \beta \in \{1/2, 1, 3/2, 2\}$  all providing the same mean  $\mu = \alpha/\beta = 1$



for  $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)^\top \in (-\infty, 0) \times \mathbb{R} \times (-\infty, 0)$ , and where  $K_{\theta_2}$  denotes the modified Bessel function of the second kind with index  $\gamma = \theta_2 \in \mathbb{R}$ . With these choices we obtain generalized inverse Gaussian density

$$\begin{aligned} dF(y; \boldsymbol{\theta}) &= \exp \left\{ \boldsymbol{\theta}^\top T(y) - \log \left( 2K_{\theta_2}(2\sqrt{\theta_1\theta_3}) \right) + \frac{\theta_2}{2} \log(\theta_1/\theta_3) - \log y \right\} dv(y) \\ &= \frac{(\alpha/\beta)^{\gamma/2}}{2K_\gamma(\sqrt{\alpha\beta})} y^{\gamma-1} \exp \left\{ -\frac{1}{2} (\alpha y + \beta y^{-1}) \right\} dv(y), \end{aligned} \quad (2.10)$$

setting  $\alpha = -2\theta_1$  and  $\beta = -2\theta_3$ . This is a vector-valued parameter EF with  $k = 3$ , and the first moment is given by

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\theta}} \left[ \left( Y, \log Y, \frac{1}{Y} \right)^\top \right] &= \nabla_{\boldsymbol{\theta}} \kappa(\boldsymbol{\theta}) \\ &= \left( \frac{K_{\gamma+1}(\sqrt{\alpha\beta})}{K_\gamma(\sqrt{\alpha\beta})} \sqrt{\frac{\beta}{\alpha}}, \log \sqrt{\frac{\beta}{\alpha}} + \frac{\partial}{\partial \gamma} \log K_\gamma(\sqrt{\alpha\beta}), \frac{K_{\gamma+1}(\sqrt{\alpha\beta})}{K_\gamma(\sqrt{\alpha\beta})} \sqrt{\frac{\alpha}{\beta}} - \frac{2\gamma}{\beta} \right)^\top. \end{aligned}$$

The effective domain  $\Theta$  is a bit complicated because the possible choices of  $(\theta_1, \theta_3)$  depend on  $\theta_2 \in \mathbb{R}$ , namely, for  $\theta_2 < 0$  the negative half-line  $(-\infty, 0]$  can be closed at the origin for  $\theta_1$ , and for  $\theta_2 > 0$  it can be closed at the origin for  $\theta_3$ . The inverse Gaussian model is obtained for  $\theta_2 = -1/2$  and the gamma model is obtained for  $\theta_3 = 0$ . For further properties of the generalized inverse Gaussian distribution we refer to the textbook of Jørgensen [200].

### 2.1.4 Vector-Valued Parameter EF: Count Variable Example

We close our EF examples by giving a discrete example with a vector-valued parameter.

#### Categorical Distribution as a Vector-Valued Parameter EF

For the categorical distribution with  $k \in \mathbb{N}$  and  $\mathbf{p} \in (0, 1)^k$  such that  $\sum_{i=1}^k p_i < 1$ , we choose as  $\nu$  the counting measure on the finite set  $\{1, \dots, k+1\}$ . Then we make the following choices:  $T(y) = (\mathbb{1}_{\{y=1\}}, \dots, \mathbb{1}_{\{y=k\}})^\top \in \mathbb{R}^k$ ,  $\boldsymbol{\theta} = (\theta_1, \dots, \theta_k)^\top$ ,  $e^\boldsymbol{\theta} = (e^{\theta_1}, \dots, e^{\theta_k})^\top$  and

$$a(y) = 0, \quad \kappa(\boldsymbol{\theta}) = \log \left( 1 + \sum_{i=1}^k e^{\theta_i} \right), \quad \mathbf{p} = \nabla_{\boldsymbol{\theta}} \kappa(\boldsymbol{\theta}) = \frac{e^\boldsymbol{\theta}}{1 + \sum_{i=1}^k e^{\theta_i}},$$

for effective domain  $\boldsymbol{\Theta} = \mathbb{R}^k$ , dual parameter space  $\mathcal{M} = (0, 1)^k$ , and the support  $\mathfrak{T}$  of  $T(Y)$  are the  $k+1$  corners of the unit simplex in  $\mathbb{R}^k$ . This representation is minimal, see Assumption 2.6. With these choices we have (set  $\theta_{k+1} = 0$ )

$$dF(y; \boldsymbol{\theta}) = \exp \left\{ \boldsymbol{\theta}^\top T(y) - \log \left( 1 + \sum_{i=1}^k e^{\theta_i} \right) \right\} d\nu(y) = \prod_{j=1}^{k+1} \left( \frac{e^{\theta_j}}{\sum_{i=1}^{k+1} e^{\theta_i}} \right)^{\mathbb{1}_{\{y=j\}}} d\nu(y).$$

This is a vector-valued parameter EF with  $k \in \mathbb{N}$ . The canonical link is slightly more complicated. Set vectors  $\mathbf{v} = \exp\{\boldsymbol{\theta}\} \in \mathbb{R}^k$  and  $\mathbf{w} = (1, \dots, 1)^\top \in \mathbb{R}^k$ . This provides  $\mathbf{p} = \nabla_{\boldsymbol{\theta}} \kappa(\boldsymbol{\theta}) = \frac{1}{1 + \mathbf{w}^\top \mathbf{v}} \mathbf{v} \in \mathbb{R}^k$ . Set matrix  $A_{\mathbf{p}} = \mathbb{1} - \mathbf{p} \mathbf{w}^\top \in \mathbb{R}^{k \times k}$ , the latter gives us  $\mathbf{p} = A_{\mathbf{p}} \mathbf{v}$ , and since  $A_{\mathbf{p}}$  has full rank  $k$ , we obtain canonical link

$$\mathbf{p} \mapsto \boldsymbol{\theta} = h(\mathbf{p}) = \log \left( A_{\mathbf{p}}^{-1} \mathbf{p} \right) = \log \left( \frac{\mathbf{p}}{1 - \mathbf{w}^\top \mathbf{p}} \right).$$

The last identity can be verified by explicit calculation

$$\log \left( \frac{\mathbf{p}}{1 - \mathbf{w}^\top \mathbf{p}} \right) = \log \left( \frac{e^\boldsymbol{\theta} / (1 + \sum_{j=1}^k e^{\theta_j})}{1 - \sum_{i=1}^k e^{\theta_i} / (1 + \sum_{j=1}^k e^{\theta_j})} \right) = \log \left( e^\boldsymbol{\theta} \right) = \boldsymbol{\theta}.$$

#### Remarks 2.11

- There are many more examples that belong to the EF. From Theorem 2.4, we know that all examples of the EF are light-tailed in the sense that all moments of  $T(Y)$  exist. If we want to model heavy-tailed distributions within the EF, we first need to apply a suitable transformation. We could model the Pareto distribution

using transformation  $T(y) = \log y$ , and assuming that the transformed random variable has an exponential distribution. Different light-tailed examples are obtained by, e.g., using transformation  $T(y) = y^\tau$  for the Weibull distribution or  $T(y) = (\log y, \log(1 - y))^\top$  for the beta distribution. We refrain from giving explicit formulas for these or other examples.

- Observe that in all examples above we have  $\mathfrak{T} \subset \overline{\mathcal{M}}$ , i.e., the support of  $T(Y)$  is contained in the closure of the dual parameter space  $\mathcal{M}$ , we come back to this observation in Sect. 2.2.4, below.

## 2.2 Exponential Dispersion Family

In the previous section we have introduced the EF, and we have explicitly studied the vector-valued parameter EF examples of the Gaussian, the gamma and the inverse Gaussian models. We have highlighted that these three vector-valued parameter EFs can be turned into single-parameter EFs by declaring one parameter to be a nuisance parameter that is not modeled (and acts as a hyper-parameter). This changes these three models into single-parameter EFs. These three single-parameter EFs with nuisance parameter can also be interpreted as EDF models. In this section we discuss the single-parameter EDF; this is sufficient for our purposes, and vector-valued parameter extensions can be obtained in a canonical way.

### 2.2.1 Definition and Properties

The EFs of Sect. 2.1 can be extended to EDFs. In the single-parameter case this is achieved by a transformation  $Y = X/\omega$ , where  $\omega > 0$  is a scaling and where  $X$  belongs to a single-parameter linear EF, i.e., with  $T(x) = x$ . We restrict ourselves to the single-parameter case  $k = 1$  throughout this section. Choose a  $\sigma$ -finite measure  $\nu_1$  on  $\mathbb{R}$  and a measurable function  $a_1 : \mathbb{R} \rightarrow \mathbb{R}$ . These choices give a single-parameter linear EF, directly modeling a real-valued random variable  $T(X) = X$ . By (2.2) we have distribution for the single-parameter linear EF random variable  $X$

$$dF(x; \theta, 1) = f(x; \theta, 1)d\nu_1(x) = \exp\{\theta x - \kappa(\theta) + a_1(x)\}d\nu_1(x),$$

on the effective domain

$$\Theta = \left\{ \theta \in \mathbb{R}; \int_{\mathbb{R}} \exp\{\theta x + a_1(x)\} d\nu_1(x) < \infty \right\}, \quad (2.11)$$



and with cumulant function

$$\theta \in \Theta \mapsto \kappa(\theta) = \log \left( \int_{\mathbb{R}} \exp \{ \theta x + a_1(x) \} d\nu_1(x) \right). \quad (2.12)$$

Throughout, we assume that the effective domain  $\Theta$  has a non-empty interior  $\overset{\circ}{\Theta}$ . Thus, since  $\Theta$  is convex, we assume that  $\overset{\circ}{\Theta}$  is a non-empty (possibly infinite) open interval in  $\mathbb{R}$ .

Following Jørgensen [201, 202], we extend this linear EF to an EDF as follows. Choose a family of  $\sigma$ -finite measures  $\nu_\omega$  on  $\mathbb{R}$  and measurable functions  $a_\omega : \mathbb{R} \rightarrow \mathbb{R}$  for a given index set  $\mathcal{W} \ni \omega$  with  $\{1\} \subset \mathcal{W} \subset \mathbb{R}_+$ . Assume that we have an  $\omega$ -independent scaled cumulant function  $\kappa$  on this index set  $\mathcal{W}$ , that is,

$$\theta \in \Theta \mapsto \kappa(\theta) = \frac{1}{\omega} \left( \log \int_{\mathbb{R}} \exp \{ \theta x + a_\omega(x) \} d\nu_\omega(x) \right) \quad \text{for all } \omega \in \mathcal{W},$$

with effective domain  $\Theta$  defined by (2.11), i.e., for  $\omega = 1$ . This allows us to consider the distribution functions

$$\begin{aligned} dF(x; \theta, \omega) &= f(x; \theta, \omega) d\nu_\omega(x) = \exp \left\{ \theta x - \omega \kappa(\theta) + a_\omega(x) \right\} d\nu_\omega(x) \\ &= \exp \left\{ \omega (\theta y - \kappa(\theta)) + a_\omega(\omega y) \right\} d\nu_\omega(\omega y), \end{aligned} \quad (2.13)$$

in the third identity we did a change of variable  $x \mapsto y = x/\omega$ . By re-parametrizing the function  $a_\omega(\omega \cdot)$  and the  $\sigma$ -finite measures  $\nu_\omega(\omega \cdot)$  slightly differently, depending on the particular structure of the chosen  $\sigma$ -finite measures, we arrive at the following single-parameter EDF.

**Definition 2.12** The (single-parameter) EDF is given by densities of the form

$$Y \sim f(y; \theta, v/\varphi) = \exp \left\{ \frac{y\theta - \kappa(\theta)}{\varphi/v} + a(y; v/\varphi) \right\}, \quad (2.14)$$

with

$\kappa : \Theta \rightarrow \mathbb{R}$  is the cumulant function (2.12),

$\theta \in \Theta$  is the canonical parameter in the effective domain (2.11),

$v > 0$  is a given weight (exposure, volume),

$\varphi > 0$  is the dispersion parameter,

$a(\cdot; \cdot)$  is the normalization, *not* depending on the canonical parameter  $\theta$ .

*Remarks 2.13*

- Exposure  $v > 0$  and dispersion parameter  $\varphi > 0$  provide the parametrization usually used for  $\omega = v/\varphi \in \mathcal{W}$ . Their meaning and interpretation will become clear below, and they will always appear as a ratio  $\omega = v/\varphi$ .
- The support of these EDF distributions does not depend on the explicit choice of the canonical parameter  $\theta \in \Theta$ , but it may depend on  $\omega = v/\varphi \in \mathcal{W}$  through the choices of the  $\sigma$ -finite measures  $\nu_\omega$ , for  $\omega \in \mathcal{W}$ . Consequently,  $a(y; \omega)$  is a normalization such that  $f(y; \theta, \omega)$  integrates to 1 w.r.t. the chosen  $\sigma$ -finite measure  $\nu_\omega$  to receive a proper distributional model.
- The transformation  $x \mapsto y = x/\omega$  in (2.13) is called duality transformation, see Section 3.1 in Jørgensen [203]. It provides the duality between the *additive form* (in variable  $x$  in (2.13)) and the *reproductive form* (in variable  $y$  in (2.13)) of the EDF; Definition 2.12 is the reproductive form.
- Lemma 2.1 tells us that  $\Theta$  is convex, thus, it is a possibly infinite interval in  $\mathbb{R}$ . To exclude trivial cases we will always assume that the  $\sigma$ -finite measure  $\nu_1$  is not concentrated in one single point (this relates to the minimal representation for  $k = 1$  in the linear EF case, see Assumption 2.6), and that the interior  $\mathring{\Theta}$  of the effective domain  $\Theta$  is non-empty.

**Corollary 2.14** *Assume  $\mathring{\Theta}$  is non-empty and that  $\nu_1$  is not concentrated in one single point. Choose  $Y \sim F(\cdot; \theta, v/\varphi)$  for fixed  $\theta \in \mathring{\Theta}$ . The moment generating function of  $Y$  for small  $r \in \mathbb{R}$  satisfies*

$$M_Y(r) = \mathbb{E}_\theta [\exp \{rY\}] = \exp \left\{ \frac{v}{\varphi} [\kappa(\theta + r\varphi/v) - \kappa(\theta)] \right\}.$$

*The first two moments of  $Y$  are given by*

$$\mu = \mathbb{E}_\theta [Y] = \kappa'(\theta) \quad \text{and} \quad \text{Var}_\theta (Y) = \frac{\varphi}{v} \kappa''(\theta) > 0.$$

*The cumulant function  $\kappa$  is smooth and strictly convex on  $\mathring{\Theta}$  with canonical link  $h = (\kappa')^{-1}$ . The variance function is defined by  $\mu \mapsto V(\mu) = (\kappa'' \circ h)(\mu)$  and, consequently, for the variance of  $Y$  we have  $\text{Var}_\mu (Y) = \frac{\varphi}{v} V(\mu)$  for  $\mu \in \mathcal{M}$ .*

**Proof** This follows analogously to Theorem 2.4. The linear case  $T(y) = y$  with  $\nu_1$  not being concentrated in one single point guarantees that the minimal dimension is  $k = 1$ , providing a minimal representation in this dimension, see Assumption 2.6.  $\square$

Before giving explicit examples we state the so-called convolution formula.

**Corollary 2.15 (Convolution Formula)** Assume  $\mathring{\Theta}$  is non-empty and that  $v_1$  is not concentrated in one single point. Assume that  $Y_i \sim F(\cdot; \theta, v_i/\varphi)$  are independent, for  $1 \leq i \leq n$ , with fixed  $\theta \in \mathring{\Theta}$ . Set  $v_+ = \sum_{i=1}^n v_i$ . Then

$$Y_+ = \frac{1}{v_+} \sum_{i=1}^n v_i Y_i \sim F(\cdot; \theta, v_+/\varphi).$$

**Proof** The proof immediately follows from calculating the moment generating function  $M_{Y_+}(r)$  and from using the independence between the  $Y_i$ 's.  $\square$

### 2.2.2 Exponential Dispersion Family Examples

The single-parameter linear EF examples introduced above can be reformulated as EDF examples.

#### Binomial Distribution as a Single-Parameter EDF

For the binomial distribution with parameters  $p \in (0, 1)$  and  $n \in \mathbb{N}$  we choose the counting measure on  $\{0, 1/n, \dots, 1\}$  with  $\omega = n$ . Then we make the following choices

$$a(y) = \log \binom{n}{ny}, \quad \kappa(\theta) = \log(1 + e^\theta), \quad p = \kappa'(\theta) = \frac{e^\theta}{1 + e^\theta}, \quad \theta = h(p) = \log \left( \frac{p}{1-p} \right),$$

for effective domain  $\Theta = \mathbb{R}$  and dual parameter space  $\mathcal{M} = (0, 1)$ . With these choices we have

$$f(y; \theta, n) = \binom{n}{ny} \exp \{ n (\theta y - \log(1 + e^\theta)) \} = \binom{n}{ny} \left( \frac{e^\theta}{1 + e^\theta} \right)^{ny} \left( \frac{1}{1 + e^\theta} \right)^{n-ny}.$$

This is a single-parameter EDF. The canonical link  $p \mapsto h(p)$  gives the logit function. Mean and variance are given by

$$p = \mathbb{E}_\theta[Y] = \kappa'(\theta) = \frac{e^\theta}{1 + e^\theta} \quad \text{and} \quad \text{Var}_\theta(Y) = \frac{1}{n} \kappa''(\theta) = \frac{1}{n} \frac{e^\theta}{(1 + e^\theta)^2} = \frac{1}{n} p(1 - p),$$

and the variance function is given by  $V(\mu) = \mu(1 - \mu)$ . The binomial random variable is obtained by setting  $X = nY \sim \text{Binom}(n, p)$ .

### Poisson Distribution as a Single-Parameter EDF

For the Poisson distribution with parameters  $\lambda > 0$  and  $v > 0$  we choose the counting measure on  $\mathbb{N}_0/v$  for exposure  $\omega = v$ . Then we make the following choices

$$a(y) = \log \left( \frac{v^{vy}}{(vy)!} \right), \quad \kappa(\theta) = e^\theta, \quad \lambda = \kappa'(\theta) = e^\theta, \quad \theta = h(\lambda) = \log(\lambda),$$

for effective domain  $\Theta = \mathbb{R}$  and dual parameter space  $\mathcal{M} = (0, \infty)$ . With these choices we have

$$f(y; \theta, v) = \frac{v^{vy}}{(vy)!} \exp \{v(\theta y - e^\theta)\} = e^{-v\lambda} \frac{(v\lambda)^{vy}}{(vy)!}. \quad (2.15)$$

This is a single-parameter EDF. The canonical link  $\lambda \mapsto h(\lambda)$  is the log-link. Mean and variance are given by

$$\lambda = \mathbb{E}_\theta[Y] = \kappa'(\theta) = e^\theta \quad \text{and} \quad \text{Var}_\theta(Y) = \frac{1}{v} \kappa''(\theta) = \frac{1}{v} e^\theta = \frac{1}{v} \lambda,$$

and the variance function is given by  $V(\lambda) = \lambda$ , that is, the variance function is linear in the mean parameter  $\lambda$ . The Poisson random variable is obtained by setting  $X = vY \sim \text{Poi}(v\lambda)$ . We choose  $\varphi = 1$ , here, meaning that we have neither under- nor over-dispersion. Thus, the choices  $v$  and  $\varphi$  in  $\omega = v/\varphi$  have the interpretation of an exposure and a dispersion parameter, respectively. This interpretation is going to be important in claim counts modeling, below.

### Gamma Distribution as a Single-Parameter EDF

For the gamma distribution with parameters  $\alpha, \beta > 0$  we choose the Lebesgue measure on  $\mathbb{R}_+$  and shape parameter  $\omega = v/\varphi = \alpha$ . We make the following choices

$$a(y) = (\alpha - 1)\log y + \alpha \log \alpha - \log \Gamma(\alpha), \quad \kappa(\theta) = -\log(-\theta), \\ \mu = \kappa'(\theta) = -1/\theta, \quad \theta = h(\mu) = -1/\mu,$$

for effective domain  $\Theta = (-\infty, 0)$  and dual parameter space  $\mathcal{M} = (0, \infty)$ . With these choices we have

$$f(y; \theta, \alpha) = \frac{\alpha^\alpha}{\Gamma(\alpha)} y^{\alpha-1} \exp \{ \alpha(y\theta + \log(-\theta)) \} = \frac{(-\theta\alpha)^\alpha}{\Gamma(\alpha)} y^{\alpha-1} \exp \{ -(-\theta\alpha)y \}.$$

This is analogous to (2.6) with shape parameter  $\alpha > 0$  and scale parameter  $\beta = -\theta > 0$ . Mean and variance are given by

$$\mu = \mathbb{E}_\theta [Y] = \kappa'(\theta) = -\theta^{-1} \quad \text{and} \quad \text{Var}_\theta (Y) = \frac{1}{\alpha} \kappa''(\theta) = \frac{1}{\alpha} \theta^{-2},$$

and the variance function is given by  $V(\mu) = \mu^2$ , that is, the variance function is quadratic in the mean parameter  $\mu$ . The gamma random variable is obtained by setting  $X = \alpha Y \sim \Gamma(\alpha, \beta)$ . This gives us for the first two moments of  $X$

$$\mu_X = \mathbb{E}_\theta [X] = \frac{\alpha}{\beta} \quad \text{and} \quad \text{Var}_\theta (X) = \frac{\alpha}{\beta^2} = \frac{1}{\alpha} \mu_X^2.$$

Suppose  $v = 1$ , for shape parameter  $\alpha > 1$ , we have under-dispersion  $\varphi = 1/\alpha < 1$  and the gamma density is unimodal; for shape parameter  $\alpha < 1$ , we have over-dispersion  $\varphi = 1/\alpha > 1$  and the gamma density is strictly decreasing, we refer to Fig. 2.1.

### Inverse Gaussian Distribution as a Single-Parameter EDF

For the inverse Gaussian distribution with parameters  $\alpha, \beta > 0$  we choose the Lebesgue measure on  $\mathbb{R}_+$  and we set  $\omega = v/\varphi = \alpha$ . We make the following choices

$$\begin{aligned} a(y) &= \log \left( \frac{\alpha^{1/2}}{(2\pi y^3)^{1/2}} \right) - \frac{\alpha}{2y}, \quad \kappa(\theta) = -(-2\theta)^{1/2}, \\ \mu &= \kappa'(\theta) = \frac{1}{(-2\theta)^{1/2}}, \quad \theta = h(\mu) = -\frac{1}{2\mu^2}, \end{aligned}$$

for  $\theta \in (-\infty, 0)$  and dual parameter space  $\mathcal{M} = (0, \infty)$ . With these choices we have

$$\begin{aligned} f(y; \theta, \alpha) dy &= \frac{\alpha^{1/2}}{(2\pi y^3)^{1/2}} \exp \left\{ \alpha \left( \theta y + (-2\theta)^{1/2} \right) - \frac{\alpha}{2y} \right\} dy \\ &= \frac{\alpha^{1/2}}{(2\pi y^3)^{1/2}} \exp \left\{ -\frac{\alpha}{2y} \left( 1 - (-2\theta)^{1/2} y \right)^2 \right\} dy \\ &= \frac{\alpha}{(2\pi x^3)^{1/2}} \exp \left\{ -\frac{\alpha^2}{2x} \left( 1 - \frac{(-2\theta)^{1/2}}{\alpha} x \right)^2 \right\} dx, \end{aligned}$$

where in the last step we did a change of variable  $y \mapsto x = \alpha y$ . This is exactly (2.8). Mean and variance are given by

$$\mu = \mathbb{E}_\theta [Y] = \kappa'(\theta) = (-2\theta)^{-1/2} \quad \text{and} \quad \text{Var}_\theta (Y) = \frac{1}{\alpha} \kappa''(\theta) = \frac{1}{\alpha} (-2\theta)^{-3/2},$$

and the variance function is given by  $V(\mu) = \mu^3$ , that is, the variance function is cubic in the mean parameter  $\mu$ . The inverse Gaussian random variable is obtained by setting  $X = \alpha Y$ . The mean and variance of  $X$  are given by, set  $\beta = (-2\theta)^{1/2} > 0$ ,

$$\mu_X = \mathbb{E}_\theta [X] = \frac{\alpha}{\beta} \quad \text{and} \quad \text{Var}_\theta (X) = \frac{\alpha}{\beta^3} = \frac{1}{\alpha^2} \mu_X^3.$$

This inverse Gaussian density is illustrated in Fig. 2.2.

Similarly to (2.9), we can extend the inverse Gaussian model to the boundary case  $\theta = 0$ , i.e., the effective domain  $\Theta = (-\infty, 0]$  is not open. This provides us with density

$$f(y; \theta = 0, \alpha) dy = \frac{\alpha}{(2\pi x^3)^{1/2}} \exp \left\{ -\frac{\alpha^2}{2x} \right\} dx, \quad (2.16)$$

using, as above, the change of variable  $y \mapsto x = \alpha y$ . An additional transformation  $x \mapsto 1/x$  gives a gamma distribution with shape parameter  $1/2$  and scale parameter  $\alpha^2/2$ .

*Remark 2.16* The inverse Gaussian case gives an example of a non-open effective domain  $\Theta = (-\infty, 0]$ . It is worth noting that for the boundary parameter  $\theta = 0$ , the first moment does not exist, i.e., Corollary 2.14 only makes statements in the interior  $\mathring{\Theta}$  of the effective domain  $\Theta$ . This also relates to Remarks 2.9 on the dual parameter space  $\mathcal{M}$ .

### 2.2.3 Tweedie's Distributions

Tweedie's compound Poisson (CP) model was introduced in 1984 by Tweedie [358], and it has been studied in detail in Jørgensen [202], Jørgensen–de Souza [204], Smyth–Jørgensen [342] and in the review paper of Delong et al. [94]. Tweedie's CP model belongs to the EDF. We spend more time on explaining Tweedie's CP model because it plays an important role in actuarial modeling.

Tweedie's CP model is received by choosing as  $\sigma$ -finite measure  $\nu_1$  a mixture of the Lebesgue measure on  $(0, \infty)$  and a point measure in 0. Furthermore, we choose *power variance parameter*  $p \in (1, 2)$  and cumulant function

$$\kappa(\theta) = \kappa_p(\theta) = \frac{1}{2-p} ((1-p)\theta)^{\frac{2-p}{1-p}}, \quad (2.17)$$

on the effective domain  $\theta \in \Theta = (-\infty, 0)$ . This provides us with Tweedie's CP model

$$Y \sim f(y; \theta, v/\varphi) = \exp \left\{ \frac{y\theta - \kappa_p(\theta)}{\varphi/v} + a(y; v/\varphi) \right\},$$

with exposure  $v > 0$  and dispersion parameter  $\varphi > 0$ ; the normalizing function  $a(\cdot; v/\varphi)$  does not have any simple closed form, we refer to Section 2.1 in Jørgensen–de Souza [204] and Section 4.2 in Jørgensen [203].

The first two moments of Tweedie's CP random variable  $Y$  are given by

$$\mu = \mathbb{E}_\theta[Y] = \kappa'_p(\theta) = ((1-p)\theta)^{\frac{1}{1-p}} \in \mathcal{M} = (0, \infty), \quad (2.18)$$

$$\text{Var}_\theta(Y) = \frac{\varphi}{v} \kappa''_p(\theta) = \frac{\varphi}{v} ((1-p)\theta)^{\frac{p}{1-p}} = \frac{\varphi}{v} \mu^p > 0. \quad (2.19)$$

The parameter  $p \in (1, 2)$  determines the power variance functions  $V(\mu) = \mu^p$  between the Poisson  $p = 1$  and the gamma  $p = 2$  cases, see Sect. 2.2.2.

The moment generating function of Tweedie's CP random variable  $X = vY/\varphi = \omega Y$  in its additive form is given by, we use Corollary 2.14,

$$M_X(r) = M_{vY/\varphi}(r) = \exp \left\{ \frac{v}{\varphi} \kappa_p(\theta) \left( \left( \frac{-\theta}{-\theta - r} \right)^{\frac{2-p}{p-1}} - 1 \right) \right\} \quad \text{for } r < -\theta.$$

Some readers will notice that this is the moment generating function of a CP distribution having i.i.d. gamma claim sizes. This is exactly the statement of the next proposition which is found, e.g., in Smyth–Jørgensen [342].

**Proposition 2.17** Assume  $S = \sum_{i=1}^N Z_i$  is CP distributed with Poisson claim counts  $N \sim \text{Poi}(\lambda v)$  and i.i.d. gamma claim sizes  $Z_i \sim \Gamma(\alpha, \beta)$  being independent of  $N$ . We have  $S \stackrel{(d)}{=} vY/\varphi$  by identifying the parameters as follows

$$p = \frac{\alpha + 2}{\alpha + 1} \in (1, 2), \quad \beta = -\theta > 0 \quad \text{and} \quad \lambda = \frac{1}{\varphi} \kappa_p(\theta) > 0.$$

**Proof of Proposition 2.17** Assume  $S$  is CP distributed with i.i.d. gamma claim sizes. From Proposition 2.11 and Section 3.2.1 in Wüthrich [387] we receive that the moment generating function of  $S$  is given by

$$M_S(r) = \exp \left\{ \lambda v \left( \left( \frac{\beta}{\beta - r} \right)^\alpha - 1 \right) \right\} \quad \text{for } r < \beta.$$

Using the proposed parameter identification, the claim immediately follows.  $\square$

Proposition 2.17 gives us a second interpretation of Tweedie's CP model which was introduced in an EDF fashion, above. This second interpretation explains the name of this EDF model, it explains the mixture of the Lebesgue measure and the point measure in 0, and it also highlights why the Poisson model and the gamma model are the boundary cases in terms of power variance functions.

An interesting question is whether the EDF can be extended beyond power variance functions  $V(\mu) = \mu^p$  with  $p \in [1, 2]$ . The answer to this question is yes, and the full answer is provided in Theorem 2 of Jørgensen [202]:

**Theorem 2.18 (Jørgensen [202], Without Proof)** *Only power variance parameters  $p \in (0, 1)$  do not allow for EDF models.*

Table 2.1 gives the EDF distributions that have a power variance function. These distributions are called *Tweedie's distributions*, with the special case of Tweedie's CP distributions for  $p \in (1, 2)$ . The densities for  $p \in \{0, 1, 2, 3\}$  have a closed form, but the other Tweedie's distributions do not have a closed-form density. Thus, they cannot explicitly be constructed as suggested in Sect. 2.2.1. Besides the constructive approach presented above, there is a uniqueness theorem saying that the variance function  $V(\cdot)$  on the domain  $\mathcal{M}$  characterizes the single-parameter linear EF, see Theorem 2.11 in Jørgensen [203]. This uniqueness theorem is the basis of the proof of Theorem 2.18. Tweedie's distributions for  $p \notin [0, 1] \cup \{2, 3\}$  involve infinite sums for the normalization  $\exp\{a(\cdot, \cdot)\}$ , we refer to formulas (4.19), (4.20) and (4.31) in Jørgensen [203], this is the reason that one has to go via the uniqueness theorem to prove Theorem 2.18. Dunn–Smyth [112] provide methods of fast calculation of some of these infinite sums; in Sect. 5.5.2, below, we present an approximation (saddlepoint approximation). The uniqueness theorem is also useful to construct new examples within the EF, see, e.g., Section 2 of Awad et al. [15].

**Table 2.1** Power variance function models  $V(\mu) = \mu^p$  within the EDF (taken from Table 4.1 in Jørgensen [203])

$p$	Distribution	Support of $Y$	$\Theta$	$\mathcal{M}$
$p < 0$	Generated by extreme stable distributions	$\mathbb{R}$	$[0, \infty)$	$(0, \infty)$
$p = 0$	Gaussian distribution	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$
$p = 1$	Poisson distribution	$\mathbb{N}_0$	$\mathbb{R}$	$(0, \infty)$
$1 < p < 2$	Tweedie's CP distribution	$[0, \infty)$	$(-\infty, 0)$	$(0, \infty)$
$p = 2$	Gamma distribution	$(0, \infty)$	$(-\infty, 0)$	$(0, \infty)$
$p > 2$	Generated by positive stable distributions	$(0, \infty)$	$(-\infty, 0]$	$(0, \infty)$
$p = 3$	Inverse Gaussian distribution	$(0, \infty)$	$(-\infty, 0]$	$(0, \infty)$



### 2.2.4 Steepness of the Cumulant Function

Assume we have a fixed EF satisfying Assumption 2.6. All random variables  $T(Y)$  belonging to this EF have the same support, not depending on the particular choice of the canonical parameter  $\theta \in \Theta$ . We denote this support of  $T(Y)$  by  $\mathfrak{T}$ .

Below, we are going to estimate the canonical parameter  $\theta \in \Theta$  from data using maximum likelihood estimation. For this it is advantageous to have the property  $\mathfrak{T} \subset \mathcal{M}$ , because, intuitively, this allows us to directly select  $\hat{\mu} = T(Y)$  as the parameter estimate in the dual parameter space  $\mathcal{M}$ , for a given observation  $T(Y) \in \mathfrak{T}$ . This then translates to a canonical parameter  $\hat{\theta} = h(\hat{\mu}) = h(T(Y)) \in \Theta$ , using the canonical link  $h$ ; this estimation approach will be better motivated in Chap. 3, below. Unfortunately, many examples of the EF do not satisfy this property  $\mathfrak{T} \subset \mathcal{M}$ . For instance, in the Poisson model the observation  $T(Y) = Y = 0$  is not included in  $\mathcal{M}$ , see Table 2.1. This poses some challenges in parameter estimation, and the purpose of this small discussion is to be prepared for these challenges.

A cumulant function  $\kappa$  is called *steep* if for all  $\theta \in \overset{\circ}{\Theta}$  and all  $\tilde{\theta}$  in the boundary of  $\Theta$

$$(\tilde{\theta} - \theta)^\top \nabla_{\theta} \kappa (\alpha \theta + (1 - \alpha) \tilde{\theta}) \rightarrow \infty \quad \text{for } \alpha \downarrow 0, \quad (2.20)$$

we refer to Formula (20) in Section 8.1 of Barndorff-Nielsen [23]. Define the convex closure of the support  $\mathfrak{T}$  by  $\mathfrak{C} = \overline{\text{conv}}(\mathfrak{T})$ .

**Theorem 2.19 (Theorem 9.2 in Barndorff-Nielsen [23], Without Proof)** *Assume we have a fixed EF satisfying Assumption 2.6. The cumulant function  $\kappa$  is steep if and only if  $\mathfrak{C} = \mathcal{M} = \nabla_{\theta} \kappa(\overset{\circ}{\Theta})$ .*

Theorem 2.19 tells us that for a steep cumulant function we have  $\mathfrak{C} = \overline{\mathcal{M}} = \nabla_{\theta} \kappa(\overset{\circ}{\Theta})$ . In this case parameter estimation can be extended to observations  $T(Y) \notin \mathcal{M}$  such that we may obtain a degenerate model at the boundary of  $\mathcal{M}$ . Coming back to our Poisson example from above, in this case we set  $\hat{\mu} = 0$ , which gives a degenerate Poisson model.

Throughout this book we will work under the assumption that  $\kappa$  is steep. The classical examples satisfy this assumption: the examples with power variance parameter  $p$  in  $\{0\} \cup [1, \infty)$  satisfy Theorem 2.19; this includes the Gaussian, the Poisson, the gamma, the inverse Gaussian and Tweedie's CP models, see Table 2.1. Moreover, the examples we have met in Sect. 2.1 fulfill this assumption; these are the single-parameter linear EF models of the Bernoulli, the binomial and the negative binomial distributions, as well as the vector-valued parameter examples of the Gaussian, the gamma and the inverse Gaussian models and of the categorical distribution. The only models we have seen that do not have a steep cumulant function are the power variance models with  $p < 0$ , see Table 2.1.

*Remark 2.20* Working within the EDF needs some additional thoughts because the support  $\mathfrak{T} = \mathfrak{T}_{\omega}$  of the single-parameter linear EDF random variable  $Y = T(Y)$  may

depend on the specific choice of the dispersion parameter  $\omega \in \mathcal{W} \supset \{1\}$  through the  $\sigma$ -finite measure  $d\nu_\omega(\omega \cdot)$ , see (2.13). For instance, in the binomial case the support of  $Y$  is given by  $\mathfrak{T}_\omega = \{0, 1/n, \dots, 1\}$  with  $\omega = n$ , see Sect. 2.2.2.

Assume that the cumulant function  $\kappa$  is steep for the single-parameter linear EF that corresponds to the single-parameter EDF with  $\omega = 1$ . Theorem 2.19 then implies that for this choice we have  $\mathfrak{C}_{\omega=1} = \nabla_\theta \kappa(\dot{\Theta})$  with convex closure  $\mathfrak{C}_{\omega=1} = \overline{\text{conv}}(\mathfrak{T}_{\omega=1})$ .

Consider  $\omega \in \mathcal{W} \setminus \{1\}$  which corresponds to the choice  $\nu_\omega$  of the  $\sigma$ -finite measure on  $\mathbb{R}$ . This choice belongs to the cumulant function  $\theta \mapsto \omega \kappa(\theta)$  in the additive form ( $x$ -parametrization in (2.13)). Since steepness (2.20) holds for any  $\omega > 0$  we receive that the convex closure of the support of this distribution in the  $x$ -parametrization in (2.13) is given by  $\overline{\nabla_\theta \omega \kappa(\dot{\Theta})} = \omega \overline{\nabla_\theta \kappa(\dot{\Theta})}$ . The duality transformation  $x \mapsto y = x/\omega$  leads to the change of measure  $d\nu_\omega(x) \mapsto d\nu_\omega(\omega y)$  and to the corresponding change of support, see (2.13). The latter implies that in the reproductive form ( $y$ -parametrization) the convex closure of the support does not depend on the specific choice of  $\omega \in \mathcal{W}$ . Since the EDF representation given in (2.14) corresponds to the  $y$ -parametrization (reproductive form), we can use Theorem 2.19 without limitation also for the single-parameter linear EDF given by (2.14), and  $\mathfrak{C}$  does not depend on  $\omega \in \mathcal{W}$ .

### 2.2.5 Lab: Large Claims Modeling

From Corollary 2.14 we know that the moment generating function exists around the origin for all examples belonging to the EDF. This implies that the moments of all orders exist, and that we have an exponentially decaying survival function  $\mathbb{P}_\theta[Y > y] = 1 - F(y; \theta, \omega) \sim \exp\{-\varrho y\}$  for some  $\varrho > 0$  as  $y \rightarrow \infty$ , see (1.2). In many applied situations the data is more heavy-tailed and, thus, cannot be modeled by such an exponentially decaying survival function. In such cases one often chooses a distribution function with a regularly varying survival function; regular variation with tail index  $\beta > 0$  has been introduced in (1.3). A popular choice is a log-gamma distribution which can be obtained from the gamma distribution (belonging to the EDF). We briefly explain how this is done and how it relates to the Pareto and the Lomax [256] distributions.

We start from the gamma density (2.6). The random variable  $Z$  has a log-gamma distribution with shape parameter  $\alpha > 0$  and scale parameter  $\beta = -\theta > 0$  if  $\log(Z) = Y$  has a gamma distribution with these parameters. Thus, the gamma density of  $Y = \log(Z)$  is given by

$$f(y; \beta, \alpha) dy = \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} \exp\{-\beta y\} dy \quad \text{for } y > 0.$$

We do a change of variable  $y \mapsto z = \exp\{y\}$  to receive the density of the log-gamma distributed random variable  $Z = \exp\{Y\}$

$$f(z; \beta, \alpha) dz = \frac{\beta^\alpha}{\Gamma(\alpha)} (\log z)^{\alpha-1} z^{-(\beta+1)} dz \quad \text{for } z > 1.$$

This log-gamma density has support  $(1, \infty)$ . The distribution function of this log-gamma distributed random variable needs to be calculated numerically, and its survival function is regularly varying with tail index  $\beta > 0$ .

A special case of the log-gamma distribution is the Pareto distribution. The Pareto distribution is more tractable and it is obtained by setting shape parameter  $\alpha = 1$  in the log-gamma density. This gives us the Pareto density

$$f(z; \beta) dz = f(z; \beta, \alpha = 1) dz = \beta z^{-(\beta+1)} dz \quad \text{for } z > 1.$$

The distribution function in this Pareto case is for  $z \geq 1$  given by

$$F(z; \beta) = 1 - z^{-\beta}.$$

Obviously, this provides a regularly varying survival function with tail index  $\beta > 0$ ; in fact, in this case we do not need to go over to the limit in (1.3) because we have an exact identity. The Pareto distribution has the nice property that it is closed under thresholding (lower-truncation) with  $M$ , that is, we remain within the family of Pareto distributions with the same tail index  $\beta$  by considering lower-truncated claims: for  $1 \leq M \leq z$  we have

$$F(z; \beta, M) = \mathbb{P}[Z \leq z | Z > M] = \frac{\mathbb{P}[M < Z \leq z]}{\mathbb{P}[Z > M]} = 1 - \left(\frac{z}{M}\right)^{-\beta}.$$

This is the classical definition of the Pareto distribution, and it allows to preserve full flexibility in the choice of the threshold  $M > 0$ .

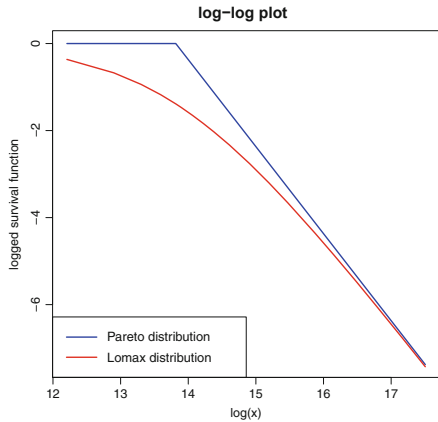
The disadvantage of the Pareto distribution is that it does not provide a continuous density on  $\mathbb{R}_+$  as there is a discontinuity in threshold  $M$ . For this reason, one sometimes explores another change of variable  $Z \mapsto X = Z - M$  for a Pareto distributed random variable  $Z \sim F(\cdot; \beta, M)$ . This provides the Lomax distribution, also called Pareto Type II distribution.  $X$  has the following distribution function on  $(0, \infty)$

$$\mathbb{P}[X \leq x] = 1 - \left(\frac{x + M}{M}\right)^{-\beta} \quad \text{for } x \geq 0.$$

This distribution has again a regularly varying survival function with tail index  $\beta > 0$ . Moreover, we have

$$\lim_{x \rightarrow \infty} \frac{\left(\frac{x+M}{M}\right)^{-\beta}}{\left(\frac{x}{M}\right)^{-\beta}} = \lim_{x \rightarrow \infty} \left(1 + \frac{M}{x}\right)^{-\beta} = 1.$$

**Fig. 2.3** Log-log plot of a Pareto and a Lomax distribution with tail index  $\beta = 2$  and threshold  $M = 1'000'000$



This says that we should choose the same threshold  $M > 0$  for both the Pareto and the Lomax distribution to receive the same asymptotic tail behavior, and this also quantifies the rate of convergence between the two survival functions. Figure 2.3 illustrates this convergence in a log-log plot choosing tail index  $\beta = 2$  and threshold  $M = 1'000'000$ .

For completeness we provide the density of the Pareto distribution

$$f(z; \beta, M) = \frac{\beta}{M} \left( \frac{z}{M} \right)^{-(\beta+1)} \quad \text{for } z \geq M,$$

and of the Lomax distribution

$$f(x; \beta, M) = \frac{\beta}{M} \left( \frac{x + M}{M} \right)^{-(\beta+1)} \quad \text{for } x \geq 0.$$

## 2.3 Information Geometry in Exponential Families

We do a short excursion to information geometry. This excursion may look a bit disconnected from what we have done so far, but it provides us with important background information for the chapter on forecast evaluation, see Chap. 4, below.

### 2.3.1 Kullback–Leibler Divergence

There is literature in information geometry which uses techniques from differential geometry to study EFs as Riemannian manifolds with points corresponding to EF densities parametrized by their canonical parameters  $\theta \in \Theta$ , we refer to Amari [10],

Ay et al. [16] and Nielsen [285] for an extended treatment of these mathematical concepts.

Choose a fixed EF (2.2) with cumulant function  $\kappa$  on the effective domain  $\Theta \subseteq \mathbb{R}^k$  and with  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$ . We define the Kullback–Leibler (KL) divergence (relative entropy) from model  $\theta_1 \in \Theta$  to model  $\theta_0 \in \Theta$  within this EF by

$$D_{\text{KL}}(f(\cdot; \theta_0) || f(\cdot; \theta_1)) = \int_{\mathbb{R}} f(y; \theta_0) \log \left( \frac{f(y; \theta_0)}{f(y; \theta_1)} \right) d\nu(y) \geq 0.$$

Recall that the support of the EF does not depend on the specific choice of the canonical parameter  $\theta$  in  $\Theta$ , see Remarks 2.3; this implies that the KL divergence is well-defined, here. The positivity of the KL divergence is obtained from Jensen's inequality; this is proved in Lemma 2.21, below.

The KL divergence has the interpretation of having a data model that is characterized by the distribution  $f(\cdot; \theta_0)$ , and we would like to measure how close another model  $f(\cdot; \theta_1)$  is to the data model. Note that the KL divergence is not a distance function because it is neither symmetric nor does it satisfy the triangle inequality.

We calculate the KL divergence within the chosen EF

$$\begin{aligned} D_{\text{KL}}(f(\cdot; \theta_0) || f(\cdot; \theta_1)) &= \int_{\mathbb{R}} f(y; \theta_0) \left[ (\theta_0 - \theta_1)^\top T(y) - \kappa(\theta_0) + \kappa(\theta_1) \right] d\nu(y) \\ &= (\theta_0 - \theta_1)^\top \nabla_{\theta} \kappa(\theta_0) - \kappa(\theta_0) + \kappa(\theta_1) \geq 0, \end{aligned} \quad (2.21)$$

where we have used Corollary 2.5, and the positivity of the KL divergence can be seen from the convexity of  $\kappa$ . This allows us to consider the following (Taylor) expansion

$$\kappa(\theta_1) = \kappa(\theta_0) + \nabla_{\theta} \kappa(\theta_0)^\top (\theta_1 - \theta_0) + D_{\text{KL}}(f(\cdot; \theta_0) || f(\cdot; \theta_1)). \quad (2.22)$$

This illustrates that the KL divergence corresponds to second and higher order differences between the cumulant value  $\kappa(\theta_0)$  and another cumulant value  $\kappa(\theta_1)$ . The gradients of the KL divergence w.r.t.  $\theta_1$  in  $\theta_1 = \theta_0$  and w.r.t.  $\theta_0$  in  $\theta_0 = \theta_1$  are given by

$$\begin{aligned} \nabla_{\theta_1} D_{\text{KL}}(f(\cdot; \theta_0) || f(\cdot; \theta_1)) \Big|_{\theta_1=\theta_0} \\ = \nabla_{\theta_0} D_{\text{KL}}(f(\cdot; \theta_0) || f(\cdot; \theta_1)) \Big|_{\theta_0=\theta_1} = \mathbf{0}. \end{aligned} \quad (2.23)$$

This emphasizes that the KL divergence reflects second and higher-order terms in cumulant function  $\kappa$ ; and that the data model  $\theta_0$  forms the minimum of this KL

divergence (as a function of  $\theta_1$ ) as we will just see. We calculate the Hessian (second order term) w.r.t.  $\theta_1$  in  $\theta_1 = \theta_0$

$$\nabla_{\theta_1}^2 D_{\text{KL}}(f(\cdot; \theta_0) || f(\cdot; \theta_1)) \Big|_{\theta_1 = \theta_0} = \nabla_{\theta}^2 \kappa(\theta) \Big|_{\theta = \theta_0} \stackrel{\text{def.}}{=} \mathcal{I}(\theta_0).$$

The positive definite matrix  $\mathcal{I}(\theta_0)$  (in a minimal representation) is called *Fisher's information*. Fisher's information is an important tool in statistics that we will meet in Theorem 3.13 of Sect. 3.3, below. A function satisfying (2.21) (with being zero if and only if  $\theta_0 = \theta_1$ ), fulfilling (2.23) and having positive definite Fisher's information is called *divergence*, see Definition 5 in Nielsen [285]. Fisher's information  $\mathcal{I}(\theta_0)$  measures the curvature of the KL divergence in  $\theta_0$  and we have the second order Taylor approximation

$$\kappa(\theta_1) \approx \kappa(\theta_0) + \nabla_{\theta} \kappa(\theta_0)^{\top} (\theta_1 - \theta_0) + \frac{1}{2} (\theta_1 - \theta_0)^{\top} \mathcal{I}(\theta_0) (\theta_1 - \theta_0).$$

Next-order terms are obtained from the so-called Amari–Chentsov tensor, see Amari [10] and Section 4.2 in Ay et al. [16]. In information geometry one studies the (possibly degenerate) Riemannian metric on the effective domain  $\Theta$  induced by Fisher's information; we refer to Section 3.7 in Nielsen [285].

**Lemma 2.21** *Consider two densities  $p$  and  $q$  w.r.t. a given  $\sigma$ -finite measure  $\nu$ . We have  $D_{\text{KL}}(p||q) \geq 0$ , and  $D_{\text{KL}}(p||q) = 0$  if and only if  $p = q$ ,  $\nu$ -a.s.*

**Proof** Assume  $Y \sim p d\nu$ , then we can rewrite the KL divergence, using Jensen's inequality,

$$\begin{aligned} D_{\text{KL}}(p||q) &= \int p(y) \log \left( \frac{p(y)}{q(y)} \right) d\nu(y) = -\mathbb{E}_p \left[ \log \left( \frac{q(Y)}{p(Y)} \right) \right] \\ &\geq -\log \mathbb{E}_p \left[ \frac{q(Y)}{p(Y)} \right] = -\log \int q(y) d\nu(y) \geq 0. \end{aligned} \quad (2.24)$$

Equality holds if and only if  $p = q$ ,  $\nu$ -a.s. The last inequality of (2.24) considers that  $q$  does not necessarily need to be a density w.r.t.  $\nu$ , i.e., we can also have  $\int q(y) d\nu(y) < 1$ .  $\square$

### 2.3.2 Unit Deviance and Bregman Divergence

In the next chapter we are going to introduce maximum likelihood estimation for parameters, see Definition 3.4, below. Maximum likelihood estimators are obtained by maximizing likelihood functions (evaluated in the observations). Maximizing likelihood functions within the EDF is equivalent to minimizing deviance loss

functions. Deviance loss functions are based on unit deviances, which, in turn, correspond to KL divergences. The purpose of this small section is to discuss this relation. This should be viewed as a preparation for Chap. 4.

Assume we work within a single-parameter linear EDF, i.e.,  $T(y) = y$ . Using the canonical link  $h$  we obtain the canonical parameter  $\theta = h(\mu) \in \Theta \subseteq \mathbb{R}$  from the mean parameter  $\mu \in \mathcal{M}$ . If we replace the (typically unknown) mean parameter  $\mu$  by an observation  $Y$ , supposed  $Y \in \mathcal{M}$ , we get the specific model that is exactly calibrated to this observation. This provides us with the canonical parameter estimate  $\hat{\theta}_Y = h(Y)$  for  $\theta$ . We can now measure the KL divergence from any model represented by  $\theta$  to the observation calibrated model  $\hat{\theta}_Y = h(Y)$ . This KL divergence is given by (we use (2.21) and we set  $\omega = v/\varphi = 1$ )

$$\begin{aligned} D_{\text{KL}}(f(\cdot; h(Y), 1) || f(\cdot; \theta, 1)) &= \int_{\mathbb{R}} f(y; \hat{\theta}_Y, 1) \log \left( \frac{f(y; \hat{\theta}_Y, 1)}{f(y; \theta, 1)} \right) dv(y) \\ &= (h(Y) - \theta) Y - \kappa(h(Y)) + \kappa(\theta) \geq 0. \end{aligned}$$

This latter object is the unit deviance (up to factor 2) of the chosen EDF. It plays a crucial role in predictive modeling.

We define the *unit deviance* under the assumption that  $\kappa$  is steep as follows:

$$\begin{aligned} \mathfrak{d} : \mathring{\mathfrak{C}} \times \mathcal{M} &\rightarrow \mathbb{R}_+ \\ (y, \mu) &\mapsto \mathfrak{d}(y, \mu) = 2 \left( yh(y) - \kappa(h(y)) - yh(\mu) + \kappa(h(\mu)) \right) \geq 0, \end{aligned} \quad (2.25)$$

where  $\mathfrak{C}$  is the convex closure of the support  $\mathfrak{T}$  of  $Y$  and  $\mathcal{M}$  is the dual parameter space of the chosen EDF. Steepness of  $\kappa$  implies  $\mathring{\mathfrak{C}} = \mathcal{M}$ , see Theorem 2.19.

This unit deviance  $\mathfrak{d}$  is received from the KL divergence, and it is (twice) the difference of two log-likelihood functions, one using canonical parameter  $h(y)$  and the other one having any canonical parameter  $\theta = h(\mu) \in \Theta$ . That is, for  $\mu = \kappa'(\theta)$ ,

$$\begin{aligned} \mathfrak{d}(y, \mu) &= 2 D_{\text{KL}}(f(\cdot; h(y), 1) || f(\cdot; \theta, 1)) \\ &= 2 \frac{\varphi}{v} (\log f(y; h(y), v/\varphi) - \log f(y; \theta, v/\varphi)), \end{aligned} \quad (2.26)$$

for general  $\omega = v/\varphi \in \mathcal{W}$ . The latter can be rewritten as

$$f(y; \theta, v/\varphi) = f(y; h(y), v/\varphi) \exp \left\{ -\frac{1}{2\varphi/v} \mathfrak{d}(y, \kappa'(\theta)) \right\}. \quad (2.27)$$

This looks like a generalization of the Gaussian distribution, where the square difference  $(y - \mu)^2$  in the exponent is replaced by the unit deviance  $\mathfrak{d}(y, \mu)$  with  $\mu = \kappa'(\theta)$ . This interpretation gets further support by the following lemma.

**Lemma 2.22** *Under Assumption 2.6 and the assumption that the cumulant function  $\kappa$  is steep, the unit deviance  $\mathfrak{d}(y, \mu) \geq 0$  of the chosen EDF is zero if and only if  $y = \mu$ . Moreover, the unit deviance  $\mathfrak{d}(y, \mu)$  is twice continuously differentiable w.r.t.  $(y, \mu)$  in  $\mathring{\mathfrak{C}} \times \mathcal{M}$ , and*

$$\left. \frac{\partial^2 \mathfrak{d}(y, \mu)}{\partial \mu^2} \right|_{y=\mu} = \left. \frac{\partial^2 \mathfrak{d}(y, \mu)}{\partial y^2} \right|_{y=\mu} = - \left. \frac{\partial^2 \mathfrak{d}(y, \mu)}{\partial \mu \partial y} \right|_{y=\mu} = 2/V(\mu) > 0.$$

**Proof** The positivity and the if and only if statement follows from Lemma 2.21 and the strict convexity of  $\kappa$ . Continuous differentiability follows from the smoothness of  $\kappa$  in the interior of  $\Theta$ . Moreover we have

$$\left. \frac{\partial^2 \mathfrak{d}(y, \mu)}{\partial \mu^2} \right|_{y=\mu} = \left. \frac{\partial}{\partial \mu} 2(-yh'(\mu) + \mu h'(\mu)) \right|_{y=\mu} = 2h'(\mu) = 2/\kappa''(h(\mu)) = 2/V(\mu) > 0,$$

where  $V(\mu)$  is the variance function of the chosen EDF introduced in Corollary 2.14. The remaining second derivatives are received by similar (straightforward) calculations.  $\square$

### Remarks 2.23

- Lemma 2.22 shows that the unit deviance definition of  $\mathfrak{d}(y, \mu)$  provides a so-called regular unit deviance according to Definition 1.1 in Jørgensen [203]. Moreover, any model that can be brought into the form (2.27) for a (regular) unit deviance is called (regular) reproductive dispersion model, see Definition 1.2 of Jørgensen [203].
- In general the unit deviance  $\mathfrak{d}(y, \mu)$  is not symmetric in its two arguments  $y$  and  $\mu$ , we come back to this in Fig. 11.1, below.

More generally, the KL divergence and the unit deviance can be embedded into the framework of Bregman loss functions [50]. We restrict to the single-parameter EDF case. Assume that  $\psi : \mathring{\mathfrak{C}} \rightarrow \mathbb{R}$  is a strictly convex function. The *Bregman divergence* w.r.t.  $\psi$  between  $y$  and  $\mu$  is defined by

$$D_\psi(y, \mu) = \psi(y) - \psi(\mu) - \psi'(\mu)(y - \mu) \geq 0, \quad (2.28)$$

where  $\psi'$  is a (sub-)gradient of  $\psi$ . The lower bound holds because of convexity of  $\psi$ . Consider the specific choice  $\psi(\mu) = \mu h(\mu) - \kappa(h(\mu))$  for the chosen EDF. Similar to Lemma 2.22 we have  $\psi''(\mu) = h'(\mu) = 1/V(\mu) > 0$ , which says that this choice is strictly convex. Using this choice for  $\psi$  gives us unit deviance (up to factor 1/2)

$$D_\psi(y, \mu) = yh(y) - \kappa(h(y)) + \kappa(h(\mu)) - h(\mu)y = \frac{1}{2}\mathfrak{d}(y, \mu). \quad (2.29)$$



Thus, the unit deviance  $\mathfrak{d}$  can be understood as a difference of log-likelihoods (2.26), as a KL divergence  $D_{\text{KL}}$  and as a Bregman divergence  $D_\psi$ .

*Example 2.24 (Poisson Model)* We start with a single-parameter EF example. Consider cumulant function  $\kappa(\theta) = \exp\{\theta\}$  for canonical parameter  $\theta \in \Theta = \mathbb{R}$ , this gives us the Poisson model. For the KL divergence from model  $\theta_1$  to model  $\theta_0$  we receive

$$D_{\text{KL}}(f(\cdot; \theta_0) || f(\cdot; \theta_1)) = \exp\{\theta_1\} - \exp\{\theta_0\} - (\theta_1 - \theta_0) \exp\{\theta_0\} \geq 0,$$

which is zero if and only if  $\theta_0 = \theta_1$ . Fisher's information is given by

$$\mathcal{I}(\theta) = \kappa''(\theta) = \exp\{\theta\} > 0.$$

If we have observation  $Y > 0$  we receive a model described by canonical parameter  $\hat{\theta}_Y = h(Y) = \log(Y)$ . This gives us unit deviance, see (2.26),

$$\begin{aligned} \mathfrak{d}(Y, \mu) &= 2D_{\text{KL}}(f(\cdot; h(Y), 1) || f(\cdot; \theta, 1)) \\ &= 2(e^\theta - Y - (\theta - \log(Y))Y) \\ &= 2\left(\mu - Y - Y \log\left(\frac{\mu}{Y}\right)\right) \geq 0, \end{aligned}$$

with  $\mu = \kappa'(\theta) = \exp\{\theta\}$ . This Poisson unit deviance will commonly be used for model fitting and forecast evaluation, see, e.g., (5.28). ■

*Example 2.25 (Gamma Model)* The second example considers a vector-valued parameter EF example. We consider the cumulant function  $\kappa(\boldsymbol{\theta}) = \log\Gamma(\theta_2) - \theta_2 \log(-\theta_1)$  for  $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top \in \Theta = (-\infty, 0) \times (0, \infty)$ ; this gives us the gamma model, see Sect. 2.1.3. For the KL divergence from model  $\boldsymbol{\theta}_1$  to model  $\boldsymbol{\theta}_0$  we receive

$$\begin{aligned} D_{\text{KL}}(f(\cdot; \boldsymbol{\theta}_0) || f(\cdot; \boldsymbol{\theta}_1)) &= (\theta_{0,2} - \theta_{1,2}) \frac{\Gamma'(\theta_{0,2})}{\Gamma(\theta_{0,2})} - \log\left(\frac{\Gamma(\theta_{0,2})}{\Gamma(\theta_{1,2})}\right) \\ &\quad + \theta_{1,2} \log\left(\frac{-\theta_{0,1}}{-\theta_{1,1}}\right) + \theta_{0,2} \left(\frac{-\theta_{1,1}}{-\theta_{0,1}} - 1\right) \geq 0. \end{aligned}$$

Fisher's information matrix is given by

$$\mathcal{I}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}}^2 \kappa(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\theta_2}{(-\theta_1)^2} & \frac{1}{-\theta_1} \\ \frac{1}{-\theta_1} & \frac{\Gamma''(\theta_2)\Gamma(\theta_2) - \Gamma'(\theta_2)^2}{\Gamma(\theta_2)^2} \end{pmatrix}.$$

The off-diagonal terms in Fisher's information matrix  $\mathcal{I}(\boldsymbol{\theta})$  are non-zero which means that the two components of the canonical parameter  $\boldsymbol{\theta}$  interact. Choosing a different parametrization  $\mu = \theta_2/(-\theta_1)$  (dual mean parametrization) and  $\alpha = \theta_2$  we receive diagonal Fisher's information in  $(\mu, \alpha)$

$$\mathcal{I}(\mu, \alpha) = \begin{pmatrix} \frac{\alpha}{\mu^2} & 0 \\ 0 & \frac{\Gamma''(\alpha)\Gamma(\alpha) - \Gamma'(\alpha)^2}{\Gamma(\alpha)^2} - \frac{1}{\alpha} \end{pmatrix} = \begin{pmatrix} \frac{\alpha}{\mu^2} & 0 \\ 0 & \Psi'(\alpha) - \frac{1}{\alpha} \end{pmatrix}, \quad (2.30)$$

where  $\Psi$  is the digamma function, see Footnote 2 on page 22. This transformation is obtained by using the corresponding Jacobian matrix for variable transformation; more details are provided in (3.16) below. In this new representation, the parameters  $\mu$  and  $\alpha$  are orthogonal; the term  $\Psi'(\alpha) - \frac{1}{\alpha}$  is further discussed in Remarks 5.26 and Remarks 5.28, below.

Using this second parametrization based on mean  $\mu$  and dispersion  $1/\alpha$ , we arrive at the EDF representation of the gamma model. This allows us to calculate the corresponding unit deviance (within the EDF), which in the gamma case is given by

$$\mathfrak{d}(Y, \mu) = 2 \left( \frac{Y}{\mu} - 1 + \log \left( \frac{\mu}{Y} \right) \right) \geq 0.$$

■

*Example 2.26 (Inverse Gaussian Model)* Our final example considers the inverse Gaussian vector-valued parameter EF case. We consider the cumulant function  $\kappa(\boldsymbol{\theta}) = -2(\theta_1\theta_2)^{1/2} - \frac{1}{2}\log(-2\theta_2)$  for  $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top \in \boldsymbol{\Theta} = (-\infty, 0] \times (-\infty, 0)$ , see Sect. 2.1.3. For the KL divergence from model  $\boldsymbol{\theta}_1$  to model  $\boldsymbol{\theta}_0$  we receive

$$\begin{aligned} D_{\text{KL}}(f(\cdot; \boldsymbol{\theta}_0) || f(\cdot; \boldsymbol{\theta}_1)) &= -\theta_{1,1} \sqrt{\frac{-\theta_{0,2}}{-\theta_{0,1}}} - \theta_{1,2} \sqrt{\frac{-\theta_{0,1}}{-\theta_{0,2}}} - 2\sqrt{\theta_{1,1}\theta_{1,2}} \\ &\quad + \frac{\theta_{0,2} - \theta_{1,2}}{-2\theta_{0,2}} + \frac{1}{2} \log \left( \frac{-\theta_{0,2}}{-\theta_{1,2}} \right) \geq 0. \end{aligned}$$

Fisher's information matrix is given by

$$\mathcal{I}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}}^2 \kappa(\boldsymbol{\theta}) = \begin{pmatrix} \frac{(-2\theta_2)^{1/2}}{(-2\theta_1)^{3/2}} & -\frac{1}{2(\theta_1\theta_2)^{1/2}} \\ -\frac{1}{2(\theta_1\theta_2)^{1/2}} & \frac{(-2\theta_1)^{1/2}}{(-2\theta_2)^{3/2}} + \frac{2}{(-2\theta_2)^2} \end{pmatrix}.$$

Again the off-diagonal terms in Fisher's information matrix  $\mathcal{I}(\boldsymbol{\theta})$  are non-zero in the canonical parametrization. We switch to the mean parametrization by setting

$\mu = (-2\theta_2/(-2\theta_1))^{1/2}$  and  $\alpha = -2\theta_2$ . This provides us with diagonal Fisher's information

$$\mathcal{I}(\mu, \alpha) = \begin{pmatrix} \frac{\alpha}{\mu^3} & 0 \\ 0 & \frac{1}{2\alpha^2} \end{pmatrix}. \quad (2.31)$$

This transformation is again obtained by using the corresponding Jacobian matrix for variable transformation, see (3.16), below. We compare the lower-right entries of (2.30) and (2.31). Remark that we have first order approximation of the digamma function

$$\Psi(\alpha) \approx \log \alpha - \frac{1}{2\alpha},$$

and taking derivatives says that these entries of Fisher's information are first order equivalent; this is also used in the saddlepoint approximation in Sect. 5.5.2, below. Using this second parametrization based on mean  $\mu$  and dispersion  $1/\alpha$ , we arrive at the EDF representation of the inverse Gaussian model with unit deviance

$$\mathfrak{d}(Y, \mu) = \frac{(Y - \mu)^2}{\mu^2 Y} \geq 0.$$

■

More examples will be given in Chap. 4, below.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.



# Chapter 3

## Estimation Theory



This chapter gives an introduction to decision and estimation theory. This introduction is based on the books of Lehmann [243, 244], the lecture notes of Künsch [229] and the book of Van der Vaart [363]. This chapter presents classical statistical estimation theory, it embeds estimation into a historical context, and it provides important aspects and intuition for modern data science and predictive modeling. For further reading we recommend the books of Barndorff-Nielsen [23], Berger [31], Bickel–Doksum [33] and Efron–Hastie [117].

### 3.1 Introduction to Decision Theory

We start from an observation vector  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  taking values in a measurable space  $\mathbb{Y} \subset \mathbb{R}^n$ , where  $n \in \mathbb{N}$  denotes the number of components  $Y_i$ ,  $1 \leq i \leq n$ , in  $\mathbf{Y}$ . Assume that this observation vector  $\mathbf{Y}$  has been generated by a distribution belonging to the family  $\mathcal{P} = \{P(\cdot; \theta); \theta \in \Theta\}$  being parametrized by a parameter set  $\Theta$ .

*Remarks 3.1* There are some subtle points in the notation that we are going to use. We use  $P(\cdot; \theta)$  for the distribution of the observation vector  $\mathbf{Y}$ , and if we consider a specific component  $Y_i$  of  $\mathbf{Y}$  we will use the notation  $Y_i \sim F(\cdot; \theta)$ . We make this distinction as in estimation theory one often considers i.i.d. observations  $Y_i \sim F(\cdot; \theta)$ ,  $1 \leq i \leq n$ , with (in this case) joint product distribution  $\mathbf{Y} \sim P(\cdot; \theta)$ . This latter distribution is then used for purposes of maximum likelihood estimation, etc. The family  $\mathcal{P}$  is parametrized by  $\theta \in \Theta$ , and if we want to emphasize that this parameter is a  $k$ -dimensional vector we use boldface notation  $\boldsymbol{\theta}$ , this is similar to the EFs introduced in Chap. 2, but in this chapter we do not restrict to EFs. Finally, we assume identifiability meaning that different parameters  $\theta$  give different distributions  $P(\cdot; \theta) \in \mathcal{P}$ .

To fix ideas, assume we want to determine  $\gamma(\theta)$  of a given functional  $\gamma(\cdot)$  on  $\Theta$ . Typically, the true value  $\theta \in \Theta$  is not known, and we are not able to determine  $\gamma(\theta)$  explicitly. Therefore, we try to *estimate*  $\gamma(\theta)$  from data  $Y \sim P(\cdot; \theta)$  that belongs to the same  $\theta \in \Theta$ . As an example we may think of working in the EDF of Chap. 2, and we are interested in the mean  $\mu = \mathbb{E}_\theta[Y] = \kappa'(\theta)$  of  $Y$ . Thus, we aim at determining  $\gamma(\theta) = \kappa'(\theta)$ . If the true  $\theta$  is unknown, and if we have an observation  $Y$  from this model, we can try to estimate  $\gamma(\theta) = \kappa'(\theta)$  from  $Y$ . This motivation is based on estimation of  $\gamma(\theta)$ , but the following framework of decision making is more general, for instance, it may also be used for statistical hypothesis testing.

Denote the *action space* of possible decisions (actions) by  $\mathbb{A}$ . In decision theory we are looking for a *decision rule* (*action rule*)

$$A : \mathbb{Y} \rightarrow \mathbb{A}, \quad Y \mapsto A(Y), \quad (3.1)$$

which should be understood as an educated guess for  $\gamma(\theta)$  based on observation  $Y$ . A decision rule is evaluated in terms of a (given) *loss function*

$$L : \Theta \times \mathbb{A} \rightarrow \mathbb{R}_+, \quad (\theta, a) \mapsto L(\theta, a) \geq 0. \quad (3.2)$$

$L(\theta, a)$  describes the loss of an action  $a \in \mathbb{A}$  w.r.t. a true parameter choice  $\theta \in \Theta$ . The *risk function* of decision rule  $A$  for data generated by  $Y \sim P(\cdot; \theta)$  is defined by

$$\theta \mapsto \mathcal{R}(\theta, A) = \mathbb{E}_\theta[L(\theta, A(Y))] = \int_{\mathbb{Y}} L(\theta, A(y)) dP(y; \theta), \quad (3.3)$$

where  $\mathbb{E}_\theta$  is the expectation w.r.t. the probability distribution  $P(\cdot; \theta)$ . Risk function (3.3) describes the long-term average loss of using decision rule  $A$ . As an example we may think of estimating  $\gamma(\theta)$  for unknown (true) parameter  $\theta$  by a decision rule  $Y \mapsto A(Y)$ . Then, the loss function  $L(\theta, A(Y))$  should describe the *estimation loss* if we consider the discrepancy between  $\gamma(\theta)$  and its estimate  $A(Y)$ , and the risk function  $\mathcal{R}(\theta, A)$  is the *average estimation loss* in that case.

Good decision rules  $A$  should provide a small risk  $\mathcal{R}(\theta, A)$ . Unfortunately, this statement is of rather theoretical nature because, in general, the true data generating parameter  $\theta$  is not known and the goodness of a decision rule for the true parameter cannot be evaluated explicitly, but the risk can only be estimated (for instance, using a bootstrap approach). Moreover, typically, there does not exist a uniformly best decision rule  $A$  over all  $\theta \in \Theta$ . For these reasons we may (just) try to eliminate decision rules that are obviously not good. We give two introductory examples.

**Example 3.2 (Minimax Decision Rule)** Decision rule  $A$  is called minimax if for all alternative decision rules  $\tilde{A} : \mathbb{Y} \rightarrow \mathbb{A}$  we have

$$\sup_{\theta \in \Theta} \mathcal{R}(\theta, A) \leq \sup_{\theta \in \Theta} \mathcal{R}(\theta, \tilde{A}).$$

A minimax decision rule is the best choice in the worst case of the true  $\theta$ , i.e., it minimizes the worst case risk. ■

*Example 3.3 (Bayesian Decision Rule)* Assume we are given a distribution  $\pi$  on  $\Theta$ . Decision rule  $A$  is called Bayesian w.r.t.  $\pi$  if it satisfies

$$A = \arg \min_{\tilde{A}} \int_{\Theta} \mathcal{R}(\theta, \tilde{A}) d\pi(\theta).$$

Distribution  $\pi$  is called *prior distribution* on  $\Theta$ . ■

The above examples give two possible choices of decision rules. The first one tries to minimize the worst case risk, whereas the second one uses additional knowledge in terms of a prior distribution  $\pi$  on  $\Theta$ . This means that we impose stronger assumptions in the second case to get stronger conclusions. The difficult part in practice is to justify these stronger assumptions in order to validate the stronger conclusions. Below, we are going to introduce other criteria that should be satisfied by good decision rules, an important one in estimation will be unbiasedness.

## 3.2 Parameter Estimation

This section focuses on estimating the (unknown) parameter  $\theta \in \Theta$  from observation  $Y \sim P(\cdot; \theta)$ . For this we consider decision rules  $A : \mathbb{Y} \rightarrow \mathbb{A} = \Theta$  with  $A(Y)$  estimating  $\theta$ . We assume there exist densities  $p(\cdot; \theta)$  w.r.t. a fixed  $\sigma$ -finite measure  $\nu$  on  $\mathbb{Y} \subset \mathbb{R}^n$ ,

$$dP(y; \theta) = p(y; \theta) d\nu(y),$$

for all distributions  $P(\cdot; \theta) \in \mathcal{P}$ , i.e., all  $\theta \in \Theta$ .

**Definition 3.4 (Maximum Likelihood Estimator, MLE)** The maximum likelihood estimator (MLE) of  $\theta$  for a given observation  $Y \in \mathbb{Y}$  is given by (subject to existence and uniqueness)

$$\hat{\theta}^{\text{MLE}} = \arg \max_{\tilde{\theta} \in \Theta} p(Y; \tilde{\theta}) = \arg \max_{\tilde{\theta} \in \Theta} \ell_Y(\tilde{\theta}),$$

where the log-likelihood function of  $p(Y; \theta)$  is defined by  $\theta \mapsto \ell_Y(\theta) = \log p(Y; \theta)$ .

The MLE  $Y \mapsto \hat{\theta}^{\text{MLE}} = \hat{\theta}^{\text{MLE}}(Y) = A(Y)$  is nothing else than a specific decision rule with action space  $\mathbb{A} = \Theta$  for estimating  $\theta$ . We can now start to explore the risk function  $\mathcal{R}(\theta, \hat{\theta}^{\text{MLE}})$  of that decision rule for a given loss function  $L$ .

*Example 3.5 (MLE within the EDF)* We emphasize that this example is used throughout these notes. Assume that the (independent) components of  $Y = (Y_1, \dots, Y_n)^\top \sim P(\cdot; \theta)$  follow a given EDF distribution. That is, we assume that  $Y_1, \dots, Y_n$  are independent and have densities w.r.t.  $\sigma$ -finite measures on  $\mathbb{R}$  given by, see (2.14),

$$Y_i \sim f(y_i; \theta, v_i/\varphi) = \exp \left\{ \frac{y_i \theta - \kappa(\theta)}{\varphi/v_i} + a(y_i; v_i/\varphi) \right\},$$

for  $1 \leq i \leq n$ . Note that these random variables are not i.i.d. because they may differ in exposures  $v_i > 0$ . Throughout, we assume that Assumption 2.6 is fulfilled and that the cumulant function  $\kappa$  is steep, see Theorem 2.19. For the latter we also refer to Remark 2.20: the supports  $\mathfrak{T}_{v_i/\varphi}$  of  $Y_i$  may differ; however, these supports share the same convex closure.

Independence between the  $Y_i$ 's implies that the joint probability  $P(\cdot; \theta)$  is the product distribution of the individual distributions  $F(\cdot; \theta, v_i/\varphi)$ ,  $1 \leq i \leq n$ . Therefore, the MLE of  $\theta$  in the EDF is found by solving

$$\hat{\theta}^{\text{MLE}} = \arg \max_{\tilde{\theta} \in \Theta} \ell_Y(\tilde{\theta}) = \arg \max_{\tilde{\theta} \in \Theta} \sum_{i=1}^n \frac{Y_i \tilde{\theta} - \kappa(\tilde{\theta})}{\varphi/v_i}.$$

Since the cumulant function  $\kappa$  is strictly convex we receive the MLE (subject to existence)

$$\hat{\theta}^{\text{MLE}} = \hat{\theta}^{\text{MLE}}(Y) = (\kappa')^{-1} \left( \frac{\sum_{i=1}^n v_i Y_i}{\sum_{i=1}^n v_i} \right) = h \left( \frac{\sum_{i=1}^n v_i Y_i}{\sum_{i=1}^n v_i} \right).$$

Thus, the MLE is received by applying the canonical link  $h = (\kappa')^{-1}$ , see Definition 2.8, and strict convexity of  $\kappa$  implies that the MLE is unique. However, existence needs to be analyzed more carefully! It may happen that the MLE  $\hat{\theta}^{\text{MLE}}$  is a boundary point of the effective domain  $\Theta$  which may not exist (if  $\Theta$  is open). We give an example. Assume we work in the Poisson model presented in Sect. 2.1.2. The canonical link in the Poisson model is the log-link  $\mu \mapsto h(\mu) = \log(\mu)$ , for  $\mu > 0$ . With positive probability we have in the Poisson case  $\sum_{i=1}^n v_i Y_i = 0$ .

Therefore, with positive probability the MLE  $\hat{\theta}^{\text{MLE}}$  does not exist (we have a degenerate Poisson model in that case).

Since the canonical link is strictly increasing we can also perform MLE in the dual (mean) parametrization. The dual parameter space is given by  $\mathcal{M} = \kappa'(\Theta)$ , see Remarks 2.9, with mean parameters  $\mu = \kappa'(\theta) \in \mathcal{M}$ . This motivates

$$\hat{\mu}^{\text{MLE}} = \arg \max_{\tilde{\mu} \in \mathcal{M}} \ell_Y(h(\tilde{\mu})) = \arg \max_{\tilde{\mu} \in \mathcal{M}} \sum_{i=1}^n \frac{Y_i h(\tilde{\mu}) - \kappa(h(\tilde{\mu}))}{\varphi/v_i}. \quad (3.4)$$

Subject to existence, this provides the unique MLE

$$\hat{\mu}^{\text{MLE}} = \hat{\mu}^{\text{MLE}}(Y) = \frac{\sum_{i=1}^n v_i Y_i}{\sum_{i=1}^n v_i}. \quad (3.5)$$

Also this dual MLE does not need to exist (in the dual parameter space  $\mathcal{M}$ ). Under the assumption that the cumulant function  $\kappa$  is steep, we know that the closure of the dual parameter space  $\overline{\mathcal{M}}$  contains the supports  $\mathfrak{T}_{v_i/\varphi}$  of  $Y_i$ , see Theorem 2.19 and Remark 2.20. Thus, in that case we can close the dual parameter space and receive MLE  $\hat{\mu}^{\text{MLE}} \in \overline{\mathcal{M}}$  (in a possibly degenerate model). In the aforementioned degenerate Poisson situation we receive  $\hat{\mu}^{\text{MLE}} = 0$  which is in the boundary  $\partial\mathcal{M}$  of the dual parameter space. ■

**Definition 3.6 (Bayesian Estimator)** The Bayesian estimator of  $\theta$  for a given observation  $Y \in \mathbb{Y}$  and a given prior distribution  $\pi$  on  $\Theta$  is given by (subject to existence)

$$\hat{\theta}^{\text{Bayes}} = \hat{\theta}^{\text{Bayes}}(Y) = \mathbb{E}_{\pi}[\theta|Y],$$

where the conditional expectation on the right-hand side is calculated under the posterior distribution  $\pi(\theta|y) \propto p(y; \theta)\pi(\theta)$  for a given observation  $Y = y$ .

*Example 3.7 (Bayesian Estimator)* Assume that  $\mathbb{A} = \Theta = \mathbb{R}$  and choose the square loss function  $L(\theta, a) = (\theta - a)^2$ . Assume that for  $\nu$ -a.e.  $y \in \mathbb{Y}$  the following decision rule  $A : \mathbb{Y} \rightarrow \mathbb{A}$  exists

$$A(y) = \arg \min_{a \in \mathbb{A}} \mathbb{E}_{\pi}[(\theta - a)^2|Y = y], \quad (3.6)$$



where the expectation is calculated w.r.t. the posterior distribution  $\pi(\theta|\mathbf{y})$ . In this case,  $A$  is a Bayesian decision rule w.r.t.  $\pi$  and  $L(\theta, a) = (\theta - a)^2$ : by assumption (3.6) we have for any other decision rule  $\tilde{A} : \mathbb{Y} \rightarrow \mathbb{A}$ , v-a.s.,

$$\mathbb{E}_\pi[(\theta - A(\mathbf{Y}))^2 | \mathbf{Y} = \mathbf{y}] \leq \mathbb{E}_\pi[(\theta - \tilde{A}(\mathbf{Y}))^2 | \mathbf{Y} = \mathbf{y}].$$

Applying the tower property we receive for any other decision rule  $\tilde{A}$

$$\int_{\Theta} \mathcal{R}(\theta, A) d\pi(\theta) = \mathbb{E}[(\theta - A(\mathbf{Y}))^2] \leq \mathbb{E}[(\theta - \tilde{A}(\mathbf{Y}))^2] = \int_{\Theta} \mathcal{R}(\theta, \tilde{A}) d\pi(\theta),$$

where the expectation  $\mathbb{E}$  is calculated over the joint distribution of  $\mathbf{Y}$  and  $\theta$ . This proves that  $A$  is a Bayesian decision rule w.r.t.  $\pi$  and  $L(\theta, a) = (\theta - a)^2$ , see Example 3.3. Finally, note that the conditional expectation given in Definition 3.6 is the minimizer of (3.6). This justifies the name Bayesian estimator in Definition 3.6 (for the square loss function). The case of the Bayesian estimator for a general loss function  $L$  is considered in Theorem 4.1.1 of Lehmann [244]. ■

**Definition 3.8 (Method of Moments Estimator)** Assume that  $\Theta \subseteq \mathbb{R}^k$  and that the components  $Y_i$  of  $\mathbf{Y}$  are i.i.d.  $F(\cdot; \theta)$  distributed with finite  $k$ -th moments for all  $\theta \in \Theta$ . The law of large numbers provides, a.s., for all  $1 \leq l \leq k$ ,

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n Y_i^l = \mathbb{E}_\theta[Y_1^l].$$

Assume that the following map is invertible (on suitable range definitions for (3.7)–(3.8))

$$\gamma : \Theta \rightarrow \mathbb{R}^k, \quad \theta \mapsto \gamma(\theta) = (\mathbb{E}_\theta[Y_1], \dots, \mathbb{E}_\theta[Y_1^k])^\top. \quad (3.7)$$

The method of moments estimator of  $\theta$  is defined by

$$\hat{\theta}^{\text{MM}} = \hat{\theta}^{\text{MM}}(\mathbf{Y}) = \gamma^{-1} \left( \frac{1}{n} \sum_{i=1}^n Y_i, \dots, \frac{1}{n} \sum_{i=1}^n Y_i^k \right)^\top. \quad (3.8)$$

The MLE, the Bayesian estimator and the method of moments estimator are the most commonly used parameter estimators. They may have additional properties (under certain assumptions) that we are going to explore below. In the remainder of this section we give an additional view on estimators which is based on the empirical distribution of the observation  $\mathbf{Y}$ .

Assume that the components  $Y_i$  of  $\mathbf{Y}$  are real-valued and i.i.d.  $F$  distributed. The empirical distribution induced by the observation  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  is given by

$$\widehat{F}_n(y) = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{\{Y_i \leq y\}} \quad \text{for } y \in \mathbb{R}, \quad (3.9)$$

we also refer to Fig. 1.2 (lhs). The Glivenko–Cantelli theorem [64, 159] tells us that the empirical distribution  $\widehat{F}_n$  converges uniformly to  $F$ , a.s., for  $n \rightarrow \infty$ .

**Definition 3.9 (Fisher-Consistency)** Denote by  $\mathfrak{P}$  the set of all distribution functions on the given probability space. Let  $Q : \mathfrak{P} \rightarrow \Theta$  be a functional with the property

$$Q(F(\cdot; \theta)) = \theta \quad \text{for all } F(\cdot; \theta) \in \mathcal{F} = \{F(\cdot; \theta); \theta \in \Theta\} \subset \mathfrak{P}.$$

Such a functional is called *Fisher-consistent* for  $\mathcal{F}$  and  $\theta \in \Theta$ , respectively.

A given Fisher-consistent functional  $Q$  motivates the estimator  $\widehat{\theta} = Q(\widehat{F}_n) \in \Theta$ . This is exactly what we have applied for the method of moments estimator (3.8) with Fisher-consistent functional induced by the inverse of (3.7). The next example shows that this also works for MLE.

*Example 3.10 (MLE and Kullback–Leibler (KL) Divergence)* The MLE can be received from a Fisher-consistent functional. Consider for  $F \in \mathfrak{P}$  the functional

$$Q(F) = \arg \max_{\tilde{\theta}} \int \log f(y; \tilde{\theta}) dF(y),$$

assuming that  $f(\cdot; \tilde{\theta})$  are densities w.r.t. a  $\sigma$ -finite measure on  $\mathbb{R}$ . Assume that  $F$  has density  $f$  w.r.t. the  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$ . Then, we can rewrite the above as

$$Q(F) = \arg \min_{\tilde{\theta}} \int \log \left( \frac{f(y)}{f(y; \tilde{\theta})} \right) f(y) d\nu(y) = \arg \min_{\tilde{\theta}} D_{\text{KL}}(f \| f(\cdot; \tilde{\theta})).$$

The latter is the Kullback–Leibler (KL) divergence which we have met in Sect. 2.3. Lemma 2.21 states that the KL divergence is non-negative, and it is zero if and only if the two densities  $f$  and  $f(\cdot; \tilde{\theta})$  are identical,  $\nu$ -a.s. This implies that  $Q(F(\cdot; \theta)) = \theta$ . Thus,  $Q$  is Fisher-consistent for  $\theta \in \Theta$ , assuming identifiability, see Remarks 3.1.

Next, we use this Fisher-consistent functional (KL divergence) to receive the MLE. Replace the unknown distribution  $F$  by the empirical one to receive

$$\begin{aligned} Q(\widehat{F}_n) &= \arg \min_{\tilde{\theta}} D_{\text{KL}}(\widehat{f}_n \| f(\cdot; \tilde{\theta})) \\ &= \arg \max_{\tilde{\theta}} \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \tilde{\theta}) = \widehat{\theta}^{\text{MLE}}, \end{aligned}$$

where we have used that the empirical density  $\hat{f}_n$  allocates point masses of size  $1/n$  to the i.i.d. observations  $Y_1, \dots, Y_n$ . Thus, the MLE  $\hat{\theta}^{\text{MLE}}$  of  $\theta$  can be obtained by choosing the model  $f(\cdot; \tilde{\theta})$ ,  $\tilde{\theta} \in \Theta$ , that is closest in KL divergence to the empirical distribution  $\hat{F}_n$  of i.i.d. observations  $Y_i \sim F$ . Note that in this construction we do not assume that the true distribution  $F$  is in  $\mathcal{F}$ , see Definition 3.9. ■

### Remarks 3.11

- Many properties of estimators of  $\theta$  are based on properties of Fisher-consistent functionals  $Q$  (in cases where they exist). For instance, asymptotic properties as  $n \rightarrow \infty$  are obtained from smoothness properties of Fisher-consistent functionals  $Q$ , or using the influence function we can analyze the impact of individual observations  $Y_i$  on decision rules  $\hat{\theta} = \hat{\theta}(Y) = Q(\hat{F}_n)$ . The latter is the basis of robust statistics, see Huber [194] and Hampel et al. [180]. Since Fisher-consistent functionals do not require that the true distribution belongs to  $\mathcal{F}$  it requires a careful consideration of the quantity to be estimated.
- The discussion on parameter estimation has implicitly assumed that the true data generating model belongs to the family  $\mathcal{P} = \{P(\cdot; \theta); \theta \in \Theta\}$ , and the only problem was to find the true parameter in  $\Theta$ . More generally, one should also consider model uncertainty w.r.t. the chosen family  $\mathcal{P}$ , i.e., the data generating model may not belong to this family. Of course, this problem is by far more difficult. We explore this in more detail in Sect. 11.1.4, below.

## 3.3 Unbiased Estimators

We introduce the property of uniformly minimum variance unbiased (UMVU) for decision rules in this section. This is a very attractive property in insurance pricing because it gives a quality statement to decision rules (and to the resulting prices). At the current stage it is not clear how unbiasedness is related, e.g., to the MLE of  $\theta$ .

### 3.3.1 Cramér–Rao Information Bound

Above we have stated some quality criteria for decision rules like the minimax property. A crucial property in financial applications is the so-called *unbiasedness* (for mean estimates) because this guarantees that the overall (price) levels are correctly specified.

**Definition 3.12 (Uniformly Minimum Variance Unbiased, UMVU)** A decision rule  $A : \mathbb{Y} \rightarrow \mathbb{A} = \mathbb{R}$  is unbiased for  $\gamma : \Theta \rightarrow \mathbb{R}$  if for all  $Y \sim P(\cdot; \theta)$ ,  $\theta \in \Theta$ , we have

$$\mathbb{E}_\theta[A(Y)] = \gamma(\theta). \quad (3.10)$$

The decision rule  $A$  is called UMVU for  $\gamma$  if additionally to the unbiasedness (3.10) we have

$$\text{Var}_\theta(A(Y)) \leq \text{Var}_\theta(\tilde{A}(Y)),$$

for all  $\theta \in \Theta$  and for any other decision rule  $\tilde{A} : \mathbb{Y} \rightarrow \mathbb{R}$  that is unbiased for  $\gamma$ .

Note that unbiasedness is not invariant under transformations, i.e., if  $A(Y)$  is unbiased for  $\gamma(\theta)$ , then, in general,  $b(A(Y))$  is not unbiased for  $b(\gamma(\theta))$ . For instance, if  $b$  is strictly convex then we get a counterexample by simply applying Jensen's inequality.

Our first step is to derive a general lower bound for  $\text{Var}_\theta(A(Y))$ . If this general lower bound is met for an unbiased decision rule  $A$  for  $\gamma$ , then we know that it is UMVU for  $\gamma$ . We start with the one-dimensional case given in Section 2.6 of Lehmann [244].

**Theorem 3.13 (Cramér–Rao Information Bound)** Assume that the distributions  $P(\cdot; \theta)$ ,  $\theta \in \Theta$ , have densities  $p(\cdot; \theta)$  for a given  $\sigma$ -finite measure  $\nu$  on  $\mathbb{Y}$ , and that  $\Theta \subset \mathbb{R}$  is an open interval such that the set  $\{\mathbf{y}; p(\mathbf{y}; \theta) > 0\}$  does not depend on  $\theta \in \Theta$ . Let  $A(Y)$  be unbiased for  $\gamma : \Theta \rightarrow \mathbb{R}$  having finite second moment. If the limit

$$\frac{\partial}{\partial \theta} \log p(\mathbf{y}; \theta) = \lim_{\Delta \rightarrow 0} \frac{1}{\Delta} \frac{p(\mathbf{y}; \theta + \Delta) - p(\mathbf{y}; \theta)}{p(\mathbf{y}; \theta)}$$

exists in  $L^2(P(\cdot; \theta))$  and if

$$\mathcal{I}(\theta) = \mathbb{E}_\theta \left[ \left( \frac{\partial}{\partial \theta} \log p(Y; \theta) \right)^2 \right] \in (0, \infty),$$

then the function  $\theta \mapsto \gamma(\theta)$  is differentiable,  $\mathbb{E}_\theta \left[ \frac{\partial}{\partial \theta} \log p(Y; \theta) \right] = 0$  and we have information bound

$$\text{Var}_\theta(A(Y)) \geq \frac{\gamma'(\theta)^2}{\mathcal{I}(\theta)}.$$

**Proof** We start from an arbitrary function  $\psi : \Theta \times \mathbb{Y} \rightarrow \mathbb{R}$  with finite variance  $\text{Var}_\theta(\psi(\theta, \mathbf{Y})) \in (0, \infty)$  for all  $\theta \in \Theta$ . The Cauchy–Schwarz inequality implies

$$\text{Var}_\theta(A(\mathbf{Y})) \geq \frac{\text{Cov}_\theta(A(\mathbf{Y}), \psi(\theta, \mathbf{Y}))^2}{\text{Var}_\theta(\psi(\theta, \mathbf{Y}))}. \quad (3.11)$$

If we manage to make the right-hand side of (3.11) independent of decision rule  $A(\cdot)$  we have a general lower bound, we also refer to Theorem 2.6.1 in Lehmann [244].

The Cauchy–Schwarz inequality implies that for any  $U \in L^2(P(\cdot; \theta))$  the following limit exists and is equal to

$$\lim_{\Delta \rightarrow 0} \mathbb{E}_\theta \left[ \frac{1}{\Delta} \frac{p(\mathbf{Y}; \theta + \Delta) - p(\mathbf{Y}; \theta)}{p(\mathbf{Y}; \theta)} U \right] = \mathbb{E}_\theta \left[ \frac{\partial}{\partial \theta} \log p(\mathbf{Y}; \theta) U \right]. \quad (3.12)$$

Setting  $U \equiv 1$  gives average score  $\mathbb{E}_\theta[\frac{\partial}{\partial \theta} \log p(\mathbf{Y}; \theta)] = 0$  because for sufficiently small  $\Delta$

$$\mathbb{E}_\theta \left[ \frac{p(\mathbf{Y}; \theta + \Delta) - p(\mathbf{Y}; \theta)}{p(\mathbf{Y}; \theta)} \right] = \int_{\mathbb{Y}} \frac{p(\mathbf{y}; \theta + \Delta) - p(\mathbf{y}; \theta)}{p(\mathbf{y}; \theta)} p(\mathbf{y}; \theta) d\nu(\mathbf{y}) = 0,$$

where we have used that the support of the random variables does not depend on  $\theta$  and that the domain  $\Theta$  of  $\theta$  is open.

Secondly, we set  $U = A(\mathbf{Y})$  in (3.12). We have similarly to above using unbiasedness w.r.t.  $\gamma$

$$\begin{aligned} \text{Cov}_\theta \left( A(\mathbf{Y}), \frac{p(\mathbf{Y}; \theta + \Delta) - p(\mathbf{Y}; \theta)}{p(\mathbf{Y}; \theta)} \right) &= \int_{\mathbb{Y}} A(\mathbf{y}) \frac{p(\mathbf{y}; \theta + \Delta) - p(\mathbf{y}; \theta)}{p(\mathbf{y}; \theta)} p(\mathbf{y}; \theta) d\nu(\mathbf{y}) \\ &= \gamma(\theta + \Delta) - \gamma(\theta). \end{aligned}$$

Existence of limit (3.12) provides the differentiability of  $\gamma$ . Finally, from (3.11) we have

$$\text{Var}_\theta(A(\mathbf{Y})) \geq \lim_{\Delta \rightarrow 0} \frac{\text{Cov}_\theta \left( A(\mathbf{Y}), \frac{p(\mathbf{Y}; \theta + \Delta) - p(\mathbf{Y}; \theta)}{p(\mathbf{Y}; \theta)} \right)^2}{\text{Var}_\theta \left( \frac{p(\mathbf{Y}; \theta + \Delta) - p(\mathbf{Y}; \theta)}{p(\mathbf{Y}; \theta)} \right)} = \frac{\gamma'(\theta)^2}{\mathcal{I}(\theta)}. \quad (3.13)$$

This completes the proof.  $\square$

*Remarks 3.14 (Fisher's Information and Score)*

- $\mathcal{I}(\theta)$  is called *Fisher's information* or *Fisher metric*.
- $s(\theta, \mathbf{Y}) = \frac{\partial}{\partial \theta} \log p(\mathbf{Y}; \theta)$  is called *score*, and  $\mathbb{E}_\theta[s(\mathbf{Y}; \theta)] = 0$  in Theorem 3.13 expresses that the average score is zero under the assumptions of that theorem.
- Under the regularity conditions of Lemma 6.1 in Section 2.6 of Lehmann [244]

$$\mathcal{I}(\theta) = \mathbb{E}_\theta \left[ \left( \frac{\partial}{\partial \theta} \log p(\mathbf{Y}; \theta) \right)^2 \right] = -\mathbb{E}_\theta \left[ \frac{\partial^2}{\partial \theta^2} \log p(\mathbf{Y}; \theta) \right]. \quad (3.14)$$

Fisher's information  $\mathcal{I}(\theta)$  expresses the variance of the score  $s(\theta, \mathbf{Y})$ . Identity (3.14) justifies the notion Fisher's information in Sect. 2.3 for the EF.

- In order to determine the Cramér–Rao information bound for unknown  $\theta$  we need to estimate Fisher's information  $\mathcal{I}(\theta)$  from the available data. There are two different ways to do so, either we choose

$$\mathcal{I}(\hat{\theta}) = \mathbb{E}_{\hat{\theta}} \left[ \left( \frac{\partial}{\partial \theta} \log p(\mathbf{Y}; \theta) \right)^2 \right],$$

or we choose the *observed Fisher's information*

$$\hat{\mathcal{I}}(\hat{\theta}) = \left( \frac{\partial}{\partial \theta} \log p(\mathbf{Y}; \theta) \right)^2 \Big|_{\theta=\hat{\theta}},$$

for given data  $\mathbf{Y}$  and where  $\hat{\theta} = \hat{\theta}(\mathbf{Y})$ . Both estimated Fisher's information  $\mathcal{I}(\hat{\theta})$  and  $\hat{\mathcal{I}}(\hat{\theta})$  play a central role in MLE of generalized linear models (GLMs). They are used in Fisher's scoring method, the iterated re-weighted least squares (IRLS) algorithm and the Newton–Raphson algorithm to determine the MLE.

- The Cramér–Rao information bound in Theorem 3.13 is stated in terms of the observation  $\mathbf{Y} \sim p(\cdot; \theta)$ . Assume that the components  $Y_i$  of  $\mathbf{Y}$  are i.i.d.  $f(\cdot; \theta)$  distributed. In this case, Fisher's information scales as

$$\mathcal{I}(\theta) = \mathcal{I}_n(\theta) = n\mathcal{I}_1(\theta), \quad (3.15)$$

with single risk's Fisher's information (contribution)

$$\mathcal{I}_1(\theta) = \mathbb{E}_\theta \left[ \left( \frac{\partial}{\partial \theta} \log f(Y_1; \theta) \right)^2 \right].$$

In general, Fisher's information is additive in independent random variables, because the product of densities is additive after applying the logarithm, and because the average score is zero.

**Proposition 3.15** *The unbiased decision rule  $A$  for  $\gamma$  attains the Cramér–Rao information bound if and only if the density is of the form  $p(\mathbf{y}; \theta) = \exp\{\delta(\theta)T(\mathbf{y}) - \beta(\theta) + a(\mathbf{y})\}$  with  $T = A$ . In that case we have  $\gamma(\theta) = \beta'(\theta)/\delta'(\theta)$ .*

**Proof of Proposition 3.15** The Cauchy–Schwarz inequality provides equality in (3.13) if and only if  $\frac{\partial}{\partial \theta} \log p(\mathbf{y}; \theta) = \delta'(\theta)A(\mathbf{y}) - \beta'(\theta)$ ,  $v$ -a.s., for some functions  $\delta'(\theta)$  and  $\beta'(\theta)$  on  $\Theta$ . Integration and the fact that  $p(\cdot; \theta)$  is a density whose support does not depend on the explicit choice of  $\theta \in \Theta$  provide the implication “ $\Rightarrow$ ”. For the implication “ $\Leftarrow$ ” we study for  $A = T$

$$0 = \mathbb{E}_\theta \left[ \frac{\partial}{\partial \theta} \log p(\mathbf{Y}; \theta) \right] = \int_{\mathbb{Y}} (\delta'(\theta)A(\mathbf{y}) - \beta'(\theta)) p(\mathbf{y}; \theta) d\nu(\mathbf{y}) = \delta'(\theta) \mathbb{E}_\theta[A(\mathbf{Y})] - \beta'(\theta).$$

In that case we have  $\gamma(\theta) = \mathbb{E}_\theta[A(\mathbf{Y})] = \beta'(\theta)/\delta'(\theta)$ . Moreover, we have equality in the Cauchy–Schwarz inequality. This finishes the proof.  $\square$

The single-parameter EF fulfills the properties of Proposition 3.15 with  $\delta(\theta) = \theta$  and  $\beta(\theta) = \kappa(\theta)$ , and decision rule  $A(\mathbf{y}) = T(\mathbf{y})$  attains the Cramér–Rao information bound for  $\gamma(\theta) = \kappa'(\theta)$ .

We give a multi-dimensional version of the Cramér–Rao information bound.

**Theorem 3.16 (Multi-Dimensional Version of the Cramér–Rao Information Bound, Without Proof)** *Assume that the distributions  $P(\cdot; \theta)$ ,  $\theta \in \Theta$ , have densities  $p(\cdot; \theta)$  for a given  $\sigma$ -finite measure  $\nu$  on  $\mathbb{Y}$ , and that  $\Theta \subseteq \mathbb{R}^k$  is an open convex set such that the set  $\{\mathbf{y}; p(\mathbf{y}; \theta) > 0\}$  does not depend on  $\theta \in \Theta$ . Let  $A(\mathbf{Y})$  be unbiased for  $\gamma : \Theta \rightarrow \mathbb{R}$  having finite second moment. Under additional regularity conditions, see Theorem 7.3 in Section 2.7 of Lehmann [244], we have*

$$\text{Var}_\theta(A(\mathbf{Y})) \geq (\nabla_\theta \gamma(\theta))^\top \mathcal{I}(\theta)^{-1} \nabla_\theta \gamma(\theta),$$

with (positive definite) Fisher’s information matrix  $\mathcal{I}(\theta) = (\mathcal{I}_{l,j}(\theta))_{1 \leq l, j \leq k}$  given by

$$\mathcal{I}_{l,j}(\theta) = \mathbb{E}_\theta \left[ \frac{\partial}{\partial \theta_l} \log p(\mathbf{Y}; \theta) \frac{\partial}{\partial \theta_j} \log p(\mathbf{Y}; \theta) \right],$$

for  $1 \leq l, j \leq k$ .

*Remarks 3.17*

- Whenever an unbiased decision rule  $A(\mathbf{Y})$  for  $\gamma(\boldsymbol{\theta})$  meets the Cramér–Rao information bound it is UMVU. Thus, it minimizes the risk function  $\mathcal{R}(\boldsymbol{\theta}, A)$  being based on the square loss  $L(\boldsymbol{\theta}, a) = (\gamma(\boldsymbol{\theta}) - a)^2$  among all unbiased decision rules, because unbiasedness for  $\gamma(\boldsymbol{\theta})$  gives  $\mathcal{R}(\boldsymbol{\theta}, A) = \text{Var}_{\boldsymbol{\theta}}(A(\mathbf{Y}))$ .
- The regularity conditions in Theorem 3.16 include that Fisher’s information matrix  $\mathcal{I}(\boldsymbol{\theta})$  is positive definite.
- Under additional regularity conditions we have the following identity for Fisher’s information matrix

$$\mathcal{I}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\theta}} \left[ (\nabla_{\boldsymbol{\theta}} \log p(\mathbf{Y}; \boldsymbol{\theta})) (\nabla_{\boldsymbol{\theta}} \log p(\mathbf{Y}; \boldsymbol{\theta}))^{\top} \right] = -\mathbb{E}_{\boldsymbol{\theta}} \left[ \nabla_{\boldsymbol{\theta}}^2 \log p(\mathbf{Y}; \boldsymbol{\theta}) \right] \in \mathbb{R}^{k \times k}.$$

Thus, Fisher’s information matrix can either be calculated from a quadratic form of the score  $s(\boldsymbol{\theta}, \mathbf{Y}) = \nabla_{\boldsymbol{\theta}} \log p(\mathbf{Y}; \boldsymbol{\theta})$  or from the Hessian  $\nabla_{\boldsymbol{\theta}}^2$  of the log-likelihood  $\ell_{\mathbf{Y}}(\boldsymbol{\theta}) = \log p(\mathbf{Y}; \boldsymbol{\theta})$ . Since the score has mean zero, Fisher’s information matrix is equal to the covariance matrix of the score  $s(\boldsymbol{\theta}, \mathbf{Y})$ .

In many situations we do not work under the canonical parametrization  $\boldsymbol{\theta}$ . Considerations then require a change of variable. Assume that

$$\boldsymbol{\zeta} \in \mathbb{R}^r \mapsto \boldsymbol{\theta} = \boldsymbol{\theta}(\boldsymbol{\zeta}) \in \mathbb{R}^k,$$

such that all derivatives  $\partial \theta_l(\boldsymbol{\zeta}) / \partial \zeta_j$  exist for  $1 \leq l \leq k$  and  $1 \leq j \leq r$ . The Jacobian matrix is given by

$$J(\boldsymbol{\zeta}) = \left( \frac{\partial}{\partial \zeta_j} \theta_l(\boldsymbol{\zeta}) \right)_{1 \leq l \leq k, 1 \leq j \leq r} \in \mathbb{R}^{k \times r}.$$

Fisher’s information matrix w.r.t.  $\boldsymbol{\zeta}$  is given by

$$\mathcal{I}^*(\boldsymbol{\zeta}) = \left( \mathbb{E}_{\boldsymbol{\theta}(\boldsymbol{\zeta})} \left[ \frac{\partial}{\partial \zeta_l} \log p(\mathbf{Y}; \boldsymbol{\theta}(\boldsymbol{\zeta})) \frac{\partial}{\partial \zeta_j} \log p(\mathbf{Y}; \boldsymbol{\theta}(\boldsymbol{\zeta})) \right] \right)_{1 \leq l, j \leq r} \in \mathbb{R}^{r \times r},$$

and we have the identity

$$\mathcal{I}^*(\boldsymbol{\zeta}) = J(\boldsymbol{\zeta})^{\top} \mathcal{I}(\boldsymbol{\theta}(\boldsymbol{\zeta})) J(\boldsymbol{\zeta}). \quad (3.16)$$

This formula is used quite frequently, e.g., in generalized linear models when changing the parametrization of the models.



### 3.3.2 Information Bound in the Exponential Family Case

The purpose of this section is to summarize the Cramér–Rao information bound results for the EF and the EDF, since these families play a distinguished role in statistical and actuarial modeling.

#### Cramér–Rao Information Bound in the EF Case

We start with the EF case. Assume we have i.i.d. observations  $Y_1, \dots, Y_n$  having densities w.r.t. a  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$  given by the EF, see (2.2),

$$dF(y; \theta) = f(y; \theta) d\nu(y) = \exp \left\{ \theta^\top T(y) - \kappa(\theta) + a(y) \right\} d\nu(y),$$

for canonical parameter  $\theta \in \Theta \subseteq \mathbb{R}^k$ . We assume to work under a minimal representation implying that the cumulant function  $\kappa$  is strictly convex on the interior  $\mathring{\Theta}$ , see Assumption 2.6. Moreover, we assume that the cumulant function  $\kappa$  is steep in the sense of Theorem 2.19. Consider the (aggregated) *statistics* of the joint EF  $\mathcal{P} = \{P(\cdot; \theta); \theta \in \Theta\}$

$$y \mapsto S(y) \stackrel{\text{def.}}{=} \left( \sum_{i=1}^n T_1(y_i), \dots, \sum_{i=1}^n T_k(y_i) \right)^\top \in \mathbb{R}^k. \quad (3.17)$$

We calculate the score of this EF

$$s(\theta, Y) = \nabla_\theta \log p(Y; \theta) = \nabla_\theta \left( \theta^\top \sum_{i=1}^n T(Y_i) - n\kappa(\theta) \right) = S(Y) - n\nabla_\theta \kappa(\theta).$$

An immediate consequence of Corollary 2.5 is that the expected value of the score is zero for any  $\theta \in \mathring{\Theta}$ . This then reads as

$$\mu = \mathbb{E}_\theta [T(Y_1)] = \mathbb{E}_\theta [S(Y)/n] = \nabla_\theta \kappa(\theta) \in \mathbb{R}^k. \quad (3.18)$$

Thus, the statistics  $S(Y)/n$  is an unbiased decision rule for the mean  $\mu = \nabla_\theta \kappa(\theta)$ , and we can study its Cramér–Rao information bound. Fisher’s information matrix is given by the positive definite matrix

$$\mathcal{I}(\theta) = \mathcal{I}_n(\theta) = \mathbb{E}_\theta [s(\theta, Y)s(\theta, Y)^\top] = -\mathbb{E}_\theta [\nabla_\theta^2 \log p(Y; \theta)] = n\nabla_\theta^2 \kappa(\theta) \in \mathbb{R}^{k \times k}.$$

Note that the multi-dimensionally extended Cramér–Rao information bound in Theorem 3.16 applies to the individual components of vector  $\mu = \nabla_\theta \kappa(\theta) \in \mathbb{R}^k$ . Assume we would like to estimate its  $j$ -th component, set  $\gamma_j(\theta) = \mu_j =$

$(\nabla_{\theta}\kappa(\theta))_j = \partial\kappa(\theta)/\partial\theta_j$ , for  $1 \leq j \leq k$ . This corresponds to the  $j$ -th component  $S_j(\mathbf{Y})$  of the statistics  $S(\mathbf{Y})$ . We have unbiasedness of  $S_j(\mathbf{Y})/n$  for  $\gamma_j(\theta) = \mu_j = (\nabla_{\theta}\kappa(\theta))_j$ , and this unbiased statistics attains the Cramér–Rao information bound

$$\text{Var}_{\theta}(S_j(\mathbf{Y})/n) = \frac{1}{n} \left( \nabla_{\theta}^2 \kappa(\theta) \right)_{j,j} = (\nabla_{\theta} \gamma_j(\theta))^{\top} \mathcal{I}(\theta)^{-1} (\nabla_{\theta} \gamma_j(\theta)). \quad (3.19)$$

Recall that  $\mathcal{I}(\theta)^{-1}$  scales as  $n^{-1}$ , see (3.15). This provides us with the following corollary.

**Corollary 3.18** *Assume  $Y_1, \dots, Y_n$  are i.i.d. and follow an EF (under a minimal representation). The components of the statistics  $S(\mathbf{Y})/n$  are UMVU for  $\gamma_j(\theta) = \partial\kappa(\theta)/\partial\theta_j$ ,  $1 \leq j \leq k$  and  $\theta \in \Theta$ , with*

$$\text{Var}_{\theta} \left( \frac{1}{n} S_j(\mathbf{Y}) \right) = \frac{1}{n} \frac{\partial^2}{\partial \theta_j^2} \kappa(\theta).$$

*The corresponding covariance terms are for  $1 \leq j, l \leq k$  given by*

$$\text{Cov}_{\theta} \left( \frac{1}{n} S_j(\mathbf{Y}), \frac{1}{n} S_l(\mathbf{Y}) \right) = \frac{1}{n} \frac{\partial^2}{\partial \theta_j \partial \theta_l} \kappa(\theta).$$

The UMVU property stated in Corollary 3.18 is, in general, not related to MLE, but within the EF there is the following link. We have (subject to existence)

$$\hat{\theta}^{\text{MLE}} = \arg \max_{\tilde{\theta} \in \Theta} p(\mathbf{Y}; \tilde{\theta}) = \arg \max_{\tilde{\theta} \in \Theta} \left( \tilde{\theta}^{\top} S(\mathbf{Y}) - n\kappa(\tilde{\theta}) \right) = h \left( \frac{1}{n} S(\mathbf{Y}) \right), \quad (3.20)$$

where  $h = (\nabla_{\theta}\kappa)^{-1}$  is the canonical link of this EF, see Definition 2.8; and where we need to ensure that a solution to (3.20) exists; e.g., the solution to (3.20) might be at the boundary of  $\Theta$  which may cause problems, see Example 3.5.<sup>1</sup> Because the cumulant function  $\kappa$  is strictly convex (in a minimal representation), we receive the

<sup>1</sup> Another example where there does not exist a proper solution to the MLE problem (3.20) is, for instance, obtained within the 2-dimensional Gaussian EF if we have only one single observation  $Y_1$ . Intuitively this is clear because we cannot estimate two parameters from one observation  $T(Y_1) = (Y_1, Y_1^2)$ .

MLE for the mean parameter  $\mu = \mathbb{E}_\theta [T(Y_1)]$

$$\hat{\mu}^{\text{MLE}} = \arg \max_{\tilde{\mu} \in \mathcal{M}} \left( h(\tilde{\mu})^\top S(\mathbf{Y}) - n\kappa(h(\tilde{\mu})) \right) = \frac{1}{n} S(\mathbf{Y}),$$

the dual parameter space  $\mathcal{M} = \nabla_\theta \kappa(\Theta) \subseteq \mathbb{R}^k$  has been introduced in Remarks 2.9. If  $S(\mathbf{Y})/n$  is contained in  $\mathcal{M}$ , then this MLE is a proper solution; otherwise, because we have assumed that the cumulant function  $\kappa$  is steep, the MLE exists in the closure  $\overline{\mathcal{M}}$ , see Theorem 2.19, and it is UMVU for  $\mu$ , see Corollary 3.18.

**Corollary 3.19 (Balance Property)** *Assume  $Y_1, \dots, Y_n$  are i.i.d. and follow an EF with  $\theta \in \Theta$  and  $T(Y_i) \in \overline{\mathcal{M}}$ , a.s. The MLE  $\hat{\mu}^{\text{MLE}} \in \overline{\mathcal{M}}$  is UMVU for  $\mu$ , and it fulfills the balance property on portfolio level, i.e.,*

$$\sum_{i=1}^n \mathbb{E}_{\hat{\mu}^{\text{MLE}}} [T(Y_i)] = n\hat{\mu}^{\text{MLE}} = S(\mathbf{Y}).$$

*Remarks 3.20*

- The balance property is a very important property in insurance pricing because it implies that the portfolio is priced on the right level: we have unbiasedness

$$\mathbb{E}_\theta \left[ \sum_{i=1}^n \mathbb{E}_{\hat{\mu}^{\text{MLE}}} [T(Y_i)] \right] = \mathbb{E}_\theta [S(\mathbf{Y})] = n\mu. \quad (3.21)$$

- We emphasize that the balance property is much stronger than unbiasedness (3.21), note that the balance property provides unbiasedness even if  $\mathbf{Y}$  follows a completely different model, i.e., even if the chosen EF  $\mathcal{P}$  is completely misspecified.
- In general, the MLE  $\hat{\theta}^{\text{MLE}}$  is not unbiased for  $\theta$ . E.g., if the canonical link  $h = (\nabla_\theta \kappa)^{-1}$  is strictly concave, we have from Jensen's inequality, subject to existence at the boundary of  $\Theta$ ,

$$\mathbb{E}_\theta [\hat{\theta}^{\text{MLE}}] = \mathbb{E}_\theta \left[ h \left( \frac{1}{n} S(\mathbf{Y}_n) \right) \right] < h \left( \mathbb{E}_\theta \left[ \frac{1}{n} S(\mathbf{Y}_n) \right] \right) = h(\mu) = \theta. \quad (3.22)$$

- The statistics  $S(\mathbf{Y})$  is a sufficient statistics of  $\mathbf{Y}$ , this follows from the factorization criterion; see Theorem 1.5.2 of Lehmann [244].

### Cramér–Rao Information Bound in the EDF Case

The single-parameter linear EDF case is very similar to the above vector-valued parameter EF case. We briefly summarize the main results in the EDF case.

Recall Example 3.5: assume that  $Y_1, \dots, Y_n$  are independent having densities w.r.t. a  $\sigma$ -finite measures on  $\mathbb{R}$  (not being concentrated in a single point) given by, see (2.14),

$$Y_i \sim f(y_i; \theta, v_i/\varphi) = \exp \left\{ \frac{y_i \theta - \kappa(\theta)}{\varphi/v_i} + a(y_i; v_i/\varphi) \right\}, \quad (3.23)$$

for  $1 \leq i \leq n$ . Note that these random variables are not i.i.d. because they may differ in the exposures  $v_i > 0$ . The MLE of  $\mu = \kappa'(\theta)$ ,  $\theta \in \mathring{\Theta}$ , is found by, see (3.5),

$$\hat{\mu}^{\text{MLE}} = \arg \max_{\tilde{\mu} \in \overline{\mathcal{M}}} \sum_{i=1}^n \frac{Y_i h(\tilde{\mu}) - \kappa(h(\tilde{\mu}))}{\varphi/v_i} = \frac{\sum_{i=1}^n v_i Y_i}{\sum_{i=1}^n v_i}, \quad (3.24)$$

we assume that  $\kappa$  is steep to ensure  $\hat{\mu}^{\text{MLE}} \in \overline{\mathcal{M}}$ . The convolution formula of Corollary 2.15 says that the MLE  $\hat{\mu}^{\text{MLE}} = Y_+$  belongs to the same EDF with the same canonical parameter  $\theta$  and the same dispersion  $\varphi$ , only the weight changes to  $v_+ = \sum_{i=1}^n v_i$ .

**Corollary 3.21 (Balance Property)** *Assume  $Y_1, \dots, Y_n$  are independent with EDF distribution (3.23) for  $\theta \in \mathring{\Theta}$  and  $Y_i \in \overline{\mathcal{M}}$ , a.s. The MLE  $\hat{\mu}^{\text{MLE}} \in \overline{\mathcal{M}}$  is UMVU for  $\mu = \kappa'(\theta)$ , and it fulfills the balance property on portfolio level, i.e.,*

$$\sum_{i=1}^n \mathbb{E}_{\hat{\mu}^{\text{MLE}}} [v_i Y_i] = \sum_{i=1}^n v_i \hat{\mu}^{\text{MLE}} = \sum_{i=1}^n v_i Y_i.$$

The score in this EDF is given by

$$s(\theta, \mathbf{Y}) = \frac{\partial}{\partial \theta} \log p(\mathbf{Y}; \theta) = \frac{\partial}{\partial \theta} \sum_{i=1}^n \frac{v_i}{\varphi} (\theta Y_i - \kappa(\theta)) = \sum_{i=1}^n \frac{v_i}{\varphi} (Y_i - \kappa'(\theta)).$$

Of course, we have  $\mathbb{E}_\theta[s(\theta, \mathbf{Y})] = 0$  and we receive Fisher's information for  $\theta \in \mathring{\Theta}$

$$\mathcal{I}(\theta) = -\mathbb{E}_\theta \left[ \frac{\partial^2}{\partial \theta^2} \log p(\mathbf{Y}; \theta) \right] = \sum_{i=1}^n \frac{v_i}{\varphi} \kappa''(\theta) > 0. \quad (3.25)$$

Corollary 2.15 gives for the variance of the MLE

$$\text{Var}_\theta \left( \hat{\mu}^{\text{MLE}} \right) = \frac{\varphi}{\sum_{i=1}^n v_i} \kappa''(\theta) = \frac{(\kappa''(\theta))^2}{\mathcal{I}(\theta)} = \frac{(\partial \mu(\theta) / \partial \theta)^2}{\mathcal{I}(\theta)}.$$

This verifies that  $\hat{\mu}^{\text{MLE}}$  meets the Cramér–Rao information bound and is UMVU for the mean  $\mu = \kappa'(\theta)$ .

*Example 3.22 (Poisson Case)* For this example, we consider independent Poisson random variables  $N_i \sim \text{Poi}(v_i \lambda)$ . In Sect. 2.2.2 we have seen that  $Y_i = N_i / v_i$  can be modeled within the single-parameter linear EDF framework using as cumulant function the exponential function  $\kappa(\theta) = e^\theta$ , and setting  $\omega_i = v_i$  and  $\varphi = 1$ . Thus, the probability weights of a single observation  $Y_i$  are given by, see (2.15),

$$f(y_i; \theta, v_i) = \exp \left\{ v_i (\theta y_i - e^\theta) + a(y_i; v_i) \right\},$$

with canonical parameter  $\theta = \log(\lambda) \in \Theta = \mathbb{R}$ . The MLE in the mean parametrization is given by, see (3.24),

$$\hat{\lambda}^{\text{MLE}} = \frac{\sum_{i=1}^n v_i Y_i}{\sum_{i=1}^n v_i} = \frac{\sum_{i=1}^n N_i}{\sum_{i=1}^n v_i} \in \overline{\mathcal{M}} = [0, \infty).$$

This estimator is unbiased for  $\lambda$ . Having independent Poisson random variables we can calculate the variance of this estimator as

$$\text{Var} \left( \hat{\lambda}^{\text{MLE}} \right) = \frac{\lambda}{\sum_{i=1}^n v_i}.$$

Moreover, from Corollary 3.21 we know that this estimator is UMVU for  $\lambda$ , which can easily be seen, and uses Fisher's information (3.25) with dispersion parameter  $\varphi = 1$

$$\mathcal{I}(\theta) = -\mathbb{E}_\theta \left[ \frac{\partial^2}{\partial \theta^2} \log p(\mathbf{Y}; \theta) \right] = \sum_{i=1}^n v_i \kappa''(\theta) = \lambda \sum_{i=1}^n v_i.$$

■

One could study many other properties of decision rules (and corresponding estimators), for instance, admissibility or uniformly minimum risk equivariance (UMRE), and we could also study other families of distribution functions such as group families. We refrain from doing so because we will not need this for our purposes.

### 3.4 Asymptotic Behavior of Estimators

All results above have been based on a finite sample  $\mathbf{Y}_n = (Y_1, \dots, Y_n)^\top$ , we add a lower index  $n$  to  $\mathbf{Y}_n$  to indicate the finite sample size  $n \in \mathbb{N}$ . The aim of this section is to analyze properties of decision rules when the sample size  $n$  tends to infinity.

#### 3.4.1 Consistency

Assume we have an infinite sequence of observations  $Y_i, i \geq 1$ , which allows us to construct an infinite sequence of decision rules  $A_n = A_n(\mathbf{Y}_n), n \geq 1$ , where  $A_n$  always considers the first  $n$  observations  $\mathbf{Y}_n = (Y_1, \dots, Y_n)^\top \sim P_n(\cdot; \theta)$ , for  $\theta \in \Theta$  not depending on  $n$ . To fix ideas, one may think of i.i.d. random variables  $Y_i$ .

**Definition 3.23 (Consistency)** The sequence  $A_n = A_n(\mathbf{Y}_n) \in \mathbb{R}^r, n \geq 1$ , is consistent for  $\gamma : \Theta \rightarrow \mathbb{R}^r$  if for all  $\theta \in \Theta$  and for all  $\varepsilon > 0$  we have

$$\lim_{n \rightarrow \infty} \mathbb{P}_\theta [\|A_n(\mathbf{Y}_n) - \gamma(\theta)\|_2 > \varepsilon] = 0.$$

Definition 3.23 says that  $A_n(\mathbf{Y}_n)$  converges in probability to  $\gamma(\theta)$  as  $n \rightarrow \infty$ . If we (even) have a.s. convergence, we call  $A_n, n \geq 1$ , *strongly consistent* for  $\gamma : \Theta \rightarrow \mathbb{R}^r$ . Consistency is a minimal property that decision rules should fulfill. Typically, in applications, this is not enough, and we are interested in (fast) rates of convergence, i.e., we would like to know the error rates between  $A_n(\mathbf{Y}_n)$  and  $\gamma(\theta)$  for  $n \rightarrow \infty$ .

*Example 3.24 (Consistency of the MLE in the EF)* We revisit Corollary 3.19 and consider an i.i.d. sequence of random variables  $Y_i, i \geq 1$ , belonging to an EF, and we assume to work under a minimal representation and to have a steep cumulant function  $\kappa$ . The MLE for  $\mu$  is given by the statistics

$$\hat{\mu}_n^{\text{MLE}} = \frac{1}{n} S(\mathbf{Y}_n) = \frac{1}{n} \sum_{i=1}^n (T_1(Y_i), \dots, T_k(Y_i))^\top \in \overline{\mathcal{M}}.$$

We add a lower index  $n$  to the MLE to indicate the sample size. The i.i.d. property of  $Y_i, i \geq 1$ , implies that we can apply the strong law of large numbers which tells us that we have  $\lim_{n \rightarrow \infty} \hat{\mu}_n^{\text{MLE}} = \mathbb{E}_\theta [T(Y_1)] = \nabla_\theta \kappa(\theta) = \mu$ , a.s., for all  $\theta \in \Theta$ . This implies strong consistency of the sequence of MLEs  $\hat{\mu}_n^{\text{MLE}}, n \geq 1$ , for  $\mu$ .

We have seen that these MLEs are also UMVU for  $\mu$ , but if we transform them to the canonical scale  $\hat{\theta}_n^{\text{MLE}}$  they are, in general, biased for  $\theta$ , see (3.22). However, since the cumulant function  $\kappa$  is strictly convex (under a minimal representation) we receive  $\lim_{n \rightarrow \infty} \hat{\theta}_n^{\text{MLE}} = \theta$ , a.s., which provides strong consistency also on the canonical scale. ■

**Proposition 3.25** Assume the real-valued random variables  $Y_i$ ,  $i \geq 1$ , are i.i.d.  $F(\cdot; \theta)$  distributed with fixed  $\theta \in \Theta$ . The resulting empirical distributions  $\widehat{F}_n$ ,  $n \geq 1$ , are given by (3.9). Assume  $Q$  is a Fisher-consistent functional for  $\gamma(\theta)$ , i.e.,  $Q(F(\cdot; \theta)) = \gamma(\theta)$  for all  $\theta \in \Theta$ . Moreover, assume that  $Q$  is continuous in  $F(\cdot; \theta)$ , for all  $\theta \in \Theta$ , w.r.t. the supremum norm. The functionals  $Q(\widehat{F}_n)$ ,  $n \geq 1$ , are consistent for  $\gamma(\theta)$ .

**Sketch of Proof** The Glivenko–Cantelli theorem [64, 159] says that the empirical distribution  $\widehat{F}_n$  converges uniformly to  $F(\cdot; \theta)$ , a.s., for  $n \rightarrow \infty$ . Using the assumptions made, we are allowed to exchange the corresponding limits, which provides consistency.  $\square$

In view of Proposition 3.25, we discuss the case of the MLE of  $\theta \in \Theta$ . In Example 3.10 we have seen that the MLE of  $\theta \in \Theta$  is obtained from a Fisher-consistent functional  $Q$  for  $\theta$  on the set of probability distributions  $\mathfrak{P}$  given by

$$Q(F) = \arg \max_{\tilde{\theta}} \int \log f(y; \tilde{\theta}) dF(y) = \arg \min_{\tilde{\theta}} D_{\text{KL}}(f || f(\cdot; \tilde{\theta})),$$

in the second step we assumed that  $F$  has a density  $f$  w.r.t. a  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$ .

Assume we have i.i.d. data  $Y_i \sim f(\cdot; \theta)$ ,  $i \geq 1$ . Thus, the true data generating distribution is described by the parameter  $\theta \in \Theta$ . MLE requires the study of the log-likelihood function (we scale with the sample size  $n$ )

$$\tilde{\theta} \mapsto \frac{1}{n} \ell_{Y_n}(\tilde{\theta}) = \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \tilde{\theta}).$$

The law of large numbers gives us, a.s.,

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \tilde{\theta}) = \mathbb{E}_{\theta} [\log f(Y; \tilde{\theta})]. \quad (3.26)$$

Thus, if we are allowed to exchange the arg max operation and the limit in  $n \rightarrow \infty$  we receive, a.s.,

$$\begin{aligned} \lim_{n \rightarrow \infty} \widehat{\theta}_n^{\text{MLE}} &= \lim_{n \rightarrow \infty} \left( \arg \max_{\tilde{\theta}} \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \tilde{\theta}) \right) \\ &\stackrel{?}{=} \arg \max_{\tilde{\theta}} \left( \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \tilde{\theta}) \right) \\ &= \arg \max_{\tilde{\theta}} \mathbb{E}_{\theta} [\log f(Y; \tilde{\theta})] = Q(F(\cdot; \theta)) = \theta. \end{aligned} \quad (3.27)$$

That is, we receive consistency of the MLE for  $\theta$  if we are allowed to exchange the arg max operation and the limit in  $n \rightarrow \infty$ . This requires regularity conditions on the considered family of distributions  $\mathcal{F} = \{F(\cdot; \theta); \theta \in \Theta\}$ . The case of a finite parameter space  $\Theta = \{\theta_1, \dots, \theta_J\}$  is easy, this is a simplified version of Wald's [374] consistency proof,

$$\mathbb{P}_{\theta_j} \left[ \theta_j \notin \arg \max_{\theta_k} \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \theta_k) \right] \leq \sum_{k \neq j} \mathbb{P}_{\theta_j} \left[ \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \theta_k) > \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \theta_j) \right].$$

The right-hand side converges to 0 as  $n \rightarrow \infty$  for all  $\theta_k \neq \theta_j$ , which gives consistency. For regularity conditions on more general parameter spaces we refer to Section 5.2 in Van der Vaart [363]. Basically, one needs that the arg max of the limiting function given on the right-hand side of (3.26) is well-separated from other large values of that function, see Theorem 5.7 in Van der Vaart [363].

#### Remarks 3.26

- The estimator from the arg max operation in (3.27) is also called M-estimator, and  $(y, a) \mapsto \log(f(y; a))$  plays the role of a scoring function (similar to a loss function). The last line of (3.27) says that this scoring function is strictly consistent for the functional  $Q : \mathcal{F} \rightarrow \Theta$ , and Fisher-consistency of this functional  $Q$  implies

$$\mathbb{E}_{\tilde{\theta}} [\log f(Y; \tilde{\theta})] \leq \mathbb{E}_{\theta} [\log f(Y; Q(F(\cdot; \theta)))] = \mathbb{E}_{\theta} [\log f(Y; \theta)],$$

for all  $\tilde{\theta} \in \Theta$ . Strict consistency of loss and scoring functions is going to be defined formally in Sect. 4.1.3, below, and we have just seen that this plays an important role for the consistency of M-estimators in the sense of Definition 3.23.

- Consistency (3.27) assumes that the data generating model  $Y \sim F$  belongs to the specified family  $\mathcal{F} = \{F(\cdot; \theta); \theta \in \Theta\}$ . Model uncertainty may imply that the data generating model does not belong to  $\mathcal{F}$ . In this situation, and if we are allowed to exchange the arg max operation and the limit in  $n$  in (3.27), the MLE will provide the model in  $\mathcal{F}$  that is closest in KL divergence to the true model  $F$ . We come back to this in Sect. 11.1.4, below.

### 3.4.2 Asymptotic Normality

As mentioned above, typically, we would like to have stronger results than just consistency. We give an introductory example based on the EF.

*Example 3.27 (Asymptotic Normality of the MLE in the EF)* We work under the same EF as in Example 3.24. This example has provided consistency of the sequence of MLEs  $\hat{\mu}_n^{\text{MLE}}$ ,  $n \geq 1$ , for  $\mu$ . Note that the i.i.d. property together with the finite



variance property immediately implies the following convergence in distribution

$$\sqrt{n} \left( \hat{\mu}_n^{\text{MLE}} - \mu \right) \Rightarrow \mathcal{N}(0, \nabla_{\theta}^2 \kappa(\theta)) \stackrel{(d)}{=} \mathcal{N}(0, \mathcal{I}_1(\theta)) \quad \text{as } n \rightarrow \infty,$$

where  $\theta = \theta(\mu) = (\nabla_{\theta} \kappa)^{-1}(\mu) \in \Theta$  for  $\mu \in \mathcal{M}$ , and  $\mathcal{N}$  denotes the Gaussian distribution. This is the multivariate version of the central limit theorem (CLT), and it tells us that the rate of convergence is  $1/\sqrt{n}$ . This asymptotic result is stated in terms of Fisher's information matrix under parametrization  $\theta$ . We transform this to the dual mean parametrization and call Fisher's information matrix under the dual mean parametrization  $\mathcal{I}_1^*(\mu)$ . This involves the change of variable  $\mu \mapsto \theta = \theta(\mu) = (\nabla_{\theta} \kappa)^{-1}(\mu)$ . The Jacobian matrix of this change of variable is given by  $J(\mu) = \mathcal{I}_1(\theta(\mu))^{-1}$  and, thus, the transformation of Fisher's information matrix gives, see also (3.16),

$$\mu \mapsto \mathcal{I}_1^*(\mu) = J(\mu)^{\top} \mathcal{I}_1(\theta(\mu)) J(\mu) = \mathcal{I}_1(\theta(\mu))^{-1}.$$

This allows us to express the above CLT w.r.t. Fisher's information matrix corresponding to  $\mu$  and it gives us

$$\sqrt{n} \left( \hat{\mu}_n^{\text{MLE}} - \mu \right) \Rightarrow \mathcal{N}\left(0, \mathcal{I}_1^*(\mu)^{-1}\right) \quad \text{as } n \rightarrow \infty. \quad (3.28)$$

We conclude that the appropriately normalized MLE  $\hat{\mu}_n^{\text{MLE}}$  converges in distribution to the centered Gaussian distribution having as covariance matrix the *inverse of Fisher's information matrix*  $\mathcal{I}_1^*(\mu)$ , and the *rate of convergence* is  $1/\sqrt{n}$ .

Assume that the effective domain  $\Theta$  is open, and that  $\theta = \theta(\mu) \in \Theta$ . This allows us to transform asymptotic normality (3.28) to the canonical scale. Consider again the change of variable  $\mu \mapsto \theta = \theta(\mu) = (\nabla_{\theta} \kappa)^{-1}(\mu)$  with Jacobian matrix  $J(\mu) = \mathcal{I}_1(\theta(\mu))^{-1} = \mathcal{I}_1^*(\mu)$ . Theorem 1.9 in Section 5.2 of Lehmann [244] tells us how the CLT transforms under such a change of variable, namely,

$$\begin{aligned} \sqrt{n} \left( \hat{\theta}_n^{\text{MLE}} - \theta \right) &= \sqrt{n} \left( (\nabla_{\theta} \kappa)^{-1} \left( \hat{\mu}_n^{\text{MLE}} \right) - (\nabla_{\theta} \kappa)^{-1}(\mu) \right) \\ &\Rightarrow \mathcal{N}\left(0, J(\mu) \mathcal{I}_1^*(\mu)^{-1} J(\mu)\right) \stackrel{(d)}{=} \mathcal{N}\left(0, \mathcal{I}_1(\theta)^{-1}\right) \quad \text{as } n \rightarrow \infty. \end{aligned} \quad (3.29)$$

We have exactly the same structural form in the two asymptotic results (3.28) and (3.29). There is a main difference,  $\hat{\mu}_n^{\text{MLE}}$  is unbiased for  $\mu$  whereas, in general,  $\hat{\theta}_n^{\text{MLE}}$  is not unbiased for  $\theta$ , but we receive the same asymptotic behavior. ■

There are many different versions of asymptotic normality results similar to (3.28) and (3.29), and the main difficulty often is to verify the assumptions made. For instance, one can prove asymptotic normality based on a Fisher-consistent functional  $Q$ . The assumptions made are, among others, that  $Q$  needs to be Fréchet differentiable in  $P(\cdot; \theta)$  which, unfortunately, is rather difficult to verify. We make a list of assumptions here that are easier to check and then we give a version of the asymptotic normality result which is stated in the book of Lehmann [244]. This list of assumptions in the one-dimensional case  $\Theta \subseteq \mathbb{R}$  reads as follows:

- (i)  $\Theta \subseteq \mathbb{R}$  is an open interval (possibly infinite).
- (ii) The real-valued random variables  $Y_i \sim F(\cdot; \theta)$ ,  $i \geq 1$ , have common support  $\mathfrak{T} = \{y \in \mathbb{R}; f(y; \theta) > 0\}$  which is independent of  $\theta \in \Theta$ .
- (iii) For every  $y \in \mathfrak{T}$ , the density  $f(y; \theta)$  is three times continuously differentiable in  $\theta$ .
- (iv) The integral  $\int f(y; \theta) d\nu(y)$  is twice differentiable under the integral sign.
- (v) Fisher's information satisfies  $\mathcal{I}_1(\theta) = \mathbb{E}_\theta[(\partial \log f(Y_1; \theta)/\partial \theta)^2] \in (0, \infty)$ .
- (vi) For every  $\theta_0 \in \Theta$  there exist a positive constant  $c$  and a function  $M(y)$  (both may depend on  $\theta_0$ ) such that  $\mathbb{E}_{\theta_0}[M(Y_1)] < \infty$  and

$$\left| \frac{\partial^3}{\partial \theta^3} \log f(y; \theta) \right| \leq M(y) \quad \text{for all } y \in \mathfrak{T} \text{ and } \theta \in (\theta_0 - c, \theta_0 + c).$$

**Theorem 3.28 (Theorem 2.3 in Section 6.2 of Lehmann [244])** Assume  $Y_i$ ,  $i \geq 1$ , are i.i.d.  $F(\cdot; \theta)$  distributed satisfying (i)–(vi) from above. Assume that  $\hat{\theta}_n = \hat{\theta}_n(\mathbf{Y}_n)$ ,  $n \geq 1$ , is a sequence of roots that solves the score equations

$$\frac{\partial}{\partial \tilde{\theta}} \sum_{i=1}^n \log f(Y_i; \tilde{\theta}) = \frac{\partial}{\partial \tilde{\theta}} \ell_{Y_n}(\tilde{\theta}) = 0,$$

and which is consistent for  $\theta$ , i.e. this sequence of roots  $\hat{\theta}_n(\mathbf{Y}_n)$  converges in probability to the true parameter  $\theta$ . Then we have asymptotic normality

$$\sqrt{n}(\hat{\theta}_n - \theta) \Rightarrow \mathcal{N}(0, \mathcal{I}_1(\theta)^{-1}) \quad \text{as } n \rightarrow \infty. \quad (3.30)$$

**Sketch of Proof** Fix  $\theta \in \Theta$  and consider a Taylor expansion of the score  $\ell'_{Y_n}(\cdot)$  in  $\theta$  for  $\hat{\theta}_n$ . It is given by

$$\ell'_{Y_n}(\hat{\theta}_n) = \ell'_{Y_n}(\theta) + \ell''_{Y_n}(\theta)(\hat{\theta}_n - \theta) + \frac{1}{2} \ell'''_{Y_n}(\theta_n)(\hat{\theta}_n - \theta)^2,$$

for  $\theta_n \in [\theta, \widehat{\theta}_n]$ . Since  $\widehat{\theta}_n$  is a root of the score, the left-hand side is equal to zero. This allows us to re-arrange the above Taylor expansion as follows

$$\sqrt{n}(\widehat{\theta}_n - \theta) = \frac{\frac{1}{\sqrt{n}}\ell'_{Y_n}(\theta)}{-\frac{1}{n}\ell''_{Y_n}(\theta) - \frac{1}{2n}\ell'''_{Y_n}(\theta_n)(\widehat{\theta}_n - \theta)}.$$

The enumerator on the right-hand side converges in distribution to  $\mathcal{N}(0, \mathcal{I}_1(\theta))$ , see (18) in Section 6.2 of [244], the first term in the denominator converges in probability to  $\mathcal{I}_1(\theta)$ , see (19) in Section 6.2 of [244], and in the second term of the denominator we have  $\frac{1}{2n}\ell'''_{Y_n}(\theta_n)$  which is bounded in probability, see (20) in Section 6.2 of [244]. The claim then follows from Slutsky's theorem.  $\square$

### Remarks 3.29

- A sequence  $(\widehat{\theta}_n)_{n \geq 1}$  satisfying Theorem 3.28 is called *efficient likelihood estimator* (ELE) of  $\theta$ . Typically, the sequence of MLEs  $\widehat{\theta}_n^{\text{MLE}}$  gives such an ELE sequence, but there are counterexamples where this is not the case, see Example 3.1 in Section 6.2 of Lehmann [244]. In that example  $\widehat{\theta}_n^{\text{MLE}}$  exists for all  $n \geq 1$ , but it converges in probability to  $\infty$ , regardless of the value of the true parameter  $\theta$ .
- Any sequence of estimators that fulfills (3.30) is called *asymptotically efficient*, because, similarly to the Cramér–Rao information bound of Theorem 3.13, it attains  $\mathcal{I}_1(\theta)^{-1}$  (which under certain assumptions is a lower variance bound except on Lebesgue measure zero, see Theorem 1.1 in Section 6.1 of Lehmann [244]). However, there are two important differences here: (1) the Cramér–Rao information bound statement needs unbiasedness of the decision rule, whereas (3.30) only requires consistency (but not unbiasedness nor asymptotically vanishing bias); and (2) the lower bound in the Cramér–Rao statement is an effective variance (on a finite sample), whereas the quantity in (3.30) is only an asymptotic variance. Moreover, any other sequence that differs in probability from an asymptotically efficient one less than  $o(1/\sqrt{n})$  is asymptotically efficient, too.
- If we consider a differentiable function  $\theta \mapsto \gamma(\theta)$ , then Theorem 3.28 implies

$$\sqrt{n}(\gamma(\widehat{\theta}_n) - \gamma(\theta)) \Rightarrow \mathcal{N}\left(0, \frac{(\gamma'(\theta))^2}{\mathcal{I}_1(\theta)}\right) \quad \text{as } n \rightarrow \infty. \quad (3.31)$$

This follows from asymptotic normality, consistency and considering a Taylor expansion around  $\theta$ .

- We were starting from the MLE problem

$$\widehat{\theta}_n^{\text{MLE}} = \arg \max_{\tilde{\theta}} \frac{1}{n} \sum_{i=1}^n \log f(Y_i; \tilde{\theta}). \quad (3.32)$$

In statistical theory a parameter estimator that is obtained through a maximization operation is called M-estimator (for maximizing or minimizing), see also Remarks 3.26. If the log-likelihood is differentiable in  $\tilde{\theta}$  we can turn the above problem into a root search problem for  $\tilde{\theta}$

$$\frac{1}{n} \sum_{i=1}^n \frac{\partial}{\partial \tilde{\theta}} \log f(Y_i; \tilde{\theta}) = 0. \quad (3.33)$$

If a parameter estimator is obtained through a root search problem it is called Z-estimator (for equating to zero). The Z-estimator (3.33) does not require a maximum of the original function, but only a critical point; this is exactly what we have been exploring in Theorem 3.28. More generally, for a sufficiently nice function  $\psi(\cdot; \theta)$  a Z-estimator  $\tilde{\theta}_n^Z$  for  $\theta$  is obtained by solving the following equation for  $\tilde{\theta}$

$$\frac{1}{n} \sum_{i=1}^n \psi(Y_i; \tilde{\theta}) = 0, \quad (3.34)$$

for i.i.d. data  $Y_i \sim F(\cdot; \theta)$ . Suppose that the first moment of  $\psi(Y_i; \tilde{\theta})$  exists. The law of large numbers gives us, a.s., see also (3.26),

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n \psi(Y_i; \tilde{\theta}) = \mathbb{E}_{\theta} [\psi(Y; \tilde{\theta})]. \quad (3.35)$$

Consistency of the Z-estimator  $\tilde{\theta}_n^Z$ ,  $n \geq 1$ , for  $\theta$  is related to the right-hand side of (3.35) being zero for  $\tilde{\theta} = \theta$ . Under additional regularity conditions (and consistency) it then holds asymptotic normality

$$\sqrt{n} \left( \tilde{\theta}_n^Z - \theta \right) \Rightarrow \mathcal{N} \left( 0, \frac{\mathbb{E}_{\theta} [\psi(Y; \theta)^2]}{\mathbb{E}_{\theta} \left[ \frac{\partial}{\partial \theta} \psi(Y; \theta) \right]^2} \right) \quad \text{as } n \rightarrow \infty. \quad (3.36)$$

For rigorous statements we refer to Theorems 5.21 and 5.41 in Van der Vaart [363]. A modification to the regression case is given in Theorem 11.6 below.

*Example 3.30* We consider the single-parameter linear EF for given strictly convex and steep cumulant function  $\kappa$  and w.r.t. a  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$ . The score equation gives requirement

$$\frac{1}{n} S(Y_n) \stackrel{!}{=} \kappa'(\theta) = \mathbb{E}_{\theta}[Y_1]. \quad (3.37)$$

Strict convexity implies that the right-hand side strictly increases in  $\theta$ . Therefore, we have at most one solution of the score equation here. We assume that the

effective domain  $\Theta \subseteq \mathbb{R}$  is open. It is easily verified that assumptions (ii)–(vi) hold, in particular, (vi) saying that the third derivative should have a uniformly bounded integrable bound holds because the third derivative is independent of  $y$  and continuous in  $\theta$ . With probability converging to 1, (3.37) has a solution  $\hat{\theta}_n$  which is unique, consistent and Theorem 3.28 holds. Note that in Example 3.5 we have mentioned the Poisson case which can be degenerate. For the asymptotic normality result we use here that this degeneracy asymptotically vanishes with probability converging to one. ■

**Remark 3.31 (Multi-Dimensional Extension)** For an extension of Theorem 3.28 to the multi-dimensional case  $\Theta \subseteq \mathbb{R}^k$  we refer to Section 6.4 in Lehmann [244]. The assumptions made in the multi-dimensional case do not essentially differ from the ones in the 1-dimensional case.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.



## Chapter 4

# Predictive Modeling and Forecast Evaluation



In the previous chapter, we have fully focused on parameter estimation  $\theta \in \Theta$  and the estimation of functions  $\theta \mapsto \gamma(\theta)$  by exploiting decision rules  $A$  for estimating  $Y_n \mapsto \hat{\theta} = A(Y_n)$  or  $Y_n \mapsto \hat{\gamma}(\theta) = A(Y_n)$ , respectively. The derivations in that chapter analyzed the quality of decision rules in terms of loss functions which compare, e.g., the action  $\hat{\theta} = A(Y_n)$  to the true parameter  $\theta$ . The Cramér–Rao information bound considers this in terms of a square loss function. In actuarial modeling, parameter estimation is only part of the problem, and the second part is to predict new random variables  $Y$ . These new random variables should be thought as claims in the future that we try to predict (and price) using decision rules being developed based on past information  $Y_n = (Y_1, \dots, Y_n)^\top$ . In this case, we would like to study how a decision rule  $A(Y_n)$  *generalizes* to new data  $Y$ , and we then call the decision rule rather a *predictor* for  $Y$ . This capability of suitable decision rules to generalize to new (unseen) data is analyzed in Sect. 4.1. Such an analysis often relies on (numerical) techniques such as cross-validation, which is examined in Sect. 4.2, or the bootstrap technique, being presented in Sect. 4.3, below. In this chapter, we denote past observations by  $Y_n = (Y_1, \dots, Y_n)^\top$  supported on  $\mathbb{Y}$ , and the (real-valued) random variables to be predicted are denoted by  $Y$  with support  $\mathcal{Y} \subset \mathbb{R}$ . Often we have  $\mathbb{Y} = \mathcal{Y} \times \dots \times \mathcal{Y}$ .

### 4.1 Generalization Loss

We start by considering the most commonly used *expected generalization loss* (GL) which is the *mean squared error of prediction* (MSEP). The MSEP is based on the square loss function, and it can be seen as a distribution-free approach to measure expected GL. In subsequent sections we will study distribution-adapted GL approaches. Expected GL measurement with MSEP is considered to be general knowledge and we do not give a specific reference in this section. Distribution-

adapted versions are mainly based on the strictly consistent scoring framework of Gneiting–Raftery [163] and Gneiting [162]. In particular, we will discuss *deviance losses* in Sect. 4.1.2 that are strictly consistent scoring functions for mean estimation and, hence, provide proper scoring rules.

### 4.1.1 Mean Squared Error of Prediction

We denote by  $\mathbf{Y}_n = (Y_1, \dots, Y_n)^\top$  (past) observations on which predictors and decision rules  $A : \mathbb{Y} \rightarrow \mathbb{A}$  are based on. The new observation that we would like to predict is denoted by  $Y$  having support  $\mathcal{Y} \subset \mathbb{R}$ . In the previous chapter we have used decision rule the  $A(\mathbf{Y}_n)$  to estimate an unknown quantity  $\gamma(\theta)$ . In this section we will use this decision rule to directly predict the new (unseen) observation  $Y$ .

**Theorem 4.1 (Mean Squared Error of Prediction, MSEP)** *Assume that  $\mathbf{Y}_n$  and  $Y$  are independent. Assume that the predictor  $A : \mathbb{Y} \rightarrow \mathbb{A} \subseteq \mathbb{R}$ ,  $\mathbf{Y}_n \mapsto A(\mathbf{Y}_n)$  has finite second moment, and that the real-valued random variable  $Y$  has finite second moment, too. The MSEP of predictor  $A$  to predict  $Y$  is given by*

$$\mathbb{E}[(Y - A(\mathbf{Y}_n))^2] = (\mathbb{E}[Y] - \mathbb{E}[A(\mathbf{Y}_n)])^2 + \text{Var}(A(\mathbf{Y}_n)) + \text{Var}(Y). \quad (4.1)$$

**Proof of Theorem 4.1** We compute

$$\begin{aligned} \mathbb{E}[(A(\mathbf{Y}_n) - Y)^2] &= \mathbb{E}[(A(\mathbf{Y}_n) - \mathbb{E}[Y] + \mathbb{E}[Y] - Y)^2] \\ &= \mathbb{E}[(A(\mathbf{Y}_n) - \mathbb{E}[Y])^2] + \mathbb{E}[(\mathbb{E}[Y] - Y)^2] \\ &\quad + 2 \mathbb{E}[(A(\mathbf{Y}_n) - \mathbb{E}[Y])(\mathbb{E}[Y] - Y)] \\ &= \mathbb{E}[(\mathbb{E}[Y] - \mathbb{E}[A(\mathbf{Y}_n)] + \mathbb{E}[A(\mathbf{Y}_n)] - A(\mathbf{Y}_n))^2] + \text{Var}(Y) \\ &= (\mathbb{E}[Y] - \mathbb{E}[A(\mathbf{Y}_n)])^2 + \text{Var}(A(\mathbf{Y}_n)) + \text{Var}(Y), \end{aligned}$$

where on the second last line we use the independence between  $\mathbf{Y}_n$  and  $Y$ . This finishes the proof.  $\square$

**Remarks 4.2 (Expected Generalization Loss)**

- The quantity  $\mathbb{E}[(Y - A(\mathbf{Y}_n))^2]$  is an expected GL because it measures how well the decision rule (predictor)  $A(\mathbf{Y}_n)$  generalizes to new (unseen) data  $Y$ . As loss

function we use the square loss function

$$L : \mathcal{Y} \times \mathbb{A} \rightarrow \mathbb{R}_+, \quad (y, a) \mapsto L(y, a) = (y - a)^2. \quad (4.2)$$

Therefore, this expected GL is called MSEP.

- MSEP (4.1) is called *expected* GL. If we condition on  $Y_n$ , then we call it GL. For the square loss function the GL (conditional MSEP) is given by

$$\mathbb{E} \left[ (Y - A(Y_n))^2 \middle| Y_n \right] = (\mathbb{E}[Y] - A(Y_n))^2 + \text{Var}(Y), \quad (4.3)$$

where we have used independence between  $Y$  and  $Y_n$ .

- We do not distinguish the terms ‘prediction’ and ‘forecast’. Sometimes the literature makes a subtle difference between the two, the latter involving a temporal component and the former not. In the context of prediction/forecasting a loss function (4.2) is also called *scoring function*. We also use these two terms interchangeably in the context of prediction/forecasting.
- The MSEP in Theorem 4.1 decouples into three terms:
  - The first term  $(\mathbb{E}[Y] - \mathbb{E}[A(Y_n)])^2$  is the (squared) *bias*. Obviously, good decision rules  $A(Y_n)$  under the MSEP should be unbiased for  $\mathbb{E}[Y]$ . If we compare this to the previous chapter, we note that now the bias is measured w.r.t. the mean of the new observation  $Y$ . Additionally, there might be a slight difference to the previous chapter if  $Y_n$  and  $Y$  do not belong to the same parameter  $\theta \in \Theta$  (if we work in a parametrized family): the risk function in (3.3) considers  $\mathcal{R}(\theta, A) = \mathbb{E}_\theta[L(\theta, A(Y_n))]$  with both components of the loss function  $L$  belonging to the same parameter value  $\theta$ . For the MSEP we replace  $\theta$  in  $L(\theta, A(Y_n))$  by the new observation  $Y$  that might originate from a different distribution (or from a randomized  $\theta$  in a Bayesian case).
  - The second term  $\text{Var}(A(Y_n))$  is called *estimation variance* or *statistical error*.
  - The last term  $\text{Var}(Y)$  is called *process variance* or *irreducible risk*. It reflects the pure randomness received from the fact that we try to predict random variables  $Y$  with deterministic means  $\mathbb{E}[Y]$ .
- All three terms on the right-hand side of (4.1) are non-negative. The *MSEP optimal predictor* for  $Y$  is its expected value  $\mathbb{E}[Y]$ . For this choice, the first two terms (squared bias and estimation variance) vanish, and we are only left with the irreducible risk. Since this MSEP optimal predictor is typically unknown it is replaced by a decision rule  $A(Y_n)$  that is based on past experience  $Y_n$ . This decision rule is used to predict  $Y$ , but it can also be seen as an *estimator* for  $\mathbb{E}[Y]$ . A good decision rule  $A(Y_n)$  is unbiased for  $\mathbb{E}[Y]$ , making the first term on the right-hand side of (4.1) equal to zero, and at the same time trying to make the estimation variance small. Typically, this cannot be achieved simultaneously and, therefore, there is a trade-off between bias and estimation variance in most applied statistical problems.



- We emphasize that in financial applications we typically aim for unbiased estimators for  $\mathbb{E}[Y]$ , we especially refer to Sect. 7.4.2 that studies the balance property in network regression models under a stationary portfolio assumption. Here, this stationarity may, e.g., translate into a (stronger) i.i.d. assumption on  $Y_1, \dots, Y_n, Y$ . Unbiasedness then implies that the predictor  $A(Y_n)$  is optimal in (4.1) if it meets the Cramér–Rao information bound, see Theorem 3.13.

Theorem 4.1 considers the MSEP which implicitly assumes that the square loss function is the objective (scoring) function of interest. The square loss function may be considered as being distribution-free, but it is motivated by a Gaussian model for  $Y_n$  and  $Y$ , respectively; this will be justified in Remarks 4.6, below. If we use the square loss function for observations different from Gaussian ones it might under- or over-weigh particular characteristics in these observations because they may not look very Gaussian (e.g. more heavy-tailed). Therefore, we should always choose a scoring function that fits the problem considered, for instance, a square loss function is not appropriate if we model claim counts following a Poisson distribution. We close this section with the example of the EDF.

*Example 4.3 (MSEP Within the EDF)* We choose a fixed single-parameter linear EDF satisfying Assumption 2.6 and having a steep cumulant function  $\kappa$ , see Theorem 2.19 and Remark 2.20. Assume we have independent random variables  $Y_1, \dots, Y_n, Y$  belonging to this EDF having densities, see Example 3.5,

$$Y_i \sim f(y_i; \theta, v_i/\varphi) = \exp \left\{ \frac{y_i \theta - \kappa(\theta)}{\varphi/v_i} + a(y_i; v_i/\varphi) \right\}, \quad (4.4)$$

and similarly for  $Y \sim f(y; \theta, v/\varphi)$ . Note that all random variables share the same canonical parameter  $\theta \in \Theta$ . The MLE of  $\mu \in \mathcal{M}$  based on  $Y_n = (Y_1, \dots, Y_n)^\top$  is found by solving, see (3.4)–(3.5),

$$\begin{aligned} \hat{\mu}^{\text{MLE}} &= \hat{\mu}^{\text{MLE}}(Y_n) = \arg \max_{\tilde{\mu} \in \overline{\mathcal{M}}} \ell_{Y_n}(\tilde{\mu}) \\ &= \arg \max_{\tilde{\mu} \in \overline{\mathcal{M}}} \sum_{i=1}^n \frac{Y_i h(\tilde{\mu}) - \kappa(h(\tilde{\mu}))}{\varphi/v_i}, \end{aligned} \quad (4.5)$$

with canonical link  $h = (\kappa')^{-1}$ . Since the cumulant function  $\kappa$  is strictly convex and assumed to be steep, there exists a unique solution  $\hat{\mu}^{\text{MLE}} \in \overline{\mathcal{M}}$ . If  $\hat{\mu}^{\text{MLE}} \in \mathcal{M}$  we have a proper solution providing  $\hat{\theta}^{\text{MLE}} = h(\hat{\mu}^{\text{MLE}}) \in \Theta$ , otherwise  $\hat{\mu}^{\text{MLE}}$  provides a degenerate model. This decision rule  $Y_n \mapsto \hat{\mu}^{\text{MLE}} = \hat{\mu}^{\text{MLE}}(Y_n)$  is now used to predict the (independent) new random variable  $Y$  and to estimate the unknown parameters  $\theta$  and  $\mu$ , respectively. That is, we use the following predictor for  $Y$

$$Y_n \mapsto \hat{Y} = \hat{\mathbb{E}}_\theta[Y] = \mathbb{E}_{\hat{\theta}^{\text{MLE}}}[Y] = \hat{\mu}^{\text{MLE}} = \hat{\mu}^{\text{MLE}}(Y_n).$$

Note that this predictor  $\hat{Y}$  is used to predict an unobserved (new) random variable  $Y$ , and it is itself a random variable as a function of (independent) past observations  $Y_n$ . We calculate the MSE in this model. Using Theorem 4.1 we obtain

$$\begin{aligned} \mathbb{E}_\theta \left[ \left( Y - \hat{\mu}^{\text{MLE}} \right)^2 \right] &= \left( \mathbb{E}_\theta [Y] - \mathbb{E}_\theta \left[ \hat{\mu}^{\text{MLE}} \right] \right)^2 + \text{Var}_\theta \left( \hat{\mu}^{\text{MLE}} \right) + \text{Var}_\theta(Y) \\ &= \left( \kappa'(\theta) - \kappa'(\theta) \right)^2 + \frac{\varphi \kappa''(\theta)}{\sum_{i=1}^n v_i} + \frac{\varphi \kappa''(\theta)}{v} \\ &= \frac{(\kappa''(\theta))^2}{\mathcal{I}(\theta)} + \frac{\varphi \kappa''(\theta)}{v}, \end{aligned} \quad (4.6)$$

see (3.25) for Fisher's information  $\mathcal{I}(\theta)$ . In this calculation we have used that the MLE  $\hat{\mu}^{\text{MLE}}$  is UMVU for  $\mu = \kappa'(\theta)$  and that  $Y_n$  and  $Y$  come from the same EDF with the same canonical parameter  $\theta \in \overset{\circ}{\Theta}$ . As a result, we are only left with estimation variance and process variance, moreover, the estimation variance asymptotically vanishes as  $\sum_{i=1}^n v_i \rightarrow \infty$ . ■

### 4.1.2 Unit Deviances and Deviance Generalization Loss

The main estimation technique used in these notes is MLE introduced in Definition 3.4. At this stage, MLE is un-related to any specific scoring function  $L$  because it has been received by maximizing the log-likelihood function. In this section we discuss the deviance loss function (as a scoring function) and we highlight its connection to the Bregman divergence introduced in Sect. 2.3. Based on the deviance loss function choice we rephrase Theorem 4.1 in terms of this scoring function. A theoretical foundation to these considerations will be given in Sect. 4.1.3, below.

For the derivations in this section we rely on the same single-parameter linear EDF as in Example 4.3, having a steep cumulant function  $\kappa$ . The MLE of  $\mu = \kappa(\theta)$  is found by solving, see (4.5),

$$\hat{\mu}^{\text{MLE}} = \hat{\mu}^{\text{MLE}}(Y_n) = \arg \max_{\tilde{\mu} \in \overline{\mathcal{M}}} \sum_{i=1}^n \frac{Y_i h(\tilde{\mu}) - \kappa(h(\tilde{\mu}))}{\varphi/v_i} \in \overline{\mathcal{M}},$$

with canonical link  $h = (\kappa')^{-1}$ . This decision rule  $Y_n \mapsto \hat{\mu}^{\text{MLE}} = \hat{\mu}^{\text{MLE}}(Y_n)$  is now used to predict the (new) random variable  $Y$  and to estimate the unknown parameters  $\theta$  and  $\mu$ , respectively. We aim at studying the expected GL under a distribution-adapted loss function choice potentially different from the square loss function. Below we will justify this second choice more extensively.

For the *saturated model* the *common* canonical parameter  $\theta$  of the independent random variables  $Y_1, \dots, Y_n$  in (4.4) is replaced by *individual* canonical parameters  $\theta_i$ ,  $1 \leq i \leq n$ . These individual canonical parameters are estimated with individual MLEs. The individual MLEs are given by, respectively,

$$\hat{\theta}_i^{\text{MLE}} = (\kappa')^{-1}(Y_i) = h(Y_i) \quad \text{and} \quad \hat{\mu}_i^{\text{MLE}} = Y_i \in \overline{\mathcal{M}},$$

the latter always exists because of strict convexity and steepness of  $\kappa$ . Since the MLE  $\hat{\mu}_i^{\text{MLE}} = Y_i$  maximizes the log-likelihood, we receive for any  $\mu \in \mathcal{M}$  the inequality

$$\begin{aligned} 0 &\leq 2 \left( \log f(Y_i; h(Y_i), v_i/\varphi) - \log f(Y_i; h(\mu), v_i/\varphi) \right) \\ &= 2 \frac{v_i}{\varphi} \left( Y_i h(Y_i) - \kappa(h(Y_i)) - Y_i h(\mu) + \kappa(h(\mu)) \right) \\ &= \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \mu). \end{aligned} \quad (4.7)$$

The function  $(y, \mu) \mapsto \mathfrak{d}(y, \mu) \geq 0$  is the unit deviance introduced in (2.25), extended to  $\mathfrak{C}$ , and it is zero if and only if  $y = \mu$ , see Lemma 2.22. The latter is also an immediate consequence of the fact that the MLE is unique within EDFs.

*Remark 4.4* The unit deviance  $\mathfrak{d}(y, \mu)$  has only been considered on  $\mathfrak{C} \times \mathcal{M}$  in (2.25). Having steepness of cumulant function  $\kappa$  implies  $\mathfrak{C} = \mathcal{M}$ , see Theorem 2.19, and in the absolutely continuous EDF case, we always have  $Y_i \in \mathcal{M}$ , a.s., which makes (4.7) well-defined for all observations  $Y_i$ , a.s. In the discrete or the mixed EDF case, an observation  $Y_i$  can be at the boundary of  $\mathcal{M}$ . In that case (4.7) must be calculated from

$$\mathfrak{d}(Y_i, \mu) = 2 \left( \sup_{\tilde{\theta} \in \Theta} [Y_i \tilde{\theta} - \kappa(\tilde{\theta})] - Y_i h(\mu) + \kappa(h(\mu)) \right). \quad (4.8)$$

This applies, e.g., to the Poisson or Bernoulli cases for observation  $Y_i = 0$ , in these cases we obtain unit deviances  $2\mu$  and  $-2\log(1 - \mu)$ , respectively.

The previous considerations (4.7)–(4.8) have been studying one single observation  $Y_i$  of  $Y_n$ . Aggregating over all observations in  $Y_n$  (and additionally using independence between the individual components of  $Y_n$ ) we arrive at the so-called *deviance loss function*

$$\begin{aligned}
\mathfrak{D}(\mathbf{Y}_n, \mu) &\stackrel{\text{def.}}{=} \frac{1}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \mu) \\
&= \frac{2}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \left( Y_i h(Y_i) - \kappa(h(Y_i)) - Y_i h(\mu) + \kappa(h(\mu)) \right) \geq 0.
\end{aligned} \tag{4.9}$$

The deviance loss function  $\mathfrak{D}(\mathbf{Y}_n, \mu)$  subtracts twice the log-likelihood  $\ell_{\mathbf{Y}_n}(\mu)$  from the one of the saturated model. Thus, it introduces a sign flip compared to (4.5). This immediately gives us the following corollary.

**Corollary 4.5 (Deviance Loss Function)** *The MLE problem (4.5) is equivalent to solving*

$$\hat{\mu}^{MLE} = \arg \max_{\tilde{\mu} \in \mathcal{M}} \ell_{\mathbf{Y}_n}(\tilde{\mu}) = \arg \min_{\tilde{\mu} \in \mathcal{M}} \mathfrak{D}(\mathbf{Y}_n, \tilde{\mu}). \tag{4.10}$$

*Remarks 4.6*

- Formula (4.10) replaces a maximization problem by a minimization problem with objective function  $\mathfrak{D}(\mathbf{Y}_n, \mu)$  being bounded below by zero. We can use this deviance loss function as a loss function not only for parameter estimation, but also as a scoring function for analyzing GLs within the EDF (similarly to Theorem 4.1).
- We draw the link to the KL divergence discussed in Sect. 2.3. In formula (2.26) we have shown that the unit deviance is equal to the KL divergence (up to scaling with factor 2), thus, equivalently, MLE aims at minimizing the average KL divergence over all observations  $\mathbf{Y}_n$

$$\hat{\theta}^{MLE} = \arg \min_{\tilde{\theta} \in \Theta} \frac{1}{n} \sum_{i=1}^n D_{\text{KL}} \left( f(\cdot; h(Y_i), v_i/\varphi) \parallel f(\cdot; \tilde{\theta}, v_i/\varphi) \right),$$

by finding an optimal parameter  $\hat{\theta}^{\text{MLE}}$  somewhere ‘in the middle’ of the observation  $\hat{\theta}_1^{\text{MLE}} = h(Y_1), \dots, \hat{\theta}_n^{\text{MLE}} = h(Y_n)$ . This then provides us with, see (2.27),

$$\begin{aligned} \prod_{i=1}^n f(Y_i; \tilde{\theta}, v_i/\varphi) &= \left[ \prod_{i=1}^n f(Y_i; h(Y_i), v_i/\varphi) \right] e^{-\frac{1}{2} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \kappa'(\tilde{\theta}))} \quad (4.11) \\ &\propto \exp \left\{ - \sum_{i=1}^n D_{\text{KL}} \left( f(\cdot; h(Y_i), v_i/\varphi) \parallel f(\cdot; \tilde{\theta}, v_i/\varphi) \right) \right\}, \end{aligned}$$

where  $\propto$  highlights that we drop all terms that do not involve  $\tilde{\theta}$ . This describes the change in joint likelihood by varying the canonical parameter  $\tilde{\theta}$  over its domain  $\Theta$ . The first line of (4.11) is in the spirit of minimizing a weighted square loss, but the Gaussian square is replaced by the unit deviance  $\mathfrak{d}$ . The second line of (4.11) is in the spirit of information geometry considered in Sect. 2.3, where we try to find a canonical parameter  $\tilde{\theta}$  that has a small KL divergence to the  $n$  individual models being parametrized by  $h(Y_1), \dots, h(Y_n)$ , thus, the MLE  $\hat{\theta}^{\text{MLE}}$  provides an optimal balance over the entire set of (independent) observations  $Y_1, \dots, Y_n$  w.r.t. the KL divergence.

- In contrast to the square loss function, the deviance loss function  $\mathfrak{D}(Y_n, \mu)$  respects the distributional properties of  $Y_n$ , see (4.11). That is, if the underlying distribution allows for larger or smaller claims, this fact is appropriately valued in the deviance loss function (supposed that we have chosen the right family of distributions; model uncertainty will be studied in Sect. 11.1, below).
- Assume we work in the Gaussian model. In this model we have  $\kappa(\theta) = \theta^2/2$  and canonical link  $h(\mu) = \mu$ , see Sect. 2.1.3. This provides unit deviance in the Gaussian case  $\mathfrak{d}(y, \mu) = (y - \mu)^2$ , which is exactly the square loss function for action space  $\mathbb{A} = \overline{\mathcal{M}}$ . Thus, the square loss function is most appropriate in the Gaussian case.
- As explained above, we use unit deviances  $\mathfrak{d}(y, \mu)$  as a measure of discrepancy. Alternatively, as in the introduction to this section, see (4.6), we can consider Pearson’s  $\chi^2$ -statistic which corresponds to the weighted square loss function

$$X^2(y, \mu) = \frac{(y - \mu)^2}{V(\mu)}, \quad (4.12)$$

where  $\mu \mapsto V(\mu)$  is the variance function of the chosen EDF. Similarly, to the deviance loss function (4.9), we can aggregate these Pearson’s  $\chi^2$ -statistics  $X^2(Y_i, \mu)$  over all observations  $Y_i$  in  $Y_n$  to receive a second overall measure of discrepancy. In the Gaussian case the deviance loss and Pearson’s  $\chi^2$ -statistic coincide and have a  $\chi^2$ -distribution, for other distributions asymptotic results are available.

In the non-Gaussian case, (4.12) is not always robust. For instance, if we work in the Poisson model, we have variance function  $V(\mu) = \mu$ . Our examples

below will have low claim frequencies which implies that  $\mu$  will be small. The appearance of a small  $\mu$  in the denominator of (4.12) will imply that Pearson's  $\chi^2$ -statistic is not very robust in small frequency applications, in particular, if we need to estimate this  $\mu$  from  $Y_n$ . Therefore, we refrain from using (4.12).

Naturally, in analogy to Theorem 4.1 and derivation (4.6), the above considerations motivate us to consider expected GLs under unit deviances within the EDF. We use the decision rule  $\hat{\mu}^{\text{MLE}}(Y_n) \in \mathbb{A} = \overline{\mathcal{M}}$  to predict a new observation  $Y$ .

The expected *deviance GL* is defined and given by

$$\begin{aligned} & \mathbb{E}_\theta \left[ \mathfrak{d} \left( Y, \hat{\mu}^{\text{MLE}}(Y_n) \right) \right] \\ &= \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)] + 2 \mathbb{E}_\theta \left[ Y h(\mu) - \kappa(h(\mu)) - Y h(\hat{\mu}^{\text{MLE}}(Y_n)) + \kappa(h(\hat{\mu}^{\text{MLE}}(Y_n))) \right] \\ &= \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)] + \mathcal{E} \left( \mu, \hat{\mu}^{\text{MLE}}(Y_n) \right), \end{aligned} \quad (4.13)$$

the last identity uses independence between  $Y_n$  and  $Y$ , and with *estimation risk function*

$$\mathcal{E} \left( \mu, \hat{\mu}^{\text{MLE}}(Y_n) \right) = \mathbb{E}_\theta \left[ \mathfrak{d} \left( \mu, \hat{\mu}^{\text{MLE}}(Y_n) \right) \right] > 0, \quad (4.14)$$

we use steepness of the cumulant function,  $\mathfrak{C} = \overline{\text{conv}}(\mathfrak{T}) = \overline{\mathcal{M}}$ , and Lemma 2.22 for the strict positivity of the estimation risk function. Thus, for the estimation risk function  $\mathcal{E}$  we replace  $Y$  by  $\mu$  in the unit deviance and the expectation  $\mathbb{E}_\theta$  is only over the observations  $Y_n$ . This looks like a very convincing generalization of the MSE, however, one needs to ensure that all terms in (4.13) exist.

**Theorem 4.7 (Expected Deviance Generalization Loss)** *Assume that  $Y_n$  and  $Y$  are independent and belong to the same linear EDF having the same canonical parameter  $\theta \in \Theta$  and having strictly convex and steep cumulant function  $\kappa$ . Choose a predictor  $A : \mathbb{Y} \rightarrow \mathbb{A} = \overline{\mathcal{M}}$ ,  $Y_n \mapsto A(Y_n)$  and assume that all expectations in the following formula exist. The expected deviance GL of predictor  $A$  to predict  $Y$  is given by*

$$\mathbb{E}_\theta [\mathfrak{d}(Y, A(Y_n))] = \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)] + \mathcal{E}(\mu, A(Y_n)) \geq \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)].$$

*Remarks 4.8*

- $\mathbb{E}_\theta[\mathfrak{d}(Y, \mu)]$  plays the role of the pure process variance (irreducible risk) of Theorem 4.1. This term does not involve any parameter estimation bias and uncertainty because it is based on the true parameter  $\theta$  and  $\mu = \kappa'(\theta)$ , respectively. In Sect. 4.1.3, below, we are going to justify the appropriateness of this object as a tool for forecast evaluation. In particular, because the unit deviance is strictly consistent for the mean functional, the true mean  $\mu = \mu(\theta)$  minimizes  $\mathbb{E}_\theta[\mathfrak{d}(Y, \mu)]$ , see (4.28), below.
- The second term  $\mathcal{E}(\mu, A(Y_n))$  measures parameter estimation bias and uncertainty of decision rule  $A(Y_n)$  versus the true parameter  $\mu = \kappa'(\theta)$ . The first remark is that we can do this for any decision rule  $A$ , i.e., we do not necessarily need to consider the MLE. The second remark is that we can no longer get a clear cut differentiation between a bias term and a parameter estimation uncertainty term for deviance loss functions not coming from the Gaussian distribution. We come back to this in Remarks 7.17, below, where we give more characterization to the individual terms of the expected deviance GL.
- An issue in applying Theorem 4.7 to the MLE decision rule  $A(Y_n) = \hat{\mu}^{\text{MLE}}(Y_n)$  is that, in general, it does not lead to a finite estimation risk function. For instance, in the Poisson case we have with positive probability  $\hat{\mu}^{\text{MLE}}(Y_n) = 0$ , which results in an infinite estimation risk. In order to avoid this, we need to bound away the decision rule from the boundary of  $\mathcal{M}$  and  $\Theta$ , respectively. In the Poisson case this can be achieved by considering a decision rule  $A(Y_n) = \max\{\hat{\mu}^{\text{MLE}}(Y_n), \epsilon\}$  for a fixed given  $\epsilon \in (0, \mu = \kappa'(\theta))$ . This decision rule has a bias which asymptotically vanishes as  $n \rightarrow \infty$ . Moreover, consistency and asymptotic normality tells us that this lower bound does not affect prediction for large sample sizes  $n$  (with large probability).
- Similar to (4.3), we can also consider the deviance GL, given  $Y_n$ . Under independence of  $Y_n$  and  $Y$  we have deviance GL

$$\begin{aligned} \mathbb{E}_\theta[\mathfrak{d}(Y, A(Y_n)) | Y_n] &= \mathbb{E}_\theta[\mathfrak{d}(Y, \mu) | Y_n] + \mathfrak{d}(\mu, A(Y_n)) \quad (4.15) \\ &\geq \mathbb{E}_\theta[\mathfrak{d}(Y, \mu)]. \end{aligned}$$

Thus, here we directly compare  $A(Y_n)$  to the true parameter  $\mu$ .

*Example 4.9 (Estimation Risk Function in the Gaussian Case)* We consider the Gaussian case with cumulant function  $\kappa(\theta) = \theta^2/2$  and canonical link  $h(\mu) = \mu$ .

The estimation risk function is in the Gaussian case for a square integrable predictor  $A(\mathbf{Y}_n)$  given by

$$\begin{aligned}\mathcal{E}(\mu, A(\mathbf{Y}_n)) &= \mathbb{E}_\theta [\mathfrak{D}(\mu, A(\mathbf{Y}_n))] \\ &= 2\left(\mu h(\mu) - \kappa(h(\mu)) - \mu \mathbb{E}_\theta [h(A(\mathbf{Y}_n))] + \mathbb{E}_\theta [\kappa(h(A(\mathbf{Y}_n)))]\right) \\ &= \mu^2 - 2\mu \mathbb{E}_\theta [A(\mathbf{Y}_n)] + \mathbb{E}_\theta [(A(\mathbf{Y}_n))^2] \\ &= (\mu - \mathbb{E}_\theta [A(\mathbf{Y}_n)])^2 + \text{Var}_\theta(A(\mathbf{Y}_n)).\end{aligned}$$

These are exactly the squared bias and the estimation variance, see (4.1). Thus, in the Gaussian case, the MSE and the expected deviance GL coincide. Moreover, adding a deterministic bias  $c \in \mathbb{R}$  to  $A(\mathbf{Y}_n)$  increases the estimation risk function, supposed that  $A(\mathbf{Y}_n)$  is unbiased for  $\mu$ . We emphasize the latter as this is an important property to have, and we refer to the next Example 4.10 for an example where this property fails to hold. ■

*Example 4.10 (Estimation Risk Function in the Poisson Case)* We consider the Poisson case with cumulant function  $\kappa(\theta) = e^\theta$  and canonical link  $h(\mu) = \log \mu$ . The estimation risk function is given by (subject to existence)

$$\mathcal{E}(\mu, A(\mathbf{Y}_n)) = 2\left(\mu \log(\mu) - \mu - \mu \mathbb{E}_\theta [\log(A(\mathbf{Y}_n))] + \mathbb{E}_\theta [A(\mathbf{Y}_n)]\right). \quad (4.16)$$

Assume that decision rule  $A(\mathbf{Y}_n)$  is non-deterministic and unbiased for  $\mu$ . Using Jensen's inequality these assumptions imply for the estimation risk function

$$\mathcal{E}(\mu, A(\mathbf{Y}_n)) = 2\mu\left(\log(\mu) - \mathbb{E}_\theta [\log(A(\mathbf{Y}_n))]\right) > 0.$$

We now add a small deterministic bias  $c \in \mathbb{R}$  to the unbiased estimator  $A(\mathbf{Y}_n)$  for  $\mu$ . This gives us estimation risk function, see (4.16) and subject to existence,

$$\mathcal{E}(\mu, A(\mathbf{Y}_n) + c) = 2\left(\mu \log(\mu) - \mu \mathbb{E}_\theta [\log(A(\mathbf{Y}_n) + c)] + c\right).$$

Consider the derivative w.r.t. bias  $c$  in 0, we use Jensen's inequality on the last line,

$$\begin{aligned}\left.\frac{\partial}{\partial c} \mathcal{E}(\mu, A(\mathbf{Y}_n) + c)\right|_{c=0} &= 2\left(-\mu \mathbb{E}_\theta \left[\frac{1}{A(\mathbf{Y}_n) + c}\right] + 1\right)\bigg|_{c=0} \\ &= -2\mu \mathbb{E}_\theta \left[\frac{1}{A(\mathbf{Y}_n)}\right] + 2 \\ &< -2\mu \frac{1}{\mathbb{E}_\theta [A(\mathbf{Y}_n)]} + 2 = 0.\end{aligned} \quad (4.17)$$



Thus, the estimation risk becomes smaller if we add a small bias to the (non-deterministic) unbiased predictor  $A(Y_n)$ . This issue has been raised in Denuit et al. [97]. Of course, this is a very unfavorable property, and it is rather different from the Gaussian case in Example 4.9. It is essentially driven by the fact that parameter estimation is based on a finite sample, which implies a strict inequality in (4.17) for the finite sample estimate  $A(Y_n)$ . A conclusion of this example is that if we use expected deviance GLs for forecast evaluation we need to insist on having unbiased predictors. This will become especially important for more complex regression models, see Sect. 7.4.2, below.

More generally, one can prove this result of a smaller estimation risk function for a small positive bias for any EDF member with power variance function  $V(\mu) = \mu^p$  with  $p \geq 1$ , see also (4.18) below. The proof uses the Fortuin–Kasteleyn–Ginibre (FKG) inequality [133] providing  $\mathbb{E}_\theta[A(Y_n)^{1-p}] < \mathbb{E}_\theta[A(Y_n)]\mathbb{E}_\theta[A(Y_n)^{-p}] = \mu\mathbb{E}_\theta[A(Y_n)^{-p}]$  to receive (4.17) for power variance parameters  $p \geq 1$ . ■

*Remarks 4.11 (Conclusion from Examples 4.9 and 4.10 and a Further Remark)*

- Working with expected deviance GLs for evaluating forecasts requires some care because a bigger bias in the (finite sample) estimate  $A(Y_n)$  may provide a smaller estimation risk function  $\mathcal{E}(\mu, A(Y_n))$ . For this reason, we typically insist on having unbiased predictors/forecasts. The latter is also an important requirement in financial applications to guarantee that the overall price is set to the right level, we refer to the balance property in Corollary 3.19 and to Sect. 7.4.2, below.
- In Theorems 4.1 and 4.7 we use independence between the predictor  $A(Y_n)$  and the random variable  $Y$  to receive the split of the expected deviance GL into irreducible risk and estimation risk function. In regression models, this independence between the predictor  $A(Y_n)$  and the random variable  $Y$  may no longer hold. In that case we will still work with the expected deviance GL  $\mathbb{E}_\theta[\mathfrak{d}(Y, A(Y_n))]$ , but a clear split between estimation and forecasting will no longer be possible, see Sect. 4.2, below.

The next example gives the most important unit deviances in actuarial modeling.

*Example 4.12 (Unit Deviances)* We give the most prominent examples of unit deviances within the single-parameter linear EDF. We recall unit deviance (2.25)

$$\mathfrak{d}(y, \mu) = 2 \left( yh(y) - \kappa(h(y)) - yh(\mu) + \kappa(h(\mu)) \right) \geq 0.$$

In Sect. 2.2 we have met the examples given in Table 4.1.

**Table 4.1** Unit deviances of selected distributions commonly used in actuarial science

Distribution	Cumulant function $\kappa(\theta)$	Unit deviance $\mathfrak{d}(y, \mu)$
Gaussian	$\theta^2/2$	$(y - \mu)^2$
Gamma	$-\log(-\theta)$	$2((y - \mu)/\mu + \log(\mu/y))$
Inverse Gaussian	$-\sqrt{-2\theta}$	$(y - \mu)^2/(\mu^2 y)$
Poisson	$e^\theta$	$2(\mu - y - y\log(\mu/y))$
Negative-binomial	$-\log(1 - e^\theta)$	$2\left(y\log\left(\frac{y}{\mu}\right) - (y + 1)\log\left(\frac{y+1}{\mu+1}\right)\right)$
Tweedie's CP	$\frac{((1-p)\theta)^{\frac{2-p}{1-p}}}{2-p}, \quad p \in (1, 2)$	$2\left(y\frac{y^{1-p}-\mu^{1-p}}{1-p} - \frac{y^{2-p}-\mu^{2-p}}{2-p}\right)$
Bernoulli	$\log(1 + e^\theta)$	$2(-y\log\mu - (1 - y)\log(1 - \mu))$

If we focus on Tweedie's distributions having power variance functions  $V(\mu) = \mu^p$ , see Table 2.1, we get a unified expression for the unit deviances for  $p \in \{0\} \cup (1, 2) \cup (2, \infty)$

$$\begin{aligned} \mathfrak{d}(y, \mu) &= 2\left(y\frac{y^{1-p}-\mu^{1-p}}{1-p} - \frac{y^{2-p}-\mu^{2-p}}{2-p}\right) \\ &= 2\left(\frac{y^{2-p}}{(1-p)(2-p)} - \frac{y\mu^{1-p}}{1-p} + \frac{\mu^{2-p}}{2-p}\right). \end{aligned} \quad (4.18)$$

For the remaining power variance cases we have:  $p = 1$  corresponds to the Poisson case,  $p = 2$  gives the gamma case, the cases  $p < 0$  do not have a steep cumulant function, and, moreover, there are no EDF models for  $p \in (0, 1)$ , see Theorem 2.18.

The unit deviance in the Bernoulli case is also called *binary cross-entropy*. This binary cross-entropy has a categorical generalization, called *multi-class cross-entropy*. Assume we have a categorical EF with levels  $\{1, \dots, k+1\}$  and corresponding probabilities  $p_1, \dots, p_{k+1} \in (0, 1)$  summing up to 1, see Sect. 2.1.4. We denote by  $\mathbf{Y} = (\mathbb{1}_{\{Y=1\}}, \dots, \mathbb{1}_{\{Y=k+1\}})^\top \in \mathbb{R}^{k+1}$  the indicator variable that shows which level the categorical random variable  $Y$  takes;  $\mathbf{Y}$  is called one-hot encoding of the categorical random variable  $Y$ . Assume  $\mathbf{y}$  is a realization of  $\mathbf{Y}$  and set  $\boldsymbol{\mu} = \mathbf{p} = (p_1, \dots, p_{k+1})^\top$ . The categorical (multi-class) cross-entropy loss function is given by

$$\mathfrak{d}(\mathbf{y}, \boldsymbol{\mu}) = \mathfrak{d}(\mathbf{y}, \mathbf{p}) = -2 \sum_{j=1}^{k+1} y_j \log p_j \geq 0. \quad (4.19)$$

This cross-entropy is closely related to the KL divergence between two categorical distributions  $\mathbf{p}$  and  $\mathbf{q}$  on  $\{1, \dots, k+1\}$ . The KL divergence from  $\mathbf{p}$  to  $\mathbf{q}$  is given by

$$D_{\text{KL}}(\mathbf{q} \parallel \mathbf{p}) = \sum_{j=1}^{k+1} q_j \log \left( \frac{q_j}{p_j} \right) = \sum_{j=1}^{k+1} q_j \log q_j - \sum_{j=1}^{k+1} q_j \log p_j.$$

If we replace the true (but unknown) distribution  $\mathbf{q}$  by observation  $\mathbf{Y} = \mathbf{y}$  we receive unit deviance (4.19) (scaled by 2), and the MLE is obtained by minimizing this KL divergence, see also Example 3.10. ■

**Outlook 4.13** In the regression modeling, below, each response  $Y_i$  will have its own mean parameter  $\mu_i = \mu(\boldsymbol{\beta}, \mathbf{x}_i)$  which will be a function of its covariate information  $\mathbf{x}_i$ , and  $\boldsymbol{\beta}$  denotes a regression parameter to be estimated with MLE. In that case, we modify the deviance loss function (4.9) to

$$\boldsymbol{\beta} \mapsto \mathfrak{D}(\mathbf{Y}_n, \boldsymbol{\beta}) = \frac{1}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \mu_i) = \frac{1}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \mu(\boldsymbol{\beta}, \mathbf{x}_i)), \quad (4.20)$$

and the MLE of  $\boldsymbol{\beta}$  can be found by solving

$$\hat{\boldsymbol{\beta}}^{\text{MLE}} = \arg \min_{\boldsymbol{\beta}} \mathfrak{D}(\mathbf{Y}_n, \boldsymbol{\beta}). \quad (4.21)$$

If  $Y$  is a new response with covariate information  $\mathbf{x}$  and following the same EDF as  $\mathbf{Y}_n$ , we will evaluate the corresponding expected scaled deviance GL given by

$$\mathbb{E}_{\boldsymbol{\beta}} \left[ \frac{v}{\varphi} \mathfrak{d}(Y, \mu(\hat{\boldsymbol{\beta}}^{\text{MLE}}, \mathbf{x})) \right], \quad (4.22)$$

where  $\mathbb{E}_{\boldsymbol{\beta}}$  is the expectation under the true regression parameter  $\boldsymbol{\beta}$  for  $\mathbf{Y}_n$  and  $Y$ . This will be discussed in Sect. 5.1.7, below. If we interpret  $(Y, \mathbf{x}, v)$  as a random vector describing a randomly selected insurance policy from our portfolio, and being independent of  $\mathbf{Y}_n$  (and the corresponding covariate information  $\mathbf{x}_i$ ,  $1 \leq i \leq n$ ), then  $\hat{\boldsymbol{\beta}}^{\text{MLE}}$  will be independent of  $(Y, \mathbf{x}, v)$ . Nevertheless, the predictor  $\mu(\hat{\boldsymbol{\beta}}^{\text{MLE}}, \mathbf{x})$  will introduce dependence between the chosen decision rule and  $Y$  through  $\mathbf{x}$ , and we no longer receive the split of the expected deviance GL as stated in Theorem 4.7, for a related discussion we also refer to Remarks 7.17, below.

If we interpret  $(Y, \mathbf{x}, v)$  as a randomly selected insurance policy, then the expected GL (4.22) is evaluated under the joint (portfolio) distribution of  $(Y, \mathbf{x}, v)$ , and the deviance loss  $\mathfrak{D}(\mathbf{Y}_n, \hat{\boldsymbol{\beta}}^{\text{MLE}})$  is an (in-sample) empirical version of (4.22). ■

### 4.1.3 A Decision-Theoretic Approach to Forecast Evaluation

We present an excursion to a decision-theoretic approach to forecast evaluation. This excursion gives the theoretical foundation to the unit deviance considerations from above. This section follows Gneiting [162], Krüger–Ziegel [227] and Denuit et al. [97], and we refrain from giving complete proofs in this section. Forecast evaluation should involve consistent loss/scoring functions and proper scoring rules

to encourage the forecaster to make careful assessments and honest forecasts. Consistent loss functions are also a necessary tool to receive consistency of M-estimators, we refer to Remarks 3.26.

### Consistency and Proper Scoring Rules

Denote by  $\mathcal{C} \subseteq \mathbb{R}$  the convex closure of the support of a real-valued random variable  $Y$ , and let the action space be  $\mathbb{A} = \mathcal{C}$ , see also (3.1). Predictions are evaluated in terms of a loss/scoring function

$$L : \mathcal{C} \times \mathbb{A} \rightarrow \mathbb{R}_+, \quad (y, a) \mapsto L(y, a) \geq 0. \quad (4.23)$$

*Remark 4.14* In (4.23) we assume that the loss function  $L$  is bounded below by zero. This can be an advantage in applications because it gives a calibration to the loss function. In general, this lower bound is not a necessary condition for forecast evaluation. If we drop this lower bound property, we rather call  $L$  (only) a scoring function. For instance, the log-likelihood  $\log(f(y, a))$  in (3.27) plays the role of a scoring function.

The forecaster can take the position of minimizing the expected loss to choose her/his action rule. That is, subject to existence, an optimal action w.r.t.  $L$  is received by

$$\hat{a} = \hat{a}(F) = \arg \min_{a \in \mathbb{A}} \mathbb{E}_F [L(Y, a)] = \arg \min_{a \in \mathbb{A}} \int_{\mathcal{C}} L(y, a) dF(y). \quad (4.24)$$

In this setup the scoring function  $L(y, a)$  describes the loss that the forecaster suffers if she/he uses action  $a \in \mathbb{A}$  and observation  $y \in \mathcal{C}$  materializes. Since we do not want to insist on uniqueness in (4.24) we rather think of set-valued functionals in this section, which may provide solutions to problems like (4.24).<sup>1</sup>

We now reverse the line of arguments, and we start from a general set-valued functional. Denote by  $\mathcal{F}$  the family of distribution functions of interest supported on  $\mathcal{C}$ . Consider the set-valued functional

$$\mathfrak{A} : \mathcal{F} \rightarrow \mathcal{P}(\mathbb{A}), \quad F \mapsto \mathfrak{A}(F) \subset \mathbb{A}, \quad (4.25)$$

that maps each distribution  $F \in \mathcal{F}$  to a subset  $\mathfrak{A}(F)$  of the action space  $\mathbb{A} = \mathcal{C}$ , that is, an element of the power set  $\mathcal{P}(\mathbb{A})$ . The main question that we want to study in this section is the following: can we find a loss function  $L$  so that the set-valued

---

<sup>1</sup> In fact, also for the MLE in Definition 3.4 we should consider a set-valued functional. We have decided to skip this distinction to avoid any kind of complication and to not disturb the flow of reading.

functional  $\mathfrak{A}$  is obtained by a loss minimization (4.24)? This motivates the following definition.

**Definition 4.15 (Strict Consistency)** The loss function  $L : \mathfrak{C} \times \mathbb{A} \rightarrow \mathbb{R}_+$  is consistent for the functional  $\mathfrak{A} : \mathcal{F} \rightarrow \mathcal{P}(\mathbb{A})$  relative to the class  $\mathcal{F}$  if

$$\mathbb{E}_F [L(Y, \hat{a})] \leq \mathbb{E}_F [L(Y, a)], \quad (4.26)$$

for all  $F \in \mathcal{F}$ ,  $\hat{a} \in \mathfrak{A}(F)$  and  $a \in \mathbb{A}$ . It is strictly consistent if it is consistent and equality in (4.26) implies that  $a \in \mathfrak{A}(F)$ .

As stated in Theorem 1 of Gneiting [162], a loss function  $L$  is consistent for the functional  $\mathfrak{A}$  relative to the class  $\mathcal{F}$  if and only if, given any  $F \in \mathcal{F}$ , every  $\hat{a} \in \mathfrak{A}(F)$  is an optimal action under  $L$  in the sense of (4.24).

We give an example. Assume we start from the functional  $F \mapsto \mathfrak{A}(F) = \mathbb{E}_F[Y]$  that maps each distribution  $F$  to its expected value. In this case we do not need to consider a set-valued functional because the expected value is a singleton (we assume that  $\mathcal{F}$  only contains distributions with a finite first moment). The question then is whether we can find a loss function  $L$  such that this mean can be received by a minimization (4.24). This question is answered in Theorem 4.19, below.

Next we relate a consistent loss function  $L$  to a *proper scoring rule*. A proper scoring rule is a function  $R : \mathfrak{C} \times \mathcal{F} \rightarrow \mathbb{R}$  such that

$$\mathbb{E}_F [R(Y, F)] \leq \mathbb{E}_F [R(Y, G)], \quad (4.27)$$

for all  $F, G \in \mathcal{F}$ , supposed that the expectations are well-defined. A scoring rule  $R$  analyzes the penalty  $R(y, G)$  if the forecaster works with a distribution  $G$  and an observation  $y$  of  $Y \sim F$  materializes. Proper scoring rules have been promoted in Gneiting–Raftery [163] and Gneiting [162]. They are important because they encourage the forecaster to make honest forecasts, i.e., it gives the forecaster the incentive to minimize the expected score by following his true belief about the true distribution, because only this minimizes the expected penalty in (4.27).

**Theorem 4.16 (Gneiting [162, Theorem 3])** Assume that  $L$  is a consistent loss function for the functional  $\mathfrak{A}$  relative to the class  $\mathcal{F}$ . For each  $F \in \mathcal{F}$ , let  $a_F \in \mathfrak{A}(F)$ . The scoring rule

$$R : \mathfrak{C} \times \mathcal{F} \rightarrow \mathbb{R}, \quad (y, F) \mapsto R(y, F) = L(y, a_F),$$

is a proper scoring rule.

**Example 4.17** Consider the unit deviance  $\mathfrak{d}(\cdot, \cdot) : \mathfrak{C} \times \mathcal{M} \rightarrow \mathbb{R}_+$  for a given EDF  $\mathcal{F} = \{F(\cdot; \theta, \nu/\varphi); \theta \in \Theta\}$  with cumulant function  $\kappa$ . Lemma 2.22 says that under suitable assumptions this unit deviance  $\mathfrak{d}(y, \mu)$  is zero if and only if  $y = \mu$ . We consider the mean functional on  $\mathcal{F}$

$$\mathfrak{A} : \mathcal{F} \rightarrow \mathbb{A} = \mathcal{M}, \quad F_\theta = F(\cdot; \theta, \nu/\varphi) \mapsto \mathfrak{A}(F_\theta) = \mu(\theta),$$

where  $\mu = \mu(\theta) = \kappa'(\theta)$  is the mean of the chosen EDF. Choosing the unit deviance as loss function we receive for any action  $a \in \mathbb{A}$ , see (4.13),

$$\begin{aligned}\mathbb{E}_\theta [\mathfrak{d}(Y, a)] &= \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)] + 2 \mathbb{E}_\theta [Yh(\mu) - \kappa(h(\mu)) - Yh(a) + \kappa(h(a))] \\ &= \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)] + 2 (\mu h(\mu) - \kappa(h(\mu)) - \mu h(a) + \kappa(h(a))) \\ &= \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)] + \mathfrak{d}(\mu, a).\end{aligned}$$

This is minimized for  $a = \mu$  and it proves that the unit deviance is strictly consistent for the mean functional  $\mathfrak{A} : F_\theta \mapsto \mathfrak{A}(F_\theta) = \mu(\theta)$  relative to the chosen EDF  $\mathcal{F} = \{F(\cdot; \theta, v/\varphi); \theta \in \Theta\}$ . Using Theorem 4.16, the scoring rule

$$R : \mathcal{C} \times \mathcal{F} \rightarrow \mathbb{R}, \quad (y, F_\theta) \mapsto R(y, F_\theta) = \mathfrak{d}(y, \mu(\theta)),$$

is a strictly proper scoring rule, that is,

$$\mathbb{E}_\theta [R(Y, F_\theta)] = \mathbb{E}_\theta [\mathfrak{d}(Y, \mu(\theta))] < \mathbb{E}_\theta [\mathfrak{d}(Y, \mu(\tilde{\theta}))] = \mathbb{E}_\theta [R(Y, F_{\tilde{\theta}})],$$

for any  $\tilde{\theta} \neq \theta$ . We conclude from this small example that the unit deviance is a strictly consistent loss function for the mean functional on the chosen EDF, and this provides us with a strictly proper scoring rule. ■

In the above Example 4.17 we have chosen the mean functional

$$\mathfrak{A} : \mathcal{F} \rightarrow \mathbb{A} = \mathcal{M}, \quad F_\theta = F(\cdot; \theta, v/\varphi) \mapsto \mathfrak{A}(F_\theta) = \mu(\theta),$$

within a given EDF  $\mathcal{F} = \{F(\cdot; \theta, v/\varphi); \theta \in \Theta\}$ . We have seen that

- the unit deviance  $\mathfrak{d}(\cdot, \cdot)$  is a strictly consistent loss function for the mean functional  $\mathfrak{A}$  relative to the EDF  $\mathcal{F}$ ;
- the function  $(y, F_\theta) \mapsto R(y, F_\theta) = \mathfrak{d}(y, \mu(\theta))$  is a strictly proper scoring rule for the EDF  $\mathcal{F}$ , i.e.,

$$\mathbb{E}_\theta [\mathfrak{d}(Y, \mu(\theta))] < \mathbb{E}_\theta [\mathfrak{d}(Y, \mu(\tilde{\theta}))],$$

for any  $\tilde{\theta} \neq \theta$ .

The consideration of the mean functional  $F \mapsto \mathfrak{A}(F) = \mathbb{E}_F[Y]$  in Example 4.17 is motivated by the fact that we typically forecast random variables by their means. However, more generally, we may ask the question for which functionals  $\mathfrak{A} : \mathcal{F} \rightarrow \mathcal{P}(\mathbb{A})$ , relative to a given set of distributions  $\mathcal{F}$ , there exists a loss function  $L$  that is strictly consistent.

**Definition 4.18 (Elicitable)** The functional  $\mathfrak{A}$  is elicitable relative to a given set of distributions  $\mathcal{F}$  if there exists a loss function  $L$  that is strictly consistent for  $\mathfrak{A}$  and  $\mathcal{F}$ .

Above we have seen that the mean functional is elicitable relative to the EDF using the unit deviance loss; expected values relative to  $\mathcal{F}$  with finite second moments are also elicitable using the square loss function. Savage [327] more generally identifies the Bregman divergences as being the only consistent scoring functions for the mean functional; recall that the unit deviance is a special case of a Bregman divergence, see (2.29). We are going to state the corresponding result.

For a general loss function  $L$  we make the following (standard) assumptions:

- (L0)  $L(y, a) \geq 0$  and we have an equality if and only if  $y = a$ ;
- (L1)  $L(y, a)$  is measurable in  $y$  and continuous in  $a$ ;
- (L2) the partial derivative  $\partial L(y, a)/\partial a$  exists and is continuous in  $a$  whenever  $a \neq y$ .

This then allows us to cite the following theorem.

**Theorem 4.19 (Gneiting [162, Theorem 7])** *Let  $\mathcal{F}$  be the class of distributions on an interval  $\mathfrak{C} \subseteq \mathbb{R}$  having finite first moments.*

- *Assume the loss function  $L : \mathfrak{C} \times \mathbb{A} \rightarrow \mathbb{R}$  satisfies (L0)–(L2) for interval  $\mathfrak{C} = \mathbb{A} \subseteq \mathbb{R}$ .  $L$  is consistent for the mean functional relative to the class  $\mathcal{F}$  of compactly supported distributions on  $\mathfrak{C}$  if and only if the loss function  $L$  is of Bregman divergence form*

$$D_\psi(y, a) = \psi(y) - \psi(a) - \psi'(a)(y - a),$$

*for a convex function  $\psi$  with (sub-)gradient  $\psi'$  on  $\mathfrak{C}$ .*

- *If  $\psi$  is strictly convex on  $\mathfrak{C}$ , then the Bregman divergence  $D_\psi$  is strictly consistent for the mean functional relative to the class  $\mathcal{F}$  on  $\mathfrak{C}$  for which both  $\mathbb{E}_F[Y]$  and  $\mathbb{E}_F[\psi(Y)]$  exist and are finite.*

Theorem 4.19 tells us that Bregman divergences are the only consistent loss functions for the mean functional (under some additional assumptions). Consider the specific choice  $\psi(a) = a^2/2$  which is a strictly convex function. For this choice, the Bregman divergence is the square loss function  $D_\psi(y, a) = (y - a)^2/2$ , which is strictly consistent for the mean functional relative to the class  $\mathcal{F} \subset L^2(\mathbb{P})$ . We remark that also quantiles are elicitable, the corresponding result is going to be stated in Theorem 5.33, below.

The second bullet point of Theorem 4.19 immediately implies that the unit deviance  $\mathfrak{d}(\cdot, \cdot)$  is a strictly consistent loss function for the mean functional within the chosen EDF, see also (2.29) and Example 4.17. In particular, for  $\theta \in \mathring{\Theta}$

$$\mu = \mu(\theta) = \arg \min_{a \in \mathcal{M}} \mathbb{E}_\theta [\mathfrak{d}(Y, a)]. \quad (4.28)$$

Explicit evaluation of (4.28) requires that the true distribution  $F_\theta$  of  $Y$  is known. Since, typically, this is not the case, we need to evaluate it empirically. Assume that the random variables  $Y_i$  are independent and  $F_\theta$  distributed, with  $F_\theta$  belonging to the fixed EDF providing the corresponding unit deviance  $\mathfrak{d}$ . Then, the objective function in (4.28) is approximated by, a.s.,

$$\mathfrak{D}(Y_n, a) = \frac{1}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, a) \rightarrow \mathbb{E}_\theta \left[ \frac{v}{\varphi} \mathfrak{d}(Y, a) \right] \quad \text{as } n \rightarrow \infty. \quad (4.29)$$

The convergence statement follows from the strong law of large numbers applied to the i.i.d. random variables  $(Y_i, v_i)$ ,  $i \geq 1$ , and supposed that the right-hand side of (4.29) exists. Thus, the deviance loss function (4.9) is an empirical version of the expected deviance loss function, and this approach is successful if we can exchange the ‘argmin’ operator of (4.28) and the limit  $n \rightarrow \infty$  in (4.29). This closes the circle and brings us back to the M-estimator considered in Remarks 3.26 and 3.29, and which also links forecast evaluation and M-estimation.

### Forecast Dominance

A consequence of Theorem 4.19 is that there are infinitely many strictly consistent loss functions for the mean functional, and, in principle, we could choose any of these for forecast evaluation. Choosing the unit deviance  $\mathfrak{d}$  that matches the distribution  $F_\theta$  of the observations  $Y_n$  and  $Y$ , respectively, gives us the MLE  $\hat{\mu}^{\text{MLE}}$ , and we have seen that the MLE  $\hat{\mu}^{\text{MLE}}$  is not only unbiased for  $\mu = \kappa'(\theta)$ , but it also meets the Cramér–Rao information bound. That is, it is UMVU within the data generating model reflected by the true unit deviance  $\mathfrak{d}$ . This provides us (in the finite sample case) with a natural candidate for  $\mathfrak{d}$  in (4.29) and, thus, a canonical proper scoring rule for (out-of-sample) forecast evaluation.

The previous statements have all been done under the assumption that there is no uncertainty about the underlying family of distribution functions that generates  $Y$  and  $Y_n$ , respectively. Uncertainty was limited to the true canonical parameter  $\theta$  and the true mean  $\mu(\theta)$ . This situation changes under model uncertainty. Krüger–Ziegel [227] study the question of having multiple strictly consistent loss functions in the situation where there is no natural candidate choice. Different choices may give different rankings to different (finite sample) predictors. Assume we have two predictors  $\hat{\mu}_1$  and  $\hat{\mu}_2$  for a random variable  $Y$ . Similarly to the definition of the expected deviance GL, we understand these predictors  $\hat{\mu}_1$  and  $\hat{\mu}_2$  as random variables, and we assume that all considered random variables have a finite first moment. Importantly, we do not assume independence between  $\hat{\mu}_1$ ,  $\hat{\mu}_2$  and  $Y$ , and in regression models we typically receive dependence between predictors  $\hat{\mu}$  and random variables  $Y$  through the features (covariates)  $\mathbf{x}$ , see also Outlook 4.13. Following Krüger–Ziegel [227] and Ehm et al. [119] we define *forecast dominance* as follows.



**Definition 4.20 (Forecast Dominance)** Predictor  $\hat{\mu}_1$  dominates predictor  $\hat{\mu}_2$  if

$$\mathbb{E}[D_\psi(Y, \hat{\mu}_1)] \leq \mathbb{E}[D_\psi(Y, \hat{\mu}_2)],$$

for all Bregman divergences  $D_\psi$  with (convex)  $\psi$  supported on  $\mathfrak{C}$ , the latter being the convex closure of the supports of  $Y$ ,  $\hat{\mu}_1$  and  $\hat{\mu}_2$ .

If we work with a fixed member of the EDF, e.g., the gamma distribution, then we typically study the corresponding expected deviance GL for forecast evaluation in one single model, see Theorem 4.7 and (4.29). This evaluation may involve model risk in the decision making process, and forecast dominance provides a robust selection criterion.

Krüger–Ziegel [227] build on Theorem 1b and Corollary 1b of Ehm et al. [119] to prove the following theorem (which prevents from considering all convex functions  $\psi$ ).

**Theorem 4.21 (Theorem 2.1 of Krüger–Ziegel [227])** *Predictor  $\hat{\mu}_1$  dominates predictor  $\hat{\mu}_2$  if and only if for all  $\tau \in \mathfrak{C}$*

$$\mathbb{E}[(Y - \tau) \mathbb{1}_{\{\hat{\mu}_1 > \tau\}}] \geq \mathbb{E}[(Y - \tau) \mathbb{1}_{\{\hat{\mu}_2 > \tau\}}]. \quad (4.30)$$

Denuit et al. [97] argue that in insurance one typically works with Tweedie's distributions having power variances  $V(\mu) = \mu^p$  with power variance parameters  $p \geq 1$ . This motivates the following weaker form of forecast dominance.

**Definition 4.22 (Tweedie's Forecast Dominance)** Predictor  $\hat{\mu}_1$  Tweedie-dominates predictor  $\hat{\mu}_2$  if

$$\mathbb{E}[\mathfrak{d}_p(Y, \hat{\mu}_1)] \leq \mathbb{E}[\mathfrak{d}_p(Y, \hat{\mu}_2)],$$

for all Tweedie's unit deviances  $\mathfrak{d}_p$  with power variance parameters  $p \geq 1$ , we refer to (4.18) for  $p \in (1, \infty) \setminus \{2\}$  and Table 4.1 for the Poisson and gamma cases  $p \in \{1, 2\}$ .

Recall that Tweedie's unit deviances  $\mathfrak{d}_p$  are a subclass of Bregman divergences, see (2.29). Define the following function for power variance parameters  $p \geq 1$

$$\gamma_p(\mu) = \begin{cases} \log \mu & \text{for } p = 2, \\ \frac{\mu^{2-p}}{2-p} & \text{otherwise.} \end{cases}$$

Denuit et al. [97] prove the following proposition.

**Proposition 4.23 (Proposition 4.1 of Denuit et al. [97])** *Predictor  $\hat{\mu}_1$  Tweedie-dominates predictor  $\hat{\mu}_2$  if*

$$\mathbb{E}[\gamma_p(\hat{\mu}_1)] \leq \mathbb{E}[\gamma_p(\hat{\mu}_2)] \quad \text{for all } p \geq 1,$$

and

$$\mathbb{E} \left[ Y \mathbb{1}_{\{\hat{\mu}_1 > \tau\}} \right] \geq \mathbb{E} \left[ Y \mathbb{1}_{\{\hat{\mu}_2 > \tau\}} \right] \quad \text{for all } \tau \in \mathfrak{C}.$$

Theorem 4.21 gives necessary and sufficient conditions to have forecast dominance, Proposition 4.23 gives sufficient conditions to have the weaker Tweedie's forecast dominance. In Theorem 7.15, below, we give another characterization of forecast dominance in terms of convex orders, under the additional assumption that the predictors are so-called auto-calibrated.

## 4.2 Cross-Validation

This section focuses on estimating the expected deviance GL (4.13) in cases where the canonical parameter  $\theta$  is not known. Of course, the same concepts apply to the MSE. In the remainder of this section we scale the unit deviances with  $v/\varphi$ , to bring them in line with the deviance loss (4.9).

### 4.2.1 In-Sample and Out-of-Sample Losses

The general aim in predictive modeling is to predict an unobserved random variable  $Y$  as good as possible based on past information  $\mathbf{Y}_n$ . Within the EDF, the predictive performance is then evaluated under an empirical version of the expected deviance GL

$$\mathbb{E}_\theta \left[ \frac{v}{\varphi} \mathfrak{d} \left( Y, A(\mathbf{Y}_n) \right) \right] = 2\mathbb{E}_\theta \left[ \frac{v}{\varphi} \left( Yh(Y) - \kappa(h(Y)) - Yh(A(\mathbf{Y}_n)) + \kappa(h(A(\mathbf{Y}_n))) \right) \right]. \quad (4.31)$$

Here, we no longer assume that  $Y$  and  $A(\mathbf{Y}_n)$  are independent, and in the dependent case Theorem 4.7 does not apply. The reason for dropping the independence assumption is that below we consider regression models of a similar type as in Outlook 4.13. The expected deviance GL (4.31) as such is not directly useful because it cannot be calculated if the true canonical parameter  $\theta$  is not known. Therefore, we are going to explain how it can be estimated empirically.

We start from the expected deviance GL in the EDF applied to the MLE decision rule  $\hat{\mu}^{\text{MLE}}(\mathbf{Y}_n)$ . It can be rewritten as

$$\mathbb{E}_\theta \left[ \frac{v}{\varphi} \mathfrak{d} \left( Y, \hat{\mu}^{\text{MLE}}(\mathbf{Y}_n) \right) \right] = \int \mathbb{E}_\theta \left[ \frac{v}{\varphi} \mathfrak{d} \left( Y, \hat{\mu}^{\text{MLE}}(\mathbf{Y}_n) \right) \middle| \mathbf{Y}_n = \mathbf{y}_n \right] dP(\mathbf{y}_n; \theta), \quad (4.32)$$

where we use the tower property for conditional expectations. In view of (4.32), there are two things to be done:

- (1) For given observations  $Y_n = y_n$ , we need to estimate the deviance GL, see also (4.15),

$$\mathbb{E}_\theta \left[ \frac{v}{\varphi} \mathfrak{d} \left( Y, \hat{\mu}^{\text{MLE}}(Y_n) \right) \middle| Y_n = y_n \right] = \mathbb{E}_\theta \left[ \frac{v}{\varphi} \mathfrak{d} \left( Y, \hat{\mu}^{\text{MLE}}(y_n) \right) \middle| Y_n = y_n \right]. \quad (4.33)$$

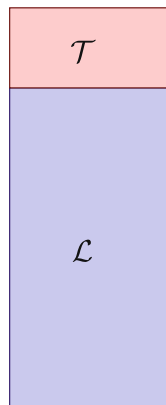
This is the part that we are going to solve empirically in this section. Typically, we assume that  $Y$  and  $Y_n$  are independent, nevertheless,  $Y$  and its MLE predictor may still be dependent because we may have a predictor  $\hat{\mu}^{\text{MLE}}(Y_n) = \hat{\mu}^{\text{MLE}}(Y_n, \mathbf{x})$ . That is, this predictor often depends on covariate information  $\mathbf{x}$  that describes  $Y$ , an example is provided in (4.22) of Outlook 4.13 and this is different from (4.15). In that case, the decision rule  $A : \mathbb{Y} \times \mathcal{X} \rightarrow \mathbb{A}$  is extended by an additional covariate component  $\mathbf{x} \in \mathcal{X}$ , we refer to Sect. 5.1.1, where  $\mathcal{X}$  is introduced and discussed.

- (2) We have to find a way to generate more observations  $Y_n$  from  $P(y_n; \theta)$  in order to evaluate the outer integral in (4.32) empirically. One way to do so is the bootstrap method that is going to be discussed in Sect. 4.3, below.

We address the first problem of estimating the deviance GL given in (4.33). We do this under the assumption that  $Y_n$  and  $Y$  are independent. In order to estimate (4.33) we need observations for  $Y$ . However, typically, there are no observations available for this random variable because it is only going to be observed in the future. For this reason, one uses past observations for both, model fitting and the GL analysis. In order to perform this analysis in a proper way, the general paradigm is to partition the entire data into two *disjoint* data sets, a so-called *learning data set*  $\mathcal{L} = \{Y_1, \dots, Y_n\}$  and a *test data set*  $\mathcal{T} = \{Y_1^\dagger, \dots, Y_T^\dagger\}$ . If we assume that all observations in  $\mathcal{L} \cup \mathcal{T}$  are independent, then we receive a suitable observation  $Y_n$  from the learning data set  $\mathcal{L}$  that can be used for model fitting. The test sample  $\mathcal{T}$  can then play the role of the unobserved random variable  $Y$  (by assumption being independent of  $Y_n$ ). Note that  $\mathcal{L}$  is *only* used for model fitting and  $\mathcal{T}$  is *only* used for the deviance GL evaluation, see Fig. 4.1.

This setup motivates to estimate the mean parameter  $\mu$  with MLE  $\hat{\mu}_{\mathcal{L}}^{\text{MLE}} = \hat{\mu}^{\text{MLE}}(Y_n)$  from the learning data  $\mathcal{L}$  and  $Y_n$ , respectively, by minimizing the deviance loss function  $\mu \mapsto \mathfrak{D}(Y_n, \mu)$  on the learning data  $\mathcal{L}$ , according to Corollary 4.5. Then we use this predictor  $\hat{\mu}_{\mathcal{L}}^{\text{MLE}}$  to empirically evaluate the conditional expectation in (4.33) on  $\mathcal{T}$ . The perception used is that we (*in-sample*) *learn a model* on  $\mathcal{L}$  and we *out-of-sample test this model* on  $\mathcal{T}$  to see how it generalizes to unobserved variables  $Y_t^\dagger$ ,  $1 \leq t \leq T$ , that are of a similar nature as  $Y$ .

**Fig. 4.1** Partition of entire data into learning data set  $\mathcal{L}$  and test data set  $\mathcal{T}$



**Definition 4.24 (In-Sample and Out-of-Sample Losses)** The *in-sample deviance loss* on the learning data  $\mathcal{L} = \{Y_1, \dots, Y_n\}$  is given by

$$\mathfrak{D}(\mathcal{L}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}}) = \frac{2}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \left( Y_i h(Y_i) - \kappa(h(Y_i)) - Y_i h(\hat{\mu}_{\mathcal{L}}^{\text{MLE}}) + \kappa(h(\hat{\mu}_{\mathcal{L}}^{\text{MLE}})) \right),$$

with MLE  $\hat{\mu}_{\mathcal{L}}^{\text{MLE}} = \hat{\mu}^{\text{MLE}}(\mathbf{Y}_n)$  on  $\mathcal{L}$ .

The out-of-sample deviance loss on the test data  $\mathcal{T} = \{Y_1^\dagger, \dots, Y_T^\dagger\}$  of predictor  $\hat{\mu}_{\mathcal{L}}^{\text{MLE}}$  is

$$\mathfrak{D}(\mathcal{T}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}}) = \frac{2}{T} \sum_{t=1}^T \frac{v_t^\dagger}{\varphi} \left( Y_t^\dagger h(Y_t^\dagger) - \kappa(h(Y_t^\dagger)) - Y_t^\dagger h(\hat{\mu}_{\mathcal{L}}^{\text{MLE}}) + \kappa(h(\hat{\mu}_{\mathcal{L}}^{\text{MLE}})) \right),$$

where the sum runs over the test sample  $\mathcal{T}$  having exposures  $v_1^\dagger, \dots, v_T^\dagger > 0$ .

For MLE we minimize the objective function (4.9), therefore, the in-sample deviance loss  $\mathfrak{D}(\mathcal{L}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}}) = \mathfrak{D}(\mathbf{Y}_n, \hat{\mu}^{\text{MLE}}(\mathbf{Y}_n))$  exactly corresponds to the minimal deviance loss (4.9) achieved on the learning data  $\mathcal{L}$ , i.e., when using MLE  $\hat{\mu}_{\mathcal{L}}^{\text{MLE}} = \hat{\mu}^{\text{MLE}}(\mathbf{Y}_n)$ . We call this *in-sample* because the *same* data  $\mathcal{L}$  is used for parameter estimation and deviance loss calculation. Typically, this loss is biased because it uses the optimal (in-sample) parameter estimate, we also refer to Sect. 4.2.3, below.

The out-of-sample loss  $\mathfrak{D}(\mathcal{T}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}})$  then empirically estimates the inner expectation in (4.32). This is a proper out-of-sample analysis because the test data  $\mathcal{T}$  is disjoint from the learning data  $\mathcal{L}$  on which the decision rule  $\hat{\mu}_{\mathcal{L}}^{\text{MLE}}$  has been trained. Note that this out-of-sample figure reflects (4.33) in the following sense.

We have a portfolio of risks  $(Y_t^\dagger, v_t^\dagger)$ ,  $1 \leq t \leq T$ , and (4.33) does not only reflect the calculation of the deviance GL of a given risk, but also the random selection of a risk from the portfolio. In this sense, (4.33) is an average over a given portfolio whose description is also included in the probability  $\mathbb{P}_\theta$ .

**Summary 4.25** Definition 4.24 gives the general principle in predictive modeling according to which model learning and the generalization analysis are done. Namely, based on two disjoint and independent data sets  $\mathcal{L}$  and  $\mathcal{T}$ , we perform model calibration on  $\mathcal{L}$ , and we analyze (conditional) GLs (using out-of-sample losses) on  $\mathcal{T}$ , respectively. For this concept to be useful, the learning data  $\mathcal{L}$  and the test data  $\mathcal{T}$  have to be sufficiently similar, i.e., ideally coming from the same model.

This approach does not estimate the outer expectation in the expected deviance GL (4.32), i.e., it is only an estimate for the deviance GL, given  $Y_n$ , see (4.33).

### 4.2.2 Cross-Validation Techniques

In many applications one is not in the comfortable situation of having two sufficiently large data sets  $\mathcal{L}$  and  $\mathcal{T}$  available to support model learning and an out-of-sample generalization analysis. That is, we are usually equipped with only one data set of average size, let us call it  $\mathcal{D}$ . In order to calculate the objects in Definition 4.24 we could partition this data set (at random) into two data sets and then calculate in-sample and out-of-sample deviance losses on this partition. The disadvantage of this approach is that it is an inefficient use of information if only little data is available. In that case we require (almost) all data for learning. However, we still need a sufficiently large share of data for testing, to receive reliable deviance GL estimates for (4.33). The classical approach in this situation is to use cross-validation for estimating out-of-sample losses. The concept works as follows:

1. Perform model learning and in-sample loss calculation  $\mathfrak{D}(\mathcal{L}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}})$  on all available data  $\mathcal{L} = \mathcal{D}$ , i.e., this part is not affected by selecting test data  $\mathcal{T}$  and it is not touched by cross-validation.
2. For out-of-sample deviance loss calculation use the data  $\mathcal{D}$  iteratively in an efficient way such that part of the data is used for model learning and the other part for the out-of-sample generalization analysis. This second step

(continued)

is (only) done for *estimating* the deviance GL of the model learned on all data. I.e. for prediction we work with MLE  $\hat{\mu}_{\mathcal{L}=\mathcal{D}}^{\text{MLE}}$ , but the out-of-sample deviance loss is estimated using this data in a different way.

The three most commonly used methods are leave-one-out,  $K$ -fold and stratified  $K$ -fold cross-validation. We briefly describe these three cross-validation methods.

### Leave-One-Out Cross-Validation

Denote all available data by  $\mathcal{D} = \{Y_1, \dots, Y_n\}$ , and assume independence between the components. For leave-one-out (loo) cross-validation we select  $1 \leq i \leq n$  and define the partition  $\mathcal{L}_{(-i)} = \mathcal{D} \setminus \{Y_i\}$  for the learning data and  $\mathcal{T}_i = \{Y_i\}$  for the test data. Based on the learning data  $\mathcal{L}_{(-i)}$  we calculate the MLE

$$\hat{\mu}^{(-i)} \stackrel{\text{def.}}{=} \hat{\mu}_{\mathcal{L}_{(-i)}}^{\text{MLE}},$$

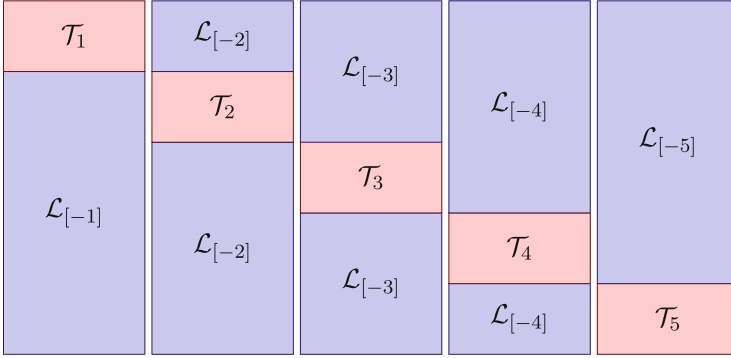
which is based on all data except observation  $Y_i$ . This observation is now used to do an out-of-sample analysis, and averaging this over all  $1 \leq i \leq n$  we receive the *leave-one-out cross-validation loss*

$$\begin{aligned} \widehat{\mathfrak{D}}^{\text{loo}} &= \frac{1}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \hat{\mu}^{(-i)}) = \frac{1}{n} \sum_{i=1}^n \mathfrak{D}(\mathcal{T}_i, \hat{\mu}^{(-i)}) \\ &= \frac{2}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \left( Y_i h(Y_i) - \kappa(h(Y_i)) - Y_i h(\hat{\mu}^{(-i)}) + \kappa(h(\hat{\mu}^{(-i)})) \right), \end{aligned} \quad (4.34)$$

where  $\mathfrak{D}(\mathcal{T}_i, \hat{\mu}^{(-i)})$  is the (out-of-sample) *cross-validation loss* on  $\mathcal{T}_i = \{Y_i\}$  using the predictor  $\hat{\mu}^{(-i)}$ . This leave-one-out cross-validation loss  $\widehat{\mathfrak{D}}^{\text{loo}}$  is now used as estimate for the out-of-sample deviance loss  $\mathfrak{D}(\mathcal{T}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}})$ . Leave-one-out cross-validation uses all data  $\mathcal{D}$  for learning and testing, namely, the data  $\mathcal{D}$  is partitioned into a learning set  $\mathcal{L}_{(-i)}$  for (partial) learning and a test set  $\mathcal{T}_i = \{Y_i\}$  for an out-of-sample generalization analysis. This is done for all instances  $1 \leq i \leq n$ , and the out-of-sample loss is estimated by the resulting average cross-validation loss. This averaging allows us to not only understand (4.34) as a conditional out-of-sample loss in the spirit of Definition 4.24. The outer empirical average in (4.34) also makes it suitable for an expected deviance GL estimate according to (4.32).

The variance of this empirical deviance GL is given by (subject to existence)

$$\text{Var}_{\theta}(\widehat{\mathfrak{D}}^{\text{loo}}) = \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n \text{Cov}_{\theta} \left( \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \hat{\mu}^{(-i)}), \frac{v_j}{\varphi} \mathfrak{d}(Y_j, \hat{\mu}^{(-j)}) \right).$$



**Fig. 4.2** Partitions of  $K$ -fold cross-validation for  $K = 5$

These covariances use exactly the same observations on  $\mathcal{D} \setminus \{Y_i, Y_j\}$ , therefore, there are strong correlations between the estimators  $\hat{\mu}^{(-i)}$  and  $\hat{\mu}^{(-j)}$ . In addition, the leave-one-out cross-validation is often computationally not feasible because it requires fitting the model  $n$  times, which in the situation of complex models and of large insurance portfolios can be too demanding. We come back to this in Sect. 5.6 where we provide the generalized cross-validation (GCV) loss approximation within generalized linear models (GLMs).

### **K-Fold Cross-Validation**

Choose a fixed integer  $K \geq 2$  and partition the entire data  $\mathcal{D}$  at random into  $K$  disjoint subsets (called folds)  $\mathcal{L}_1, \dots, \mathcal{L}_K$  of approximately the same size. The learning data for fixed  $1 \leq k \leq K$  is then defined by  $\mathcal{L}_{[-k]} = \mathcal{D} \setminus \mathcal{L}_k$  and the test data by  $\mathcal{T}_k = \mathcal{L}_k$ , see Fig. 4.2. Based on learning data  $\mathcal{L}_{[-k]}$  we calculate the MLE

$$\hat{\mu}^{[-k]} \stackrel{\text{def.}}{=} \hat{\mu}_{\mathcal{L}_{[-k}}}^{\text{MLE}},$$

which is based on all data except  $\mathcal{T}_k$ .

These observations are now used to do an (out-of-sample) cross-validation analysis, and averaging this over all  $1 \leq k \leq K$  we receive the *K-fold cross-validation (CV) loss*.

$$\begin{aligned}
\widehat{\mathfrak{D}}^{\text{CV}} &= \frac{1}{K} \sum_{k=1}^K \mathfrak{D} \left( \mathcal{T}_k, \widehat{\mu}^{[-k]} \right) \\
&= \frac{1}{K} \sum_{k=1}^K \frac{1}{|\mathcal{T}_k|} \sum_{Y_i \in \mathcal{T}_k} \frac{v_i}{\varphi} \mathfrak{d} \left( Y_i, \widehat{\mu}^{[-k]} \right) \\
&\approx \frac{1}{n} \sum_{k=1}^K \sum_{Y_i \in \mathcal{T}_k} \frac{v_i}{\varphi} \mathfrak{d} \left( Y_i, \widehat{\mu}^{[-k]} \right).
\end{aligned} \tag{4.35}$$

The last step is an approximation because not all  $\mathcal{T}_k$  may have exactly the same sample size if  $n$  is not a multiple of  $K$ . We can understand (4.35) not only as a conditional out-of-sample loss estimate in the spirit of Definition 4.24. The outer empirical average in (4.35) also makes it suitable for an expected deviance GL estimate according to (4.32). The variance of this empirical deviance GL is given by (subject to existence)

$$\text{Var}_{\theta} \left( \widehat{\mathfrak{D}}^{\text{CV}} \right) \approx \frac{1}{n^2} \sum_{k,l=1}^K \sum_{Y_i \in \mathcal{T}_k} \sum_{Y_j \in \mathcal{T}_l} \text{Cov}_{\theta} \left( \frac{v_i}{\varphi} \mathfrak{d} \left( Y_i, \widehat{\mu}^{[-k]} \right), \frac{v_j}{\varphi} \mathfrak{d} \left( Y_j, \widehat{\mu}^{[-l]} \right) \right).$$

Typically, in applications, one uses  $K$ -fold cross-validation with  $K = 10$ .

### Stratified $K$ -Fold Cross-Validation

A disadvantage of the above  $K$ -fold cross-validation is that it may happen that there are two outliers in the data, and there is a positive probability that these two outliers belong to the same subset  $\mathcal{L}_k$ . This may substantially distort  $K$ -fold cross-validation because in that case the subsets  $\mathcal{L}_k$ ,  $1 \leq k \leq K$ , are of different quality. Stratified  $K$ -fold cross-validation aims at distributing outliers more equally across the partition. Order the observations  $Y_i$ ,  $1 \leq i \leq n$ , as follows

$$Y_{(1)} \geq Y_{(2)} \geq \dots \geq Y_{(n)}.$$

For stratified  $K$ -fold cross-validation, we randomly distribute (partition) the  $K$  biggest claims  $Y_{(1)}, \dots, Y_{(K)}$  to the subsets  $\mathcal{L}_k$ ,  $1 \leq k \leq K$ , then we randomly partition the next  $K$  biggest claims  $Y_{(K+1)}, \dots, Y_{(2K)}$  to the subsets  $\mathcal{L}_k$ ,  $1 \leq k \leq K$ , and so forth. This implies, e.g., that the two biggest claims cannot fall into the same set  $\mathcal{L}_k$ . This stratified partition  $\mathcal{L}_k$ ,  $1 \leq k \leq K$ , is then used for  $K$ -fold cross-validation.



### Summary 4.26 (Cross-Validation)

- A model is calibrated on the learning data set  $\mathcal{L}$  by minimizing the in-sample deviance loss  $\mathfrak{D}(\mathcal{L}, \mu)$  in  $\mu$ . This provides MLE  $\hat{\mu}_{\mathcal{L}}^{\text{MLE}}$ .
- The quality of this model is assessed on test data  $\mathcal{T}$  being disjoint of  $\mathcal{L}$  considering the corresponding out-of-sample deviance loss  $\mathfrak{D}(\mathcal{T}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}})$ .
- If there is no test data set  $\mathcal{T}$  available we perform (stratified)  $K$ -fold cross-validation. This provides the (stratified)  $K$ -fold cross-validation loss  $\hat{\mathfrak{D}}^{\text{CV}}$  which is an estimate for the out-of-sample deviance loss and for the expected deviance GL (4.32).

*Example 4.27 (Out-of-Sample Deviance Loss Estimation)* We consider a claim counts example using the Poisson EDF model. The claim counts  $N_i$  and exposures  $v_i > 0$  used come from the French motor insurance data given in Listing 13.2 of Chap. 13.1. We model the claim frequencies  $Y_i = N_i/v_i$  with the Poisson EDF model having cumulant function  $\kappa(\theta) = \exp\{\theta\}$  and dispersion parameter  $\varphi = 1$  for all  $1 \leq i \leq n$ . The expected frequency is given by  $\mu = \mathbb{E}_{\theta}[Y_i] = \kappa'(\theta)$ . Moreover, we assume that all claim counts  $N_i$ ,  $1 \leq i \leq n$ , are independent. This provides us with the Poisson deviance loss function for observations  $\mathbf{Y}_n = (Y_1, \dots, Y_n)^{\top}$ , see Example 4.12,

$$\begin{aligned} \mathfrak{D}(\mathbf{Y}_n, \mu) &= \frac{1}{n} \sum_{i=1}^n v_i \mathfrak{d}(Y_i, \mu) = \frac{1}{n} \sum_{i=1}^n 2v_i \left( \mu - Y_i - Y_i \log \left( \frac{\mu}{Y_i} \right) \right) \\ &= \frac{1}{n} \sum_{i=1}^n 2 \left( v_i \mu - N_i - N_i \log \left( \frac{v_i \mu}{N_i} \right) \right) \geq 0, \end{aligned}$$

where, for  $Y_i = 0$ , we set  $\mathfrak{d}(Y_i = 0, \mu) = 2\mu$ . Minimizing the Poisson deviance loss function  $\mathfrak{D}(\mathbf{Y}_n, \mu)$  in  $\mu$  gives us the MLE for  $\mu$  and  $\theta = h(\mu)$ , respectively. It is given by, see (3.24),

$$\hat{\mu}^{\text{MLE}} = \hat{\mu}_{\mathcal{L}}^{\text{MLE}} = \frac{\sum_{i=1}^n N_i}{\sum_{i=1}^n v_i} = 7.36\%,$$

for learning data set  $\mathcal{L} = \{Y_1, \dots, Y_n\}$ . This provides us with an in-sample Poisson deviance loss of  $\mathfrak{D}(\mathbf{Y}_n, \hat{\mu}_{\mathcal{L}}^{\text{MLE}}) = \mathfrak{D}(\mathcal{L}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}}) = 25.213 \cdot 10^{-2}$ .

Since we do not have test data  $\mathcal{T}$ , we explore tenfold cross-validation. We therefore partition the entire data at random into  $K = 10$  disjoint sets  $\mathcal{L}_1, \dots, \mathcal{L}_{10}$ , and compute the tenfold cross-validation loss as described in (4.35). This gives us  $\hat{\mathfrak{D}}^{\text{CV}} = 25.213 \cdot 10^{-2}$ , thus, we receive the same value as for the in-sample loss which says that we do not have in-sample over-fitting, here. This is not surprising

in the homogeneous model  $\lambda = \mathbb{E}_\theta[Y_i]$ . We can also quantify the uncertainty in this estimate by the corresponding empirical standard deviation for  $\mathcal{T}_k = \mathcal{L}_k$

$$\sqrt{\frac{1}{K-1} \sum_{k=1}^K (\mathfrak{D}(\mathcal{T}_k, \hat{\mu}^{[-k]}) - \widehat{\mathfrak{D}}^{\text{CV}})^2} = 0.234 \cdot 10^{-2}. \quad (4.36)$$

This says that there is quite some fluctuation in the data because uncertainty in estimate  $\widehat{\mathfrak{D}}^{\text{CV}} = 25.213 \cdot 10^{-2}$  is roughly 1%. This finishes this example, and we will come back to it in Sect. 5.2.4, below. ■

### 4.2.3 Akaike's Information Criterion

The out-of-sample analysis in terms of GLs and cross-validation evaluates the predictive performance on unseen data. Another way of model selection is to study in-sample losses instead, but penalize model complexity. Akaike's information criterion (AIC), see Akaike [5], is the most popular tool that follows such a model selection methodology. AIC is based on a set of assumptions which should be fulfilled to apply, this is going to be discussed in this section; we therefore follow the lecture notes of Künsch [229].

Assume we have independent random variables  $Y_i$  from some (unknown) density  $f$ . Assume we have two candidate models with densities  $h_\theta$  and  $g_\vartheta$  from which we would like to select the preferred one for the given data  $\mathbf{Y}_n = (Y_1, \dots, Y_n)$ . The two unknown parameters in these densities  $h_\theta$  and  $g_\vartheta$  are called  $\theta$  and  $\vartheta$ , respectively. We neither assume that one of the two models  $h_\theta$  and  $g_\vartheta$  contains the true model  $f$ , nor that the two models are nested. That is,  $f$ ,  $h_\theta$  and  $g_\vartheta$  are quite general densities w.r.t. a given  $\sigma$ -finite measure  $\nu$ .

Assume that both models under consideration have a unique MLE  $\widehat{\theta}^{\text{MLE}} = \widehat{\theta}^{\text{MLE}}(\mathbf{Y}_n)$  and  $\widehat{\vartheta}^{\text{MLE}} = \widehat{\vartheta}^{\text{MLE}}(\mathbf{Y}_n)$  which is based on the same observations  $\mathbf{Y}_n$ . AIC [5] says that model  $h_{\widehat{\theta}^{\text{MLE}}}$  should be preferred over model  $g_{\widehat{\vartheta}^{\text{MLE}}}$  if

$$-2 \sum_{i=1}^n \log(h_{\widehat{\theta}^{\text{MLE}}}(Y_i)) + 2 \dim(\theta) < -2 \sum_{i=1}^n \log(g_{\widehat{\vartheta}^{\text{MLE}}}(Y_i)) + 2 \dim(\vartheta), \quad (4.37)$$

where  $\dim(\cdot)$  denotes the dimension of the corresponding parameter. Thus, we compute the log-likelihoods of the data  $\mathbf{Y}_n$  in the corresponding MLEs  $\widehat{\theta}^{\text{MLE}}$  and  $\widehat{\vartheta}^{\text{MLE}}$ , and we penalize the resulting values with the number of parameters to correct for model complexity. We give some remarks.

*Remarks 4.28*

- AIC is neither an in-sample loss nor an out-of-sample loss to measure generalization accuracy, but it considers penalized log-likelihoods. Under certain assumptions one can prove that asymptotically minimizing AICs is equivalent to minimizing leave-one-out cross-validation mean squared errors.
- The two penalized log-likelihoods have to be evaluated on the *same* data  $\mathbf{Y}_n$  and they need to consider the MLEs  $\hat{\theta}^{\text{MLE}}$  and  $\hat{\vartheta}^{\text{MLE}}$  because the justification of AIC is based on the asymptotic normality of MLEs, otherwise there is no mathematical justification why (4.37) should be a reasonable model selection tool.
- AIC does not require (but allows for) nested models  $h_\theta$  and  $g_\vartheta$  nor need they be Gaussian, it is only based on asymptotic normality. We give a heuristic argument below.
- Evaluation of (4.37) involves all terms of the log-likelihoods, also those that do not depend on the parameters  $\theta$  and  $\vartheta$ .
- Both models should consider the data  $\mathbf{Y}_n$  in the same units, i.e., AIC does not apply if  $h_\theta$  is a density for  $Y_i$  and  $g_\vartheta$  is a density for  $cY_i$ . In that case, one has to perform a transformation of variables to ensure that both densities consider the data in the same units. We briefly highlight this by considering a Gaussian example. We choose i.i.d. observations  $Y_i \sim \mathcal{N}(\theta, \sigma^2)$  for known variance  $\sigma^2 > 0$ . Choose  $c > 0$ , we have  $cY_i \sim \mathcal{N}(\vartheta = c\theta, c^2\sigma^2)$ . We obtain MLE  $\hat{\theta}^{\text{MLE}} = \sum_{i=1}^n Y_i / n$  and log-likelihood in MLE  $\hat{\theta}^{\text{MLE}}$

$$\sum_{i=1}^n \log(h_{\hat{\theta}^{\text{MLE}}}(Y_i)) = -\frac{n}{2} \log(2\pi\sigma^2) - \sum_{i=1}^n \frac{1}{2\sigma^2} (Y_i - \hat{\theta}^{\text{MLE}})^2.$$

On the transformed scale we have MLE  $\hat{\vartheta}^{\text{MLE}} = \sum_{i=1}^n cY_i / n = c\hat{\theta}^{\text{MLE}}$  and log-likelihood in MLE  $\hat{\vartheta}^{\text{MLE}}$

$$\sum_{i=1}^n \log(g_{\hat{\vartheta}^{\text{MLE}}}(cY_i)) = -\frac{n}{2} \log(2\pi c^2\sigma^2) - \sum_{i=1}^n \frac{1}{2c^2\sigma^2} (cY_i - \hat{\vartheta}^{\text{MLE}})^2.$$

Thus, find that the two log-likelihoods differ by  $-n\log(c)$ , but we consider the same model only under different measurement units of the data. The same applies when we work, e.g., with a log-normal model or logged data in a Gaussian model.

We give a heuristic justification of AIC. In Example 3.10 we have seen that the MLE is obtained by minimizing the KL divergence from  $h_\theta$  to the empirical distribution  $\hat{f}_n$  of  $\mathbf{Y}_n$ . This motivates to use the KL divergence also for comparing

the MLE estimated models to the true model, i.e., we consider the difference (supposed the densities are defined on the same domain)

$$\begin{aligned}
 & D_{\text{KL}}(f \| h_{\hat{\theta}^{\text{MLE}}}(\cdot)) - D_{\text{KL}}(f \| g_{\hat{\vartheta}^{\text{MLE}}}(\cdot)) \\
 &= \int \log\left(\frac{f(y)}{h_{\hat{\theta}^{\text{MLE}}}(y)}\right) f(y) d\nu(y) - \int \log\left(\frac{f(y)}{g_{\hat{\vartheta}^{\text{MLE}}}(y)}\right) f(y) d\nu(y) \\
 &= \int \log(g_{\hat{\vartheta}^{\text{MLE}}}(y)) f(y) d\nu(y) - \int \log(h_{\hat{\theta}^{\text{MLE}}}(y)) f(y) d\nu(y). \quad (4.38)
 \end{aligned}$$

If this difference is negative, model  $h_{\hat{\theta}^{\text{MLE}}}$  should be preferred over model  $g_{\hat{\vartheta}^{\text{MLE}}}$  because it is closer to the true model  $f$  w.r.t. the KL divergence. Thus, we need to calculate the two integrals in (4.38). Since the true density  $f$  is not known, these two integrals need to be estimated.

As a first idea we estimate the integrals on the right-hand side empirically using the observations  $Y_n$ , say, the first integral is estimated by

$$\frac{1}{n} \sum_{i=1}^n \log(g_{\hat{\vartheta}^{\text{MLE}}}(Y_i)).$$

However, this will lead to a biased estimate because the MLE  $\hat{\vartheta}^{\text{MLE}}$  exactly maximizes this empirical estimate (as a function of  $\vartheta$ ). The integrals in (4.38), on the other hand, can be interpreted as an out-of-sample calculation between independent random variables  $Y_n$  (used for MLE) and  $Y \sim f d\nu$  used in the integral. The bias results from the fact that in the empirical estimate the independence gets lost. Therefore, we need to correct this estimate for the bias in order to obtain a reasonable estimate for the difference of the KL divergences. Under the following assumptions this bias correction is asymptotically given by  $-\dim(\vartheta)/n$ : (1)  $\sqrt{n}(\hat{\vartheta}^{\text{MLE}}(Y_n) - \vartheta_0)$  is asymptotically normally distributed  $\mathcal{N}(0, \Sigma(\vartheta_0)^{-1})$  as  $n \rightarrow \infty$ , where  $\vartheta_0$  is the parameter that minimizes the KL divergence from  $g_{\vartheta}$  to  $f$ ; we also refer to Remarks 3.26. (2) The true  $f$  is sufficiently close to  $g_{\vartheta_0}$  such that the  $\mathbb{E}_f$ -covariance matrix of the score  $\nabla_{\vartheta} \log g_{\vartheta_0}$  is close to the negative  $\mathbb{E}_f$ -expected Hessian  $\nabla_{\vartheta}^2 \log g_{\vartheta_0}$ ; see also (3.36) and Sect. 11.1.4, below. In that case,  $\Sigma(\vartheta_0)$  approximately corresponds to Fisher's information matrix  $\mathcal{I}_1(\vartheta_0)$  and AIC is justified.

This shows that AIC applies if both models are evaluated under the same observations  $Y_n$ , the models need to use the MLEs, and asymptotic normality needs to hold with limits such that the true model is close to a member of the selected model classes  $\{h_{\theta}; \theta\}$  and  $\{g_{\vartheta}; \vartheta\}$ . We remark that this is not the only set-up under which AIC can be justified, but other set-ups do not essentially differ.

The Bayesian information criterion (BIC) is similar to AIC but in a Bayesian context. The BIC says that model  $h_{\hat{\theta}^{\text{MLE}}}$  should be preferred over model  $g_{\hat{\vartheta}^{\text{MLE}}}$  if

$$-2 \sum_{i=1}^n \log(h_{\hat{\theta}^{\text{MLE}}}(Y_i)) + \log(n) \dim(\theta) < -2 \sum_{i=1}^n \log(g_{\hat{\vartheta}^{\text{MLE}}}(Y_i)) + \log(n) \dim(\vartheta),$$

where  $n$  is the sample size of  $\mathbf{Y}_n$  used for model fitting. The BIC has been derived by Schwarz [331]. Therefore, it is also called Schwarz' information criterion (SIC).

### 4.3 Bootstrap

The bootstrap method has been invented by Efron [115] and Efron–Tibshirani [118]. The bootstrap is used to simulate new data from either the empirical distribution  $\hat{F}_n$  or from an estimated model  $F(\cdot; \hat{\theta})$ . This allows, for instance, to evaluate the outer expectation in the expected deviance GL (4.32) which requires a data model for  $\mathbf{Y}_n$ . The presentation in this section is based on the lecture notes of Bühlmann–Mächler [59, Chapter 5].

#### 4.3.1 Non-parametric Bootstrap Simulation

Assume we have i.i.d. observations  $Y_1, \dots, Y_n$  from an unknown distribution function  $F(\cdot; \theta)$ . Based on these observations  $\mathbf{Y} = (Y_1, \dots, Y_n)$  we choose a decision rule  $A : \mathbb{Y} \rightarrow \mathbb{A} = \Theta \subseteq \mathbb{R}$  which provides us with an estimator for  $\theta$

$$\mathbf{Y} \mapsto \hat{\theta} = A(\mathbf{Y}). \quad (4.39)$$

Typically, the decision rule  $A(\cdot)$  is a known function and we would like to determine the distributional properties of parameter estimator (4.39) as a function of the (random) observations  $\mathbf{Y}$ . E.g., for any measurable set  $C$ , we might want to compute

$$\mathbb{P}_\theta [\hat{\theta} \in C] = \mathbb{P}_\theta [A(\mathbf{Y}) \in C] = \int \mathbb{1}_{\{A(\mathbf{y}) \in C\}} dP(\mathbf{y}; \theta). \quad (4.40)$$

Since, typically, the true data generating distribution  $Y_i \sim F(\cdot; \theta)$  is not known, the distributional properties of  $\hat{\theta}$  cannot be determined, also not by Monte Carlo simulation. The idea behind bootstrap is to approximate  $F(\cdot; \theta)$ . Choose as approximation to  $F(\cdot; \theta)$  the empirical distribution of the i.i.d. observations  $\mathbf{Y}$  given by, see (3.9),

$$\hat{F}_n(y) = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{\{Y_i \leq y\}} \quad \text{for } y \in \mathbb{R}.$$

The Glivenko–Cantelli theorem [64, 159] tells us that the empirical distribution  $\hat{F}_n$  converges uniformly to  $F(\cdot; \theta)$ , a.s., for  $n \rightarrow \infty$ , so it should be a good approximation to  $F(\cdot; \theta)$  for large  $n$ . The idea now is to simulate from the empirical distribution  $\hat{F}_n$ .

---

(Non-parametric) bootstrap algorithm

---

- (1) Repeat for  $m = 1, \dots, M$ 
  - (a) simulate i.i.d. observations  $Y_1^*, \dots, Y_n^*$  from  $\hat{F}_n$  (these are obtained by random drawings with replacements from the observations  $Y_1, \dots, Y_n$ ; we denote this resampling distribution of  $\mathbf{Y}^* = (Y_1^*, \dots, Y_n^*)$  by  $\mathbb{P}^* = \mathbb{P}_{\mathbf{Y}^*}^*$ );
  - (b) calculate the estimator  $\hat{\theta}^{(m*)} = A(\mathbf{Y}^*)$ .
- (2) Return  $\hat{\theta}^{(1*)}, \dots, \hat{\theta}^{(M*)}$  and the resulting empirical bootstrap distribution

$$\hat{F}_M^*(\vartheta) = \frac{1}{M} \sum_{m=1}^M \mathbb{1}_{\{\hat{\theta}^{(m*)} \leq \vartheta\}},$$

for the estimated distribution of  $\hat{\theta}$ .

---

We can use the *empirical bootstrap distribution*  $\hat{F}_M^*$  as an estimate of the true distribution of  $\hat{\theta}$ , that is, we estimate and approximate

$$\mathbb{P}_{\hat{\theta}} [\hat{\theta} \in C] \approx \hat{\mathbb{P}}_{\hat{\theta}} [\hat{\theta} \in C] \stackrel{\text{def.}}{=} \mathbb{P}_{\mathbf{Y}^*}^* [\hat{\theta}^* \in C] \approx \frac{1}{M} \sum_{m=1}^M \mathbb{1}_{\{\hat{\theta}^{(m*)} \in C\}}, \quad (4.41)$$

where  $\mathbb{P}_{\mathbf{Y}^*}^*$  corresponds to the *bootstrap distribution* of Step (1a) of the above algorithm, and where we set  $\hat{\theta}^* = A(\mathbf{Y}^*)$ . This bootstrap distribution  $\mathbb{P}_{\mathbf{Y}^*}^*$  is empirically approximated by the empirical bootstrap distribution  $\hat{F}_M^*$  for studying  $\hat{\theta}^*$ .

*Remarks 4.29*

- The quality of the approximations in (4.41) depend on the richness of the observation  $\mathbf{Y} = (Y_1, \dots, Y_n)$ , because the bootstrap distribution

$$\mathbb{P}_{\mathbf{Y}^*}^* [\hat{\theta}^* \in C] = \mathbb{P}_{\mathbf{Y}=\mathbf{y}}^* [\hat{\theta}^* \in C],$$

depends on the realization  $\mathbf{y}$  of the data  $\mathbf{Y}$  from which we generate the bootstrap sample  $\mathbf{Y}^*$ . It also depends on  $M$  and the explicit random drawings  $Y_i^*$  providing the empirical bootstrap distribution  $\hat{F}_M^*$ . The latter uncertainty can be controlled since the bootstrap distribution  $\mathbb{P}_{\mathbf{Y}^*}^*$  corresponds to a multinomial distribution, and the Glivenko–Cantelli theorem [64, 159] applies to  $\hat{F}_M^*$  and  $\mathbb{P}_{\mathbf{Y}^*}^*$  for  $M \rightarrow \infty$ . The former uncertainty inherited from the realization  $\mathbf{Y} = \mathbf{y}$  cannot be diminished because we cannot enrich the observation  $\mathbf{Y}$ .

- The empirical bootstrap distribution  $\widehat{F}_M^*$  can be used to estimate the mean of the estimator  $\widehat{\theta}$  given in (4.39)

$$\widehat{\mathbb{E}}_{\theta} [\widehat{\theta}] = \mathbb{E}_{\mathbf{Y}^*} [\widehat{\theta}^*] \approx \frac{1}{M} \sum_{m=1}^M \widehat{\theta}^{(m*)},$$

and its variance

$$\widehat{\text{Var}}_{\theta} (\widehat{\theta}) = \text{Var}_{\mathbb{P}_{\mathbf{Y}^*}} (\widehat{\theta}^*) \approx \frac{1}{M-1} \sum_{m=1}^M \left( \widehat{\theta}^{(m*)} - \frac{1}{M} \sum_{k=1}^M \widehat{\theta}^{(k*)} \right)^2.$$

- The previous item discusses the approximation of the bootstrap mean and variance, respectively. Bootstrap intervals for coverage ratios need some care, and there are different versions. The naive way of just calculating quantiles from  $\widehat{F}_M^*$  often does not work well, and methods like a double bootstrap may need to be considered.
- In (4.39) we have assumed that the quantity of interest is the parameter  $\theta$ , but similar considerations also apply to general decision rules estimating  $\gamma(\theta)$ .
- The bootstrap as defined above directly acts on the observations  $Y_1, \dots, Y_n$ , and the basic assumption is that these observations are i.i.d. If this is not the case, one may first need to transform the observations, for instance, one can calculate residuals and assume that these residuals are i.i.d. In more complicated cases, one even drops the i.i.d. assumption and replaces it by an identical mean and variance assumption, that is, that all residuals are assumed to be independent, centered and with unit variance. This is sometimes also called *residual bootstrap* and it may be suitable in regression models as will be introduced below. Thus, in this latter case we estimate for each observation  $Y_i$  its mean  $\widehat{\mu}_i$  and its standard deviation  $\widehat{\sigma}_i$ , for instance, using the variance function of the chosen EDF. This then allows for calculating the residuals  $\widehat{\varepsilon}_i = (Y_i - \widehat{\mu}_i)/\widehat{\sigma}_i$ . For the residual bootstrap we resample the residuals  $\widehat{\varepsilon}_i^*$  from  $\widehat{\varepsilon}_1, \dots, \widehat{\varepsilon}_n$ . This provides bootstrap observations

$$Y_i^* = \widehat{\mu}_i + \widehat{\sigma}_i \widehat{\varepsilon}_i^*.$$

The *wild bootstrap* proposed by Wu [386] additionally uses a centered and normalized i.i.d. random variable  $V_i$  (also being independent of  $\widehat{\varepsilon}_i^*$ ) to modify the residual bootstrap observations to

$$Y_i^* = \widehat{\mu}_i + \widehat{\sigma}_i V_i \widehat{\varepsilon}_i^*.$$

The bootstrap is called *consistent* for  $\hat{\theta}$  if we have for all  $z \in \mathbb{R}$  the following convergence in probability as  $n \rightarrow \infty$

$$\mathbb{P}_\theta [\sqrt{n} (\hat{\theta} - \theta) \leq z] - \mathbb{P}_Y^* [\sqrt{n} (\hat{\theta}^* - \hat{\theta}) \leq z] \xrightarrow{\text{prob.}} 0,$$

the quantities  $\hat{\theta} = \hat{\theta}_n$  and  $\hat{\theta}^* = \hat{\theta}_n^*$  depend on (the size  $n$  of) the observation  $Y = Y_n$ ; the convergence in probability is needed because  $Y = Y_n$  are random vectors. Assume that  $\hat{\theta}^{\text{MLE}} = \hat{\theta}$  is the MLE of  $\theta$  satisfying the assumptions of Theorem 3.28. Then we have asymptotic normality, see (3.30),

$$\sqrt{n} (\hat{\theta} - \theta) \implies \mathcal{N}(0, \mathcal{I}_1(\theta)^{-1}) \quad \text{as } n \rightarrow \infty,$$

with Fisher's information  $\mathcal{I}_1(\theta)$ . Bootstrap consistency then requires

$$\sqrt{n} (\hat{\theta}^* - \hat{\theta}) \xrightarrow{\mathbb{P}_Y^*} \mathcal{N}(0, \mathcal{I}_1(\theta)^{-1}) \quad \text{in probability as } n \rightarrow \infty.$$

Bootstrap consistency typically holds if  $\hat{\theta}$  is asymptotically normal (as  $n \rightarrow \infty$ ) and if the underlying data  $Y_i$  is i.i.d. Moreover, bootstrap consistency usually implies consistent variance and bias estimation

$$\frac{\text{Var}_{\mathbb{P}_Y^*}(\hat{\theta}^*)}{\text{Var}_\theta(\hat{\theta})} \xrightarrow{\text{prob.}} 1 \quad \text{and} \quad \frac{\mathbb{E}_Y^*[\hat{\theta}^*] - \hat{\theta}}{\mathbb{E}_\theta[\hat{\theta}] - \theta} \xrightarrow{\text{prob.}} 1 \quad \text{as } n \rightarrow \infty.$$

For more information and bootstrap confidence intervals we refer to Chapter 5 in the lecture notes of Bühlmann–Mächler [59].

### 4.3.2 Parametric Bootstrap Simulation

For the parametric bootstrap we assume to know the parametric family  $\mathcal{F} = \{F(\cdot; \theta); \theta \in \Theta\}$  from which the i.i.d. observations  $Y_1, \dots, Y_n \sim F(\cdot; \theta)$  have been generated from, and only the explicit choice of the parameter  $\theta \in \Theta$  is not known. Based on these observations we construct an estimator  $\hat{\theta} = A(Y)$ , for the unknown parameter  $\theta \in \Theta$ .

---

(Parametric) bootstrap algorithm

---

(1) Repeat for  $m = 1, \dots, M$

- (a) simulate i.i.d. observations  $Y_1^*, \dots, Y_n^*$  from  $F(\cdot; \hat{\theta})$  (we denote the resampling distribution of  $Y^* = (Y_1^*, \dots, Y_n^*)$  by  $\mathbb{P}^* = \mathbb{P}_Y^*$ );
- (b) calculate the estimator  $\hat{\theta}^{(m*)} = A(Y^*)$ .



(2) Return  $\hat{\theta}^{(1*)}, \dots, \hat{\theta}^{(M*)}$  and the resulting empirical bootstrap distribution

$$\hat{F}_M^*(\vartheta) = \frac{1}{M} \sum_{m=1}^M \mathbb{1}_{\{\hat{\theta}^{(m*)} \leq \vartheta\}}.$$

---

We then estimate and approximate the distribution of  $\hat{\theta}$  analogously to (4.41), and the same remarks apply as for the non-parametric bootstrap. The parametric bootstrap has the advantage that it can enrich the data by sampling new observations from the distribution  $F(\cdot; \hat{\theta})$ . A shortfall of the parametric bootstrap will occur if the family  $\mathcal{F}$  is misspecified, then the bootstrap sample  $Y^*$  will only poorly describe the true data  $Y$ , e.g., if the data shows over-dispersion but the select family  $\mathcal{F}$  does not allow to model such over-dispersion.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.



## Chapter 5

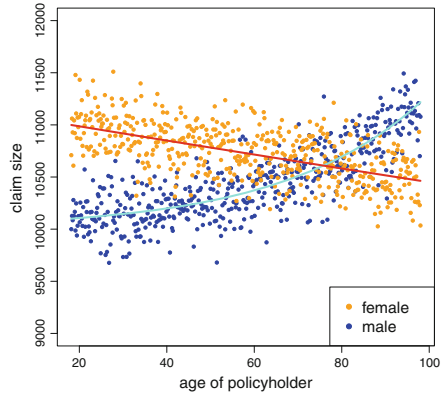
# Generalized Linear Models



Most of the theory in the previous chapters has been based on the assumption of having similarity (or homogeneity) between the different observations. This was expressed by making an i.i.d. assumption on the observations, see, e.g., Sect. 3.3.2. In many practical applications such a homogeneity assumption is not reasonable, one may for example think of car insurance pricing where different car drivers have different driving experience and they drive different cars, or of health insurance where policyholders may have different genders and ages. Figure 5.1 shows a health insurance example where the claim sizes depend on the gender and the age of the policyholders. The most popular statistical models that are able to cope with such heterogeneous data are the *generalized linear models* (GLMs). The notion of GLMs has been introduced in the seminal work of Nelder–Wedderburn [283] in 1972. Their work has introduced a unified procedure for modeling and fitting distributions within the EDF to data having systematic differences (effects) that can be described by explanatory variables. Today, GLMs are the state-of-the-art statistical models in many applied fields including statistics, actuarial science and economics. However, the specific use of GLMs in the different fields may substantially differ. In fields like actuarial science these models are mainly used for *predictive* modeling, in other fields like economics or social sciences GLMs have become the main tool in exploring and *explaining* (hopefully) causal relations. For a discussion on “predicting” versus “explaining” we refer to Shmueli [338].

It is difficult to give a good list of references for GLMs, since GLMs and their offsprings are present in almost every statistical modeling publication and in every lecture on statistics. Classical statistical references are the books of McCullagh–Nelder [265], Fahrmeir–Tutz [123] and Dobson [107], in the actuarial literature we mention the textbooks (in alphabetical order) of Charpentier [67], De Jong–Heller [89], Denuit et al. [99–101], Frees [134] and Ohlsson–Johansson [290], but this list is far from being complete.

**Fig. 5.1** Claim sizes in health insurance as a function of the age of the policyholder, and split by gender



In this chapter we introduce and discuss GLMs in the context of actuarial modeling. We do this in such a way that GLMs can be seen as a building block of network regression models which will be the main topic of Chap. 7 on deep learning.

## 5.1 Generalized Linear Models and Log-Likelihoods

### 5.1.1 Regression Modeling

We start by assuming of having independent random variables  $Y_1, \dots, Y_n$  which are described by a fixed member of the EDF. That is, we assume that all  $Y_i$  are independent and have densities w.r.t. a  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$  given by

$$Y_i \sim f(y_i; \theta_i, v_i/\varphi) = \exp \left\{ \frac{y_i \theta_i - \kappa(\theta_i)}{\varphi/v_i} + a(y_i; v_i/\varphi) \right\} \quad \text{for } 1 \leq i \leq n, \quad (5.1)$$

with canonical parameters  $\theta_i \in \overset{\circ}{\Theta}$ , exposures  $v_i > 0$  and dispersion parameter  $\varphi > 0$ . Throughout, we assume that the effective domain  $\Theta$  has a non-empty interior. There is a fundamental difference between (5.1) and Example 3.5. We now allow every random variable  $Y_i$  to have its own canonical parameter  $\theta_i \in \overset{\circ}{\Theta}$ . We call this a *heterogeneous* situation because the observations are allowed to differ in a systematic way expressed by different canonical parameters. This is highlighted by the lines in the health insurance example of Fig. 5.1 where (expected) claim sizes differ by gender and age of policyholder.

In Sect. 4.1.2 we have introduced the *saturated model* where every observation  $Y_i$  has its own parameter  $\theta_i$ . In general, if we have  $n$  observations  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  we can estimate at most  $n$  parameters. The other extreme case is the homogeneous one, meaning that  $\theta_i = \theta \in \overset{\circ}{\Theta}$  for all  $1 \leq i \leq n$ . In this latter case we have exactly one parameter to estimate, and we call this model *null model*, *intercept model* or *homogeneous model*, because all components of  $\mathbf{Y}$  are assumed to follow the

same law expressed in a single common parameter  $\theta$ . Both the saturated model and the null model may behave very poorly in predicting new observations. Typically, the saturated model fully reflects the data  $\mathbf{Y}$  including the noisy part (random component, irreducible risk, see Remarks 4.2) and, therefore, it is not useful for prediction. We also say that this model (in-sample) over-fits to the data  $\mathbf{Y}$  and does not generalize (out-of-sample) to new data. The null model often has a poor predictive performance because if the data has systematic effects these cannot be captured by a null model. GLMs try to find a good balance between these two extreme cases, by trying to extract (only) the systematic effects from noisy data  $\mathbf{Y}$ . We therefore model the canonical parameters  $\theta_i$  as a low-dimensional function of *explanatory variables* which capture the systematic effects in the data. In Fig. 5.1 gender and age of policyholder play the role of such explanatory variables.

Assume that each observation  $Y_i$  is equipped with a *feature* (explanatory variable, covariate)  $\mathbf{x}_i$  that belongs to a fixed given *feature space*  $\mathcal{X}$ . These features  $\mathbf{x}_i$  are assumed to describe the *systematic effects* in the observations  $Y_i$ , i.e., these features are assumed to be appropriate descriptions of the heterogeneity between the observations. In a nutshell, we then assume of having a suitable *regression function*

$$\theta : \mathcal{X} \rightarrow \mathring{\Theta}, \quad \mathbf{x} \mapsto \theta(\mathbf{x}),$$

such that we can appropriately describe the observations by

$$Y_i \stackrel{\text{ind.}}{\sim} f(y_i; \theta_i = \theta(\mathbf{x}_i), v_i/\varphi) = \exp \left\{ \frac{y_i \theta(\mathbf{x}_i) - \kappa(\theta(\mathbf{x}_i))}{\varphi/v_i} + a(y_i; v_i/\varphi) \right\}, \quad (5.2)$$

for  $1 \leq i \leq n$ . As a result we receive for the first moment of  $Y_i$ , see Corollary 2.14,

$$\mu_i = \mu(\mathbf{x}_i) = \mathbb{E}_{\theta(\mathbf{x}_i)} [Y_i] = \kappa'(\theta(\mathbf{x}_i)). \quad (5.3)$$

Thus, the regression function  $\theta : \mathcal{X} \rightarrow \mathring{\Theta}$  is assumed to describe the systematic differences (effects) between the random variables  $Y_1, \dots, Y_n$  being expressed by the means  $\mu(\mathbf{x}_i)$  for features  $\mathbf{x}_1, \dots, \mathbf{x}_n$ . In GLMs this regression function takes a linear form after a suitable transformation, which exactly motivates the terminology *generalized linear model*.

### 5.1.2 Definition of Generalized Linear Models

We start with the discussion of the features  $\mathbf{x} \in \mathcal{X}$ . Features are also called explanatory variables, covariates, independent variables or regressors. Throughout, we assume that the features  $\mathbf{x} = (x_0, x_1, \dots, x_q)^\top$  include a first component  $x_0 = 1$ , and we choose feature space  $\mathcal{X} \subset \{1\} \times \mathbb{R}^q$ . The inclusion of this first component  $x_0 = 1$  is useful in what follows. We call this first component *intercept* or *bias component* because it will be modeling an intercept of a regression model. The

null model (homogeneous model) has features that only consist of this intercept component. For later purposes it will be useful to introduce the *design matrix*  $\mathfrak{X}$  which collects the features  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$  of all responses  $Y_1, \dots, Y_n$ . The design matrix is defined by

$$\mathfrak{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^\top = \begin{pmatrix} 1 & x_{1,1} & \cdots & x_{1,q} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \cdots & x_{n,q} \end{pmatrix} \in \mathbb{R}^{n \times (q+1)}. \quad (5.4)$$

Based on these choices we assume existence of a *regression parameter*  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$  and of a strictly monotone and smooth *link function*  $g : \mathcal{M} \rightarrow \mathbb{R}$  such that we can express (5.3) by the following function (we drop index  $i$ )

$$\mathbf{x} \mapsto g(\mu(\mathbf{x})) = g(\mathbb{E}_{\theta(\mathbf{x})}[Y]) = \eta(\mathbf{x}) = \langle \boldsymbol{\beta}, \mathbf{x} \rangle = \beta_0 + \sum_{j=1}^q \beta_j x_j. \quad (5.5)$$

Here,  $\langle \cdot, \cdot \rangle$  describes the scalar product in the Euclidean space  $\mathbb{R}^{q+1}$ ,  $\theta(\mathbf{x}) = h(\mu(\mathbf{x}))$  is the resulting canonical parameter (using canonical link  $h = (\kappa')^{-1}$ ), and  $\eta(\mathbf{x})$  is the so-called *linear predictor*. After applying a suitable link function  $g$ , the systematic effects of the random variable  $Y$  with features  $\mathbf{x}$  can be described by a linear predictor  $\eta(\mathbf{x}) = \langle \boldsymbol{\beta}, \mathbf{x} \rangle$ , linear in the components of  $\mathbf{x} \in \mathcal{X}$ . This gives a particular functional form to (5.3), and the random variables  $Y_1, \dots, Y_n$  share a common regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ . Remark that the link function  $g$  used in (5.5) can be different from the canonical link  $h$  used to calculate  $\theta(\mathbf{x}) = h(\mu(\mathbf{x}))$ . We come back to this distinction below.

### Summary of (5.5)

1. The independent random variables  $Y_i$  follow a fixed member of the EDF (5.1) with individual canonical parameters  $\theta_i \in \mathring{\Theta}$ , for all  $1 \leq i \leq n$ .
2. The canonical parameters  $\theta_i$  and the corresponding mean parameters  $\mu_i$  are related by the canonical link  $h = (\kappa')^{-1}$  as follows  $h(\mu_i) = \theta_i$ , where  $\kappa$  is the cumulant function of the chosen EDF, see Corollary 2.14.
3. We assume that the systematic effects in the random variables  $Y_i$  can be described by linear predictors  $\eta_i = \eta(\mathbf{x}_i) = \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$  and a strictly monotone and smooth link function  $g$  such that we have  $g(\mu_i) = \eta_i = \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$ , for all  $1 \leq i \leq n$ , with common regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ .

We can either express this GLM regression structure in the dual (mean) parameter space  $\mathcal{M}$  or in the effective domain  $\mathring{\Theta}$ , see Remarks 2.9,

$$\begin{aligned} \mathbf{x} &\mapsto \mu(\mathbf{x}) = g^{-1}(\eta(\mathbf{x})) = g^{-1}\langle \boldsymbol{\beta}, \mathbf{x} \rangle \in \mathcal{M} & \text{or} \\ \mathbf{x} &\mapsto \theta(\mathbf{x}) = (h \circ g^{-1})(\eta(\mathbf{x})) = (h \circ g^{-1})\langle \boldsymbol{\beta}, \mathbf{x} \rangle \in \mathring{\Theta}, \end{aligned}$$

where  $(h \circ g^{-1})$  is the composition of the inverse link  $g^{-1}$  and the canonical link  $h$ . For the moment, the link function  $g$  is quite general. In practice, the explicit choice needs some care. The right-hand side of (5.5) is defined on the whole real line if at least one component of  $\mathbf{x}$  is both-sided unbounded. On the other hand,  $\mathcal{M}$  and  $\mathring{\Theta}$  may be bounded sets. Therefore, the link function  $g$  may require some restrictions such that the domain and the range fulfill the necessary constraints. The dimension of  $\boldsymbol{\beta}$  should satisfy  $1 \leq 1 + q \leq n$ , the lower bound will provide a null model and the upper bound a saturated model.

### 5.1.3 Link Functions and Feature Engineering

As link function we choose a strictly monotone and smooth function  $g : \mathcal{M} \rightarrow \mathbb{R}$  such that we do not have any conflicts in domains and ranges. Beside these requirements, we may want further properties for the link function  $g$  and the features  $\mathbf{x}$ . From (5.5) we have

$$\mu(\mathbf{x}) = \mathbb{E}_{\theta(\mathbf{x})} [Y] = g^{-1}\langle \boldsymbol{\beta}, \mathbf{x} \rangle. \quad (5.6)$$

Of course, a basic requirement is that the selected features  $\mathbf{x}$  can appropriately describe the mean of  $Y$  by the function in (5.6), see also Fig. 5.1. This may require so-called *feature engineering* of  $\mathbf{x}$ , for instance, we may want to replace the first component  $x_1$  of the *raw features*  $\mathbf{x}$  by, say,  $x_1^2$  in the *pre-processed features*. For example, if this first component describes the age of the insurance policyholder, then, in some regression problems, it might be more appropriate to consider  $\text{age}^2$  instead of  $\text{age}$  to bring the predictive problem into structure (5.6). It may also be that we would like to enforce a certain type of *interaction* between the components of the raw features. For instance, we may include in a pre-processed feature a component  $x_1/x_2^2$  which might correspond to  $\text{weight}/\text{height}^2$  if the policyholder has body weight  $x_1$  and body height  $x_2$ . In fact, this pre-processed feature is exactly the body mass index of the policyholder. We will come back to feature engineering in Sect. 5.2.2, below.

Another important requirement is the ability of model interpretation. In insurance pricing problems, one often prefers additive and multiplicative effects in feature components. Choosing the identity link  $g(m) = m$  we receive a model with additive effects

$$\mu(\mathbf{x}) = \mathbb{E}_{\theta(\mathbf{x})} [Y] = \langle \boldsymbol{\beta}, \mathbf{x} \rangle = \beta_0 + \sum_{j=1}^q \beta_j x_j,$$

and choosing the log-link  $g(m) = \log(m)$  we receive a model with multiplicative effects

$$\mu(\mathbf{x}) = \mathbb{E}_{\theta(\mathbf{x})} [Y] = \exp \langle \boldsymbol{\beta}, \mathbf{x} \rangle = e^{\beta_0} \prod_{j=1}^q e^{\beta_j x_j}.$$

The latter is probably the most commonly used GLM in insurance pricing because it leads to explainable tariffs where feature values directly relate to price de- and increases in percentages of a base premium  $\exp\{\beta_0\}$ .

Another very popular choice is the canonical (natural) link, i.e.,  $g = h = (\kappa')^{-1}$ . The canonical link substantially simplifies the analysis and it has very favorable statistical properties (as we will see below). However, in some applications practical needs overrule good statistical properties. Under the canonical link  $g = h$  we have in the dual mean parameter space  $\mathcal{M}$  and in the effective domain  $\Theta$ , respectively,

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = \kappa'(\eta(\mathbf{x})) = \kappa' \langle \boldsymbol{\beta}, \mathbf{x} \rangle \quad \text{and} \quad \mathbf{x} \mapsto \theta(\mathbf{x}) = \eta(\mathbf{x}) = \langle \boldsymbol{\beta}, \mathbf{x} \rangle.$$

Thus, the linear predictor  $\eta$  and the canonical parameter  $\theta$  coincide under the canonical link choice  $g = h = (\kappa')^{-1}$ .

#### 5.1.4 Log-Likelihood Function and Maximum Likelihood Estimation

After having a fully specified GLM within the EDF, there remains estimation of the regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ . This is done within the framework of MLE.

The log-likelihood function of  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  for regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$  is given by, see (5.2) and we use the independence between the  $Y_i$ 's,

(continued)

$$\boldsymbol{\beta} \mapsto \ell_Y(\boldsymbol{\beta}) = \sum_{i=1}^n \frac{v_i}{\varphi} \left[ Y_i h(\mu(\mathbf{x}_i)) - \kappa(h(\mu(\mathbf{x}_i))) \right] + a(Y_i; v_i/\varphi), \quad (5.7)$$

where we set  $\mu(\mathbf{x}_i) = g^{-1}\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$ . For the canonical link  $g = h = (\kappa')^{-1}$  this simplifies to

$$\boldsymbol{\beta} \mapsto \ell_Y(\boldsymbol{\beta}) = \sum_{i=1}^n \frac{v_i}{\varphi} \left[ Y_i \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle - \kappa \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle \right] + a(Y_i; v_i/\varphi). \quad (5.8)$$

MLE of  $\boldsymbol{\beta}$  needs maximization of log-likelihoods (5.7) and (5.8), respectively; these are the GLM counterparts to the homogeneous case treated in Section 3.3.2. We calculate the score, we set  $\eta_i = \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$  and  $\mu_i = \mu(\mathbf{x}_i) = g^{-1}\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$ ,

$$\begin{aligned} s(\boldsymbol{\beta}, Y) &= \nabla_{\boldsymbol{\beta}} \ell_Y(\boldsymbol{\beta}) = \sum_{i=1}^n \frac{v_i}{\varphi} [Y_i - \mu_i] \nabla_{\boldsymbol{\beta}} h(\mu(\mathbf{x}_i)) \\ &= \sum_{i=1}^n \frac{v_i}{\varphi} [Y_i - \mu_i] \frac{\partial h(\mu_i)}{\partial \mu_i} \frac{\partial \mu_i}{\partial \eta_i} \nabla_{\boldsymbol{\beta}} \eta(\mathbf{x}_i) \quad (5.9) \\ &= \sum_{i=1}^n \frac{v_i}{\varphi} \frac{Y_i - \mu_i}{V(\mu_i)} \left( \frac{\partial g(\mu_i)}{\partial \mu_i} \right)^{-1} \mathbf{x}_i, \end{aligned}$$

where we use the definition of the variance function  $V(\mu) = (\kappa'' \circ h)(\mu)$ , see Corollary 2.14. We define the diagonal working weight matrix, which in general depends on  $\boldsymbol{\beta}$  through the means  $\mu_i = g^{-1}\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$ ,

$$W(\boldsymbol{\beta}) = \text{diag} \left( \left( \frac{\partial g(\mu_i)}{\partial \mu_i} \right)^{-2} \frac{v_i}{\varphi} \frac{1}{V(\mu_i)} \right)_{1 \leq i \leq n} \in \mathbb{R}^{n \times n},$$

and the working residuals

$$\mathbf{R} = \mathbf{R}(Y, \boldsymbol{\beta}) = \left( \frac{\partial g(\mu_i)}{\partial \mu_i} (Y_i - \mu_i) \right)_{1 \leq i \leq n}^{\top} \in \mathbb{R}^n.$$

This allows us to write the score equations in a compact form, which provides the following proposition.



**Proposition 5.1** *The MLE for  $\beta$  is found by solving the score equations*

$$s(\beta, Y) = \nabla_{\beta} \ell_Y(\beta) = \mathfrak{X}^{\top} W(\beta) \mathbf{R}(Y, \beta) = 0.$$

*For the canonical link  $g = h = (\kappa')^{-1}$  the score equations simplify to*

$$s(\beta, Y) = \nabla_{\beta} \ell_Y(\beta) = \mathfrak{X}^{\top} \text{diag} \left( \frac{v_i}{\varphi} \right)_{1 \leq i \leq n} (Y - \kappa'(\mathfrak{X}\beta)) = 0,$$

*where  $\kappa'(\mathfrak{X}\beta) \in \mathbb{R}^n$  is understood element-wise.*

### Remarks 5.2

- In general, the MLE of  $\beta$  is not calculated by maximizing the log-likelihood function  $\ell_Y(\beta)$ , but rather by solving the score equations  $s(\beta, Y) = 0$ ; we also refer to Remarks 3.29 on M- and Z-estimators. The score equations provide the critical points for  $\beta$ , from which the global maximum of the log-likelihood function can be determined, supposed it exists.
- Existence of a MLE of  $\beta$  is not always given, similarly to Example 3.5, we may face the problem that the solution lies at the boundary of the parameter space (which itself may be an open set).
- If the log-likelihood function  $\beta \mapsto \ell_Y(\beta)$  is strictly concave, then the critical point of the score equations  $s(\beta, Y) = 0$  is unique, supposed it exists, and, henceforth, we have a unique MLE  $\hat{\beta}^{\text{MLE}}$  for  $\beta$ . Below, we give cases where the strict concavity of the log-likelihood holds.
- In general, there is no closed form solution for the MLE of  $\beta$ , except in the Gaussian case with canonical link, thus, we need to solve the score equations numerically.

Similarly to Remarks 3.17 we can calculate Fisher's information matrix w.r.t.  $\beta$  through the negative expected Hessian of  $\ell_Y(\beta)$ .

We get Fisher's information matrix w.r.t.  $\beta$

$$\mathcal{I}(\beta) = \mathbb{E}_{\beta} \left[ \nabla_{\beta} \ell_Y(\beta) (\nabla_{\beta} \ell_Y(\beta))^{\top} \right] = -\mathbb{E}_{\beta} \left[ \nabla_{\beta}^2 \ell_Y(\beta) \right] = \mathfrak{X}^{\top} W(\beta) \mathfrak{X}. \quad (5.10)$$

If the design matrix  $\mathfrak{X} \in \mathbb{R}^{n \times (q+1)}$  has full rank  $q + 1 \leq n$ , Fisher's information matrix  $\mathcal{I}(\beta)$  is positive definite.

Dispersion parameter  $\varphi > 0$  has been treated as a nuisance parameter above. Its explicit specification does not influence the MLE of  $\boldsymbol{\beta}$  because it cancels in the score equations. If necessary, we can also estimate this dispersion parameter with MLE. This requires solving the additional score equation

$$\frac{\partial}{\partial \varphi} \ell_Y(\boldsymbol{\beta}, \varphi) = \sum_{i=1}^n -\frac{v_i}{\varphi^2} \left[ Y_i h(\mu(\mathbf{x}_i)) - \kappa(h(\mu(\mathbf{x}_i))) \right] + \frac{\partial}{\partial \varphi} a(Y_i; v_i/\varphi) = 0, \quad (5.11)$$

and we can plug in the MLE of  $\boldsymbol{\beta}$  (which can be estimated independently of  $\varphi$ ). Fisher's information matrix is in this extended framework given by

$$\mathcal{I}(\boldsymbol{\beta}, \varphi) = -\mathbb{E}_{\boldsymbol{\beta}} \left[ \nabla_{(\boldsymbol{\beta}, \varphi)}^2 \ell_Y(\boldsymbol{\beta}, \varphi) \right] = \begin{pmatrix} \mathbf{X}^\top W(\boldsymbol{\beta}) \mathbf{X} & 0 \\ 0 & -\mathbb{E}_{\boldsymbol{\beta}} [\partial^2 \ell_Y(\boldsymbol{\beta}, \varphi) / \partial \varphi^2] \end{pmatrix},$$

that is, the off-diagonal terms between  $\boldsymbol{\beta}$  and  $\varphi$  are zero.

In view of Proposition 5.1 we need a root search algorithm to obtain the MLE of  $\boldsymbol{\beta}$ . Typically, one uses Fisher's scoring method or the iterative re-weighted least squares (IRLS) algorithm to solve this root search problem. This is a main result derived in the seminal work of Nelder–Wedderburn [283] and it explains the popularity of GLMs, namely, GLMs can be solved efficiently by this algorithm. Fisher's scoring method/IRLS algorithm explore the updates for  $t \geq 0$  until convergence

$$\widehat{\boldsymbol{\beta}}^{(t)} \mapsto \widehat{\boldsymbol{\beta}}^{(t+1)} = \left( \mathbf{X}^\top W(\widehat{\boldsymbol{\beta}}^{(t)}) \mathbf{X} \right)^{-1} \mathbf{X}^\top W(\widehat{\boldsymbol{\beta}}^{(t)}) \left( \mathbf{X} \widehat{\boldsymbol{\beta}}^{(t)} + \mathbf{R}(Y, \widehat{\boldsymbol{\beta}}^{(t)}) \right), \quad (5.12)$$

where all terms on the right-hand side are evaluated at algorithmic time  $t$ . If we have  $n$  observations  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  we can estimate at most  $n$  parameters. Therefore, in our GLM we assume to have a regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$  of dimension  $q + 1 \leq n$ . Moreover, we require that the design matrix  $\mathbf{X}$  has full rank  $q + 1 \leq n$ . Otherwise the regression parameter is not uniquely identifiable since linear dependence in the columns of  $\mathbf{X}$  allows us to reduce the dimension of the parameter space to a smaller representation. This is also needed to calculate the inverse matrix in (5.12). This motivates the following assumption.

**Assumption 5.3** *Throughout, we assume that the design matrix  $\mathfrak{X} \in \mathbb{R}^{n \times (q+1)}$  has full rank  $q + 1 \leq n$ .*

*Remarks 5.4 (Justification of Fisher's Scoring Method/IRLS Algorithm)*

- We give a short justification of Fisher's scoring method/IRLS algorithm, for a more detailed treatment we refer to Section 2.5 in McCullagh–Nelder [265] and Section 2.2 in Fahrmeir–Tutz [123].

The Newton–Raphson algorithm provides a numerical scheme to find solutions to the score equations. It requires to iterate for  $t \geq 0$

$$\widehat{\boldsymbol{\beta}}^{(t)} \mapsto \widehat{\boldsymbol{\beta}}^{(t+1)} = \widehat{\boldsymbol{\beta}}^{(t)} + \widehat{\mathcal{I}}(\widehat{\boldsymbol{\beta}}^{(t)})^{-1} s(\widehat{\boldsymbol{\beta}}^{(t)}, \mathbf{Y}),$$

where  $\widehat{\mathcal{I}}(\boldsymbol{\beta}) = -\nabla_{\boldsymbol{\beta}}^2 \ell_Y(\boldsymbol{\beta})$  denotes the observed information matrix in  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ . The calculation of the inverse of the observed information matrix  $(\widehat{\mathcal{I}}(\widehat{\boldsymbol{\beta}}^{(t)}))^{-1}$  can be time consuming and unstable because we need to calculate second derivatives and the eigenvalues of the observed information matrix can be close to zero. A stable scheme is obtained by replacing the observed information matrix  $\widehat{\mathcal{I}}(\boldsymbol{\beta})$  by Fisher's information matrix  $\mathcal{I}(\boldsymbol{\beta}) = \mathbb{E}_{\boldsymbol{\beta}}[\widehat{\mathcal{I}}(\boldsymbol{\beta})]$  being positive definite under Assumption 5.3; this provides a quasi-Newton method. Thus, for Fisher's scoring method we iterate for  $t \geq 0$

$$\widehat{\boldsymbol{\beta}}^{(t)} \mapsto \widehat{\boldsymbol{\beta}}^{(t+1)} = \widehat{\boldsymbol{\beta}}^{(t)} + \mathcal{I}(\widehat{\boldsymbol{\beta}}^{(t)})^{-1} s(\widehat{\boldsymbol{\beta}}^{(t)}, \mathbf{Y}), \quad (5.13)$$

and rewriting this provides us exactly with (5.12). The latter can also be interpreted as an IRLS scheme where the response  $g(Y_i)$  is replaced by an adjusted linearized version  $Z_i = g(\mu_i) + \frac{\partial g(\mu_i)}{\partial \mu_i}(Y_i - \mu_i)$ . This corresponds to the last bracket in (5.12), and with corresponding weights.

- Under the canonical link choice, Fisher's information matrix and the observed information matrix coincide, i.e.  $\mathcal{I}(\boldsymbol{\beta}) = \widehat{\mathcal{I}}(\boldsymbol{\beta})$ , and the Newton–Raphson algorithm, Fisher's scoring method and the IRLS algorithm are identical. This can easily be seen from Proposition 5.1. We receive under the canonical link choice

$$\begin{aligned} \nabla_{\boldsymbol{\beta}}^2 \ell_Y(\boldsymbol{\beta}) &= -\widehat{\mathcal{I}}(\boldsymbol{\beta}) = -\mathfrak{X}^\top \text{diag} \left( \frac{v_i}{\varphi} V(\mu_i) \right)_{1 \leq i \leq n} \mathfrak{X} \quad (5.14) \\ &= -\mathfrak{X}^\top W(\boldsymbol{\beta}) \mathfrak{X} = -\mathcal{I}(\boldsymbol{\beta}). \end{aligned}$$

The full rank assumption  $q + 1 \leq n$  on the design matrix  $\mathfrak{X}$  implies that Fisher's information matrix  $\mathcal{I}(\boldsymbol{\beta})$  is positive definite. This in turn implies that the log-likelihood function  $\ell_Y(\boldsymbol{\beta})$  is strictly concave, providing uniqueness of a critical point (supposed it exists). This indicates that the canonical link has very favorable properties for MLE. Examples 5.5 and 5.6 give two examples not using the canonical link, the first one is a concave maximization problem, the second one is not for  $p > 2$ .

*Example 5.5 (Gamma Model with Log-Link)* We study the gamma distribution as a single-parameter EDF model, choosing the shape parameter  $\alpha = 1/\varphi$  as the inverse of the dispersion parameter, see Sect. 2.2.2. Cumulant function  $\kappa(\theta) = -\log(-\theta)$  gives us the canonical link  $\theta = h(\mu) = -1/\mu$ . Moreover, we choose the log-link  $\eta = g(\mu) = \log(\mu)$  for the GLM. This gives a canonical parameter  $\theta = -\exp\{-\eta\}$ . We receive the score

$$s(\boldsymbol{\beta}, Y) = \nabla_{\boldsymbol{\beta}} \ell_Y(\boldsymbol{\beta}) = \sum_{i=1}^n \frac{v_i}{\varphi} \left[ \frac{Y_i}{\mu_i} - 1 \right] \mathbf{x}_i = \mathfrak{X}^\top \text{diag} \left( \frac{v_i}{\varphi} \right)_{1 \leq i \leq n} \mathbf{R}(Y, \boldsymbol{\beta}).$$

Unlike in other examples with non-canonical links, we receive a favorable expression here because only one term in the square bracket depends on the regression parameter  $\boldsymbol{\beta}$ , or equivalently, the working weight matrix  $W$  does not depend on  $\boldsymbol{\beta}$ . We calculate the negative Hessian (observed information matrix)

$$\widehat{\mathcal{I}}(\boldsymbol{\beta}) = -\nabla_{\boldsymbol{\beta}}^2 \ell_Y(\boldsymbol{\beta}) = \mathfrak{X}^\top \text{diag} \left( \frac{v_i}{\varphi} \frac{Y_i}{\mu_i} \right)_{1 \leq i \leq n} \mathfrak{X}.$$

In the gamma model all observations  $Y_i$  are strictly positive, a.s., and under the full rank assumption  $q + 1 \leq n$ , the observed information matrix  $\widehat{\mathcal{I}}(\boldsymbol{\beta})$  is positive definite, thus, we have a strictly concave log-likelihood function in the gamma case with log-link. ■

*Example 5.6 (Tweedie's Models with Log-Link)* We study Tweedie's models for power variance parameters  $p > 1$  as a single-parameter EDF model, see Sect. 2.2.3. The cumulant function  $\kappa_p$  is given in Table 4.1. This gives us the canonical link  $\theta = h_p(\mu) = \mu^{1-p}/(1-p) < 0$  for  $\mu > 0$  and  $p > 1$ . Moreover, we choose the log-link  $\eta = g(\mu) = \log(\mu)$  for the GLM. This implies  $\theta = \exp\{(1-p)\eta\}/(1-p) < 0$  for  $p > 1$ . We receive the score

$$s(\boldsymbol{\beta}, Y) = \nabla_{\boldsymbol{\beta}} \ell_Y(\boldsymbol{\beta}) = \sum_{i=1}^n \frac{v_i}{\varphi} \frac{Y_i - \mu_i}{\mu_i^{p-1}} \mathbf{x}_i = \mathfrak{X}^\top \text{diag} \left( \frac{v_i}{\varphi} \frac{1}{\mu_i^{p-2}} \right)_{1 \leq i \leq n} \mathbf{R}(Y, \boldsymbol{\beta}).$$

We calculate the negative Hessian (observed information matrix) for  $\mu_i > 0$

$$\widehat{\mathcal{I}}(\boldsymbol{\beta}) = -\nabla_{\boldsymbol{\beta}}^2 \ell_Y(\boldsymbol{\beta}) = \mathfrak{X}^\top \text{diag} \left( \frac{v_i (p-1)Y_i - (p-2)\mu_i}{\varphi \mu_i^{p-1}} \right)_{1 \leq i \leq n} \mathfrak{X}.$$

This matrix is positive definite for  $p \in [1, 2]$ , and for  $p > 2$  it is not positive definite because  $(p-1)Y_i - (p-2)\mu_i$  may have positive or negative values if we vary  $\mu_i > 0$  over its domain  $\mathcal{M}$ . Thus, we do not have concavity of the optimization problem under the log-link choice in Tweedie's GLMs for power variance parameters  $p > 2$ . This in particular applies to the inverse Gaussian GLM with log-link. ■

### 5.1.5 Balance Property Under the Canonical Link Choice

Throughout this section we work under the canonical link choice  $g = h = (\kappa')^{-1}$ . This choice has very favorable statistical properties. We have already seen in Remarks 5.4 that the derivation of the MLE of  $\boldsymbol{\beta}$  becomes particularly easy under the canonical link choice and the observed information matrix  $\widehat{\mathcal{I}}(\boldsymbol{\beta})$  coincides with Fisher's information matrix  $\mathcal{I}(\boldsymbol{\beta})$  in this case, see (5.14).

For insurance pricing, canonical links have another very remarkable property, namely, that the estimated model automatically fulfills the balance property and, henceforth, is unbiased. This is particularly important in insurance pricing because it tells us that the insurance prices (over the entire portfolio) are on the right level. We have already met the balance property in Corollary 3.19.

**Corollary 5.7 (Balance Property)** *Assume that  $\mathbf{Y}$  has independent components being modeled by a GLM under the canonical link choice  $g = h = (\kappa')^{-1}$ . Assume that the MLE of regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$  exists and denote it by  $\widehat{\boldsymbol{\beta}}^{\text{MLE}}$ . We have balance property on portfolio level (for constant dispersion  $\varphi$ )*

$$\sum_{i=1}^n \mathbb{E}_{\widehat{\boldsymbol{\beta}}^{\text{MLE}}} [v_i Y_i] = \sum_{i=1}^n v_i \kappa'(\widehat{\boldsymbol{\beta}}^{\text{MLE}}, \mathbf{x}_i) = \sum_{i=1}^n v_i Y_i.$$

**Proof** The first column of the design matrix  $\mathfrak{X}$  is identically equal to 1 representing the intercept, see (5.4). The second part of Proposition 5.1 then provides for this first column of  $\mathfrak{X}$ , we cancel the (constant) dispersion  $\varphi$ ,

$$(1, \dots, 1) \text{diag}(v_1, \dots, v_n) \kappa'(\mathfrak{X} \widehat{\boldsymbol{\beta}}^{\text{MLE}}) = (1, \dots, 1) \text{diag}(v_1, \dots, v_n) \mathbf{Y}.$$

This proves the claim. □

*Remark 5.8* We mention once more that this balance property is very strong and useful, see also Remarks 3.20. In particular, the balance property holds, even though the chosen GLM might be completely misspecified. Misspecification may include an incorrect distributional model, not the right link function choice, or if we have not pre-processed features appropriately, etc. Such misspecification will imply that we have a poor model on an insurance policy level (observation level). However, the total premium charged over the entire portfolio will be on the right level (supposed that the structure of the portfolio does not change) because it matches the observations, and henceforth, we have unbiasedness for the portfolio mean.

From the log-likelihood function (5.8) we see that under the canonical link choice we consider the statistics  $S(\mathbf{Y}) = \mathfrak{X}^\top \text{diag}(v_i/\varphi)_{1 \leq i \leq n} \mathbf{Y} \in \mathbb{R}^{q+1}$ , and to prove the balance property we have used the first component of this statistics. Considering all components,  $S(\mathbf{Y})$  is an unbiased estimator (decision rule) for

$$\mathbb{E}_{\boldsymbol{\beta}} [S(\mathbf{Y})] = \mathfrak{X}^\top \text{diag}(v_i/\varphi)_{1 \leq i \leq n} \kappa'(\mathfrak{X}\boldsymbol{\beta}) = \left( \sum_{i=1}^n \frac{v_i}{\varphi} \kappa'(\boldsymbol{\beta}, \mathbf{x}_i) x_{i,j} \right)_{0 \leq j \leq q}^\top. \quad (5.15)$$

This unbiased estimator  $S(\mathbf{Y})$  meets the Cramér–Rao information bound, hence it is UMVU: taking the partial derivatives of the previous expression gives  $\nabla_{\boldsymbol{\beta}} \mathbb{E}_{\boldsymbol{\beta}} [S(\mathbf{Y})] = \mathcal{I}(\boldsymbol{\beta})$ , the latter also being the multivariate Cramér–Rao information bound for the unbiased decision rule  $S(\mathbf{Y})$  for (5.15). Focusing on the first component we have

$$\text{Var}_{\boldsymbol{\beta}} \left( \sum_{i=1}^n \mathbb{E}_{\hat{\boldsymbol{\beta}}^{\text{MLE}}} [v_i Y_i] \right) = \text{Var}_{\boldsymbol{\beta}} \left( \sum_{i=1}^n v_i Y_i \right) = \sum_{i=1}^n \varphi v_i V(\mu_i) = \varphi^2 (\mathcal{I}(\boldsymbol{\beta}))_{0,0}, \quad (5.16)$$

where the component  $(0, 0)$  in the last expression is the top-left entry of Fisher’s information matrix  $\mathcal{I}(\boldsymbol{\beta})$  under the canonical link choice.

### 5.1.6 Asymptotic Normality

Formula (5.16) quantifies the uncertainty in the premium calculation of the insurance policies if we use the MLE estimated model (under the canonical link choice). That is, this quantifies the uncertainty in the dual mean parametrization in terms of the resulting variance. We could also focus on the MLE  $\hat{\boldsymbol{\beta}}^{\text{MLE}}$  itself (for general link function  $g$ ). In general, this MLE is not unbiased but we have

consistency and asymptotic normality similar to Theorem 3.28. Under “certain regularity conditions”<sup>1</sup> we have for  $n$  large

$$\hat{\beta}_n^{\text{MLE}} \stackrel{(d)}{\approx} \mathcal{N}(\beta, \mathcal{I}_n(\beta)^{-1}), \quad (5.17)$$

where  $\hat{\beta}_n^{\text{MLE}}$  is the MLE based on the observations  $\mathbf{Y}_n = (Y_1, \dots, Y_n)^\top$ , and  $\mathcal{I}_n(\beta)$  is Fisher’s information matrix of  $\mathbf{Y}_n$ , which scales linearly in  $n$  in the homogeneous EF case, see Remarks 3.14, and in the homogeneous EDF case it scales as  $\sum_{i=1}^n v_i$ , see (3.25).

### 5.1.7 Maximum Likelihood Estimation and Unit Deviances

From formula (5.7) we conclude that the MLE  $\hat{\beta}^{\text{MLE}}$  of  $\beta \in \mathbb{R}^{q+1}$  is found by the solution of (subject to existence)

$$\hat{\beta}^{\text{MLE}} = \arg \max_{\beta} \ell_Y(\beta) = \arg \max_{\beta} \sum_{i=1}^n \frac{v_i}{\varphi} \left[ Y_i h(\mu(\mathbf{x}_i)) - \kappa(h(\mu(\mathbf{x}_i))) \right],$$

with  $\mu_i = \mu(\mathbf{x}_i) = \mathbb{E}_{\theta(\mathbf{x}_i)}[Y] = g^{-1}\langle \beta, \mathbf{x}_i \rangle$  under the link choice  $g$ . If we prefer to work with an objective function that reflects the notion of a loss function, we can work under the unit deviances  $\mathfrak{d}(Y_i, \mu_i)$  studied in Sect. 4.1.2. The MLE is then obtained by, see (4.20)–(4.21),

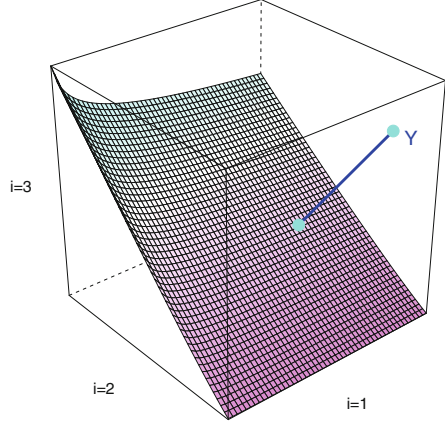
$$\hat{\beta}^{\text{MLE}} = \arg \max_{\beta} \ell_Y(\beta) = \arg \min_{\beta} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \mu_i), \quad (5.18)$$

the latter satisfying  $\mathfrak{d}(Y_i, \mu_i) \geq 0$  for all  $1 \leq i \leq n$ , and being zero if and only if  $Y_i = \mu_i$ , see Lemma 2.22. Thus, using the unit deviances we have a loss function that is bounded below by zero, and we determine the regression parameter  $\beta$  such that this loss is (in-sample) minimized. This can also be interpreted in a more geometric way. Consider the  $(q+1)$ -dimensional manifold  $\mathfrak{M} \subset \mathbb{R}^n$  spanned by the GLM function

$$\beta \mapsto \mu(\beta) = g^{-1}(\mathfrak{X}\beta) = (g^{-1}\langle \beta, \mathbf{x}_1 \rangle, \dots, g^{-1}\langle \beta, \mathbf{x}_n \rangle)^\top \in \mathbb{R}^n. \quad (5.19)$$

<sup>1</sup> The regularity conditions for asymptotic normality results will depend on the particular regression problem studied, we refer to pages 43–44 in Fahrmeir–Tutz [123].

**Fig. 5.2** 2-dimensional manifold  $\mathfrak{M} \subset \mathbb{R}^3$  for observation  $\mathbf{Y} = (Y_1, Y_2, Y_3)^\top \in \mathbb{R}^3$ , the straight line illustrates the projection (w.r.t. the unit deviance distances  $\mathfrak{d}$ ) of  $\mathbf{Y}$  onto  $\mathfrak{M}$  which gives MLE  $\hat{\boldsymbol{\beta}}^{\text{MLE}}$  satisfying  $\boldsymbol{\mu}(\hat{\boldsymbol{\beta}}^{\text{MLE}}) \in \mathfrak{M}$



Minimization (5.18) then tries to find the point  $\boldsymbol{\mu}(\boldsymbol{\beta})$  in this manifold  $\mathfrak{M} \subset \mathbb{R}^n$  that minimizes simultaneously all unit deviances  $\mathfrak{d}(Y_i, \cdot)$  w.r.t. the observation  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top \in \mathbb{R}^n$ . Or in other words, the optimal parameter  $\boldsymbol{\beta}$  is obtained by “projecting” observation  $\mathbf{Y}$  onto this manifold  $\mathfrak{M}$ , where “projection” is understood as a simultaneous minimization of loss function  $\sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \mu_i)$ , see Fig. 5.2. In the un-weighted Gaussian case, this corresponds to the usual orthogonal projection as the next example shows, and in the non-Gaussian case it is understood in the KL divergence minimization sense as displayed in formula (4.11).

*Example 5.9 (Gaussian Case)* Assume we have the Gaussian EDF case  $\kappa(\theta) = \theta^2/2$  with canonical link  $g(\mu) = h(\mu) = \mu$ . In this case, the manifold (5.19) is the linear space spanned by the columns of the design matrix  $\mathfrak{X}$

$$\boldsymbol{\beta} \mapsto \boldsymbol{\mu}(\boldsymbol{\beta}) = \mathfrak{X}\boldsymbol{\beta} = (\langle \boldsymbol{\beta}, \mathbf{x}_1 \rangle, \dots, \langle \boldsymbol{\beta}, \mathbf{x}_n \rangle)^\top \in \mathbb{R}^n.$$

If additionally we assume  $v_i/\varphi = c > 0$  for all  $1 \leq i \leq n$ , the minimization problem (5.18) reads as

$$\hat{\boldsymbol{\beta}}^{\text{MLE}} = \arg \min_{\boldsymbol{\beta}} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \mu_i) = \arg \min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathfrak{X}\boldsymbol{\beta}\|_2^2,$$

where we have used that the unit deviances in the Gaussian case are given by the square loss function, see Example 4.12. As a consequence, the MLE  $\hat{\boldsymbol{\beta}}^{\text{MLE}}$  is found by orthogonally projecting  $\mathbf{Y}$  onto  $\mathfrak{M} = \{\mathfrak{X}\boldsymbol{\beta} \mid \boldsymbol{\beta} \in \mathbb{R}^{q+1}\} \subset \mathbb{R}^n$ , and this orthogonal projection is given by  $\mathfrak{X}\hat{\boldsymbol{\beta}}^{\text{MLE}} \in \mathfrak{M}$ . ■



## 5.2 Actuarial Applications of Generalized Linear Models

The purpose of this section is to illustrate how the concept of GLMs is used in actuarial modeling. We therefore explore the typical actuarial examples of claim counts and claim size modeling.

### 5.2.1 *Selection of a Generalized Linear Model*

The selection of a predictive model within GLMs for solving an applied actuarial problem requires the following choices.

**Choice of the Member of the EDF** Select a member of the EDF that fits the modeling problem. In a first step, we should try to understand the properties of the data  $Y$  before doing this selection, for instance, do we have count data, do we have a classification problem, do we have continuous observations?

All members of the EDF are light-tailed because the moment generating function exists around the origin, see Corollary 2.14, and the EDF is not suited to model heavy-tailed data, for instance, having a regularly varying tail. Therefore, a datum  $Y$  is sometimes first transformed before being modeled by a member of the EDF. A popular transformation is the logarithm for positive observations. After this transformation a member of the EDF can be chosen to model  $\log(Y)$ . For instance, if we choose the Gaussian distribution for  $\log(Y)$ , then  $Y$  will be log-normally distributed, or if we choose the exponential distribution for  $\log(Y)$ , then  $Y$  will be Pareto distributed, see Sect. 2.2.5. One can then model the transformed datum with a GLM. Often this provides very accurate models, say, on the log scale for the log-transformed data. There is one issue with this approach, namely, if a model is unbiased on the transformed scale then it is typically biased on the original observation scale; if the transformation is concave this easily follows from Jensen's inequality. The problematic part now is that the bias correction itself often has systematic effects which means that the transformation (or the involved nuisance parameters) should be modeled with a regression model, too, see Sect. 5.3.9. In many cases this will not easily work, unfortunately. Therefore, if possible, clear preference should be given to modeling the data on the original observation scale (if unbiasedness is a central requirement).

**Choice of Link Function** From a statistical point of view we should choose the canonical link  $g = h$  to connect the mean  $\mu$  of the model to the linear predictor  $\eta$  because this implies many favorable mathematical properties. However, as seen, sometimes we have different needs. Practical reasons may require that we have a model with additive or multiplicative effects, which favors the identity or the log-link, respectively. Another requirement is that the resulting canonical parameter  $\theta = (h \circ g^{-1})(\eta)$  needs to be within the effective domain  $\Theta$ . If this effective domain is bounded, for instance, if it covers the negative real line as for the gamma model,

a (transformation of the) log-link might be more suitable than the canonical link because  $g^{-1}(\cdot) = -\exp(\cdot)$  has a strictly negative range, see Example 5.5.

**Choice of Features and Feature Engineering** Assume we have selected the member of the EDF and the link function  $g$ . This gives us the relationship between the mean  $\mu$  and the linear predictor  $\eta$ , see (5.5),

$$\mu(\mathbf{x}) = \mathbb{E}_{\theta(\mathbf{x})}[Y] = g^{-1}(\eta(\mathbf{x})) = g^{-1}(\boldsymbol{\beta}, \mathbf{x}). \quad (5.20)$$

Thus, the features  $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^{q+1}$  need to be in the right functional form so that they can appropriately describe the systematic effect via the function (5.20). We distinguish the following feature types:

- *Continuous real-valued feature components*, examples are age of policyholder, weight of car, body mass index, etc.
- *Ordinal categorical feature components*, examples are ratings like good-medium-bad or A-B-C-D-E.
- *Nominal categorical feature components*, examples are vehicle brands, occupation of policyholders, provinces of living places of policyholders, etc. The values that the categorical feature components can take are called *levels*.
- *Binary feature components* are special categorical features that only have two levels, e.g. female-male, open-closed. Because binary variables often play a distinguished role in modeling they are separated from categorical variables which are typically assumed to have more than two levels.

All these components need to be brought into a suitable form so that they can be used in a linear predictor  $\eta(\mathbf{x}) = \langle \boldsymbol{\beta}, \mathbf{x} \rangle$ , see (5.20). This requires the consideration of the following points (1) transformation of continuous components so that they can describe the systematic effects in a linear form, (2) transformation of categorical components to real-valued components, (3) interaction of components beyond an additive structure in the linear predictor, and (4) the resulting design matrix  $\mathfrak{X}$  should have full rank  $q + 1 \leq n$ . We are going to describe these points (1)–(4) in the next section.

### 5.2.2 Feature Engineering

#### Categorical Feature Components: Dummy Coding

Categorical variables need to be embedded into a Euclidean space. This embedding needs to be done such that the resulting design matrix  $\mathfrak{X}$  has full rank  $q + 1 \leq n$ . There are many different ways to do so, and the particular choice depends on the modeling purpose. The most popular way is *dummy coding*. We only describe dummy coding here because it is sufficient for our purposes, but we mention that

**Table 5.1** Dummy coding example that maps the  $K = 11$  levels (colors) to the unit vectors of the 10-dimensional Euclidean space  $\mathbb{R}^{10}$  selecting the last level  $a_{11}$  (brown color) as reference level, and showing the resulting dummy vectors  $\mathbf{x}_j^\top$  as row vectors

$a_1 = \text{white}$	1	0	0	0	0	0	0	0	0	0
$a_2 = \text{yellow}$	0	1	0	0	0	0	0	0	0	0
$a_3 = \text{orange}$	0	0	1	0	0	0	0	0	0	0
$a_4 = \text{red}$	0	0	0	1	0	0	0	0	0	0
$a_5 = \text{magenta}$	0	0	0	0	1	0	0	0	0	0
$a_6 = \text{violet}$	0	0	0	0	0	1	0	0	0	0
$a_7 = \text{blue}$	0	0	0	0	0	0	1	0	0	0
$a_8 = \text{cyan}$	0	0	0	0	0	0	0	1	0	0
$a_9 = \text{green}$	0	0	0	0	0	0	0	0	1	0
$a_{10} = \text{beige}$	0	0	0	0	0	0	0	0	0	1
$a_{11} = \text{brown}$	0	0	0	0	0	0	0	0	0	0

there are also other codings like effects coding or Helmert's contrast coding.<sup>2</sup> The choice of the coding will not influence the predictive model (if we work with a full rank design matrix), but it may influence parameter selection, parameter reduction and model interpretation. For instance, the choice of the coding is (more) important in medical studies where one tries to understand the effects between certain therapies.

Assume that the raw feature component  $\tilde{x}_j$  is a categorical variable taking  $K$  different levels  $\{a_1, \dots, a_K\}$ . For dummy coding we declare one level, say  $a_K$ , to be the reference level and all other levels are described relative to that reference level. Formally, this can be described by an embedding map

$$\tilde{x}_j \mapsto \mathbf{x}_j = (\mathbb{1}_{\{\tilde{x}_j=a_1\}}, \dots, \mathbb{1}_{\{\tilde{x}_j=a_{K-1}\}})^\top \in \mathbb{R}^{K-1}. \quad (5.21)$$

This is closely related to the categorical distribution in Sect. 2.1.4. An explicit example is given in Table 5.1.

*Example 5.10 (Multiplicative Model)* If we choose the log-link function  $\eta = g(\mu) = \log(\mu)$ , we receive the regression function for the categorical example of Table 5.1

$$\tilde{x}_j \mapsto \exp\langle \boldsymbol{\beta}, \mathbf{x}_j \rangle = \exp\{\beta_0\} \prod_{k=1}^{K-1} \exp\{\beta_k \mathbb{1}_{\{\tilde{x}_j=a_k\}}\}, \quad (5.22)$$

including an intercept component. Thus, the base value  $\exp\{\beta_0\}$  is determined by the reference level  $a_{11} = \text{brown}$ , and any color different from brown has a deviation from the base value described by the multiplicative correction term  $\exp\{\beta_k \mathbb{1}_{\{\tilde{x}_j=a_k\}}\}$ . ■

<sup>2</sup> There is an example of Helmert's contrast coding in Remarks 2.7 of lecture notes [392], and for more examples we refer to the UCLA statistical consulting website: <https://stats.idre.ucla.edu/r/library/r-library-contrast-coding-systems-for-categorical-variables/>.

*Remarks 5.11*

- Importantly, dummy coding leads to full rank design matrices  $\mathfrak{X}$  and, henceforth, Assumption 5.3 is fulfilled.
- Dummy coding is different from one-hot encoding which is going to be introduced in Sect. 7.3.1, below.
- Dummy coding needs some care if we have categorical feature components with many levels, for instance, considering car brands and car models we can get hundreds of levels. In that case we will have sparsity in the resulting design matrix. This may cause computational issues, and, as the following example will show, it may lead to high uncertainty in parameter estimation. In particular, the columns of the design matrix  $\mathfrak{X}$  of very rare levels will be almost collinear which implies that we do not receive very well-conditioned matrices in Fisher's scoring method (5.12). For this reason, it is recommended to merge levels to bigger classes. In Sect. 7.3.1, below, we are going to present a different treatment. Categorical variables are embedded into low-dimensional spaces, so that proximity in these spaces has a reasonable meaning for the regression task at hand.

*Example 5.12 (Balance Property and Dummy Coding)* A main argument for the use of the canonical link function has been the fulfillment of the balance property, see Corollary 5.7. If we have categorical feature components and if we apply dummy coding to those, then the balance property is projected down to the individual levels of that categorical variable. Assume that columns 2 to  $K$  of design matrix  $\mathfrak{X}$  are used to model a raw categorical feature  $\tilde{x}_1$  with  $K$  levels according to (5.21). In that case, columns  $2 \leq k \leq K$  will indicate all observations  $Y_i$  which belong to levels  $a_{k-1}$ . Analogously to the proof of Corollary 5.7, we receive (summation  $i$  runs over the different instances/policies)

$$\sum_{i: \tilde{x}_{i,1}=a_{k-1}} \mathbb{E}_{\hat{\beta}^{\text{MLE}}} [v_i Y_i] = \sum_{i=1}^n x_{i,k} \mathbb{E}_{\hat{\beta}^{\text{MLE}}} [v_i Y_i] = \sum_{i=1}^n x_{i,k} v_i Y_i = \sum_{i: \tilde{x}_{i,1}=a_{k-1}} v_i Y_i. \quad (5.23)$$

Thus, we receive the balance property for all policies  $1 \leq i \leq n$  that belong to level  $a_{k-1}$ .

If we have many levels, then it will happen that some levels have only very few observations, and the above summation (5.23) only runs over very few insurance policies with  $\tilde{x}_{i,1} = a_{k-1}$ . Suppose additionally the volumes  $v_i$  are small. This can lead to considerable estimation uncertainty, because the estimated prices on the left-hand side of (5.23) will be based too much on individual observations  $Y_i$  having the corresponding level, and we are not in the regime of a law of large numbers that balances these observations.

Thus, this balance property from dummy coding is a natural property under the canonical link choice. Actuarial pricing is very familiar with such a property. Early

distribution-free approaches have postulated this property resulting in the method of the total marginal sums, see Bailey and Jung [22, 206], where the balance property is enforced for marginal sums of all categorical levels in parameter estimation. However, if we have scarce levels in categorical variables, this approach needs careful consideration. ■

## Binary Feature Components

Binary feature components do not need a treatment different from the categorical ones, they are Bernoulli variables which can be encoded as 0 or 1. This is exactly dummy coding for  $K = 2$  levels.

## Continuous Feature Components

Continuous feature components are already real-valued. Therefore, from the viewpoint of ‘variable types’, the continuous feature components do not need any pre-processing because they are already in the right format to be included in scalar products.

Nevertheless, in many cases, also continuous feature components need feature engineering because only in rare cases they directly fit the functional form (5.20). We give an example. Consider car drivers that have different driving experience and different driving skills. To explain experience and skills we typically choose the age of driver as explanatory variable. Modeling the claim frequency as a function of the age of driver, we often observe a U-shaped function, thus, a function that is non-monotone in the age of driver variable. Since the link function  $g$  needs to be strictly monotone, this regression problem cannot be modeled by (5.20), directly including the age of driver as a feature because this leads to monotonicity of the regression function in the age of driver variable.

Typically, in such situations, the continuous variable is discretized to categorical classes. In the driver’s age example, we build age classes. These age classes are then treated as categorical variables using dummy coding (5.21). We will give examples below. These age classes should fulfill the requirement of being sufficiently homogeneous in the sense that insurance policies that fall into the same class should have a similar propensity to claims. This implies that we would like to have many small homogeneous classes. However, the classes should be sufficiently large, otherwise parameter estimation involves high uncertainty, see also Example 5.12. Thus, there is a trade-off to sufficiently meet both of these two requirements.

A disadvantage of this discretization approach is that neighboring age classes will not be recognized by the regression function because, per se, dummy coding is based on nominal variables not having any topology. This is also illustrated by the fact, that all categorical levels (excluding the reference level) have, in view

of embedding (5.21), the same mutual Euclidean distance. Therefore, in some applications, one prefers a different approach by rather trying to find an appropriate functional form. For instance, we can pre-process a strictly positive raw feature component  $\tilde{x}_l$  to a higher-dimensional functional form

$$\tilde{x}_l \mapsto \beta_1 \tilde{x}_l + \beta_2 \tilde{x}_l^2 + \beta_3 \tilde{x}_l^3 + \beta_4 \log(\tilde{x}_l), \quad (5.24)$$

with regression parameter  $(\beta_1, \dots, \beta_4)^\top$ , i.e., we have a polynomial function of degree 3 plus a logarithmic term in this choice. If one does not want to choose a specific functional form, one often chooses natural cubic splines. This, together with regularization, leads to the framework of generalized additive models (GAMs), which is popular family of regression models besides GLMs; for literature on GAMs we refer to Hastie–Tibshirani [182], Wood [384], Ohlsson–Johansson [290], Denuit et al. [99] and Wüthrich–Buser [392]. In these notes we will not further pursue GAMs.

*Example 5.13 (Multiplicative Model)* If we choose the log-link function  $\eta = g(\mu) = \log(\mu)$  we receive a multiplicative regression function

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = \exp\langle \boldsymbol{\beta}, \mathbf{x} \rangle = \exp\{\beta_0\} \prod_{j=1}^q \exp\{\beta_j x_j\}.$$

That is, all feature components  $x_j$  enter the regression function in an exponential form. In general insurance, one may have specific variables for which it is explicitly known that they should enter the regression function as a power function. Having a raw feature  $\tilde{x}_l$  we can pre-process it as  $\tilde{x}_l \mapsto x_l = \log(\tilde{x}_l)$ . This implies

$$\mu(\mathbf{x}) = \exp\langle \boldsymbol{\beta}, \mathbf{x} \rangle = \exp\{\beta_0\} \tilde{x}_l^{\beta_l} \prod_{j=1, j \neq l}^q \exp\{\beta_j x_j\},$$

which gives a power term of order  $\beta_l$ . The GLM estimates in this case the power parameter that should be used for  $\tilde{x}_l$ . If the power parameter is known, then one can even include this component as an offset; offsets are discussed in Sect. 5.2.3, below. ■

## Interactions

Naturally, GLMs only allow for an additive structure in the linear predictor. Similar to continuous feature components, such an additive structure may not always be suitable and one wants to model more complex interaction terms. Such interactions need to be added manually by the modeler, for instance, if we have two raw feature

components  $\tilde{x}_l$  and  $\tilde{x}_k$ , we may want to consider a functional form

$$(\tilde{x}_l, \tilde{x}_k) \mapsto \beta_1 \tilde{x}_l + \beta_2 \tilde{x}_k + \beta_3 \tilde{x}_l \tilde{x}_k + \beta_4 \tilde{x}_l^2 \tilde{x}_k,$$

with regression parameter  $(\beta_1, \dots, \beta_4)^\top$ .

More generally, this manual feature engineering of adding interactions and of specifying functional forms (5.24) can be understood as a new representation of raw features. Representation learning in relation to deep learning is going to be discussed in Sect. 7.1, and this discussion is also related to Mercer's kernels.

### 5.2.3 Offsets

In many heterogeneous portfolio problems with observations  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$ , there are known prior differences between the individual risks  $Y_i$ , for instance, the time exposure varies between the different policies  $i$ . Such known prior differences can be integrated into the predictors, and this integration typically does not involve any additional model parameters. A simple way is to use an *offset* (constant) in the linear predictor of a GLM. Assume that each observation  $Y_i$  is equipped with a feature  $\mathbf{x}_i \in \mathcal{X}$  and a known offset  $o_i \in \mathbb{R}$  such that the linear predictor  $\eta_i$  takes the form

$$(\mathbf{x}_i, o_i) \mapsto g(\mu_i) = \eta_i = \eta(\mathbf{x}_i, o_i) = o_i + \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle, \quad (5.25)$$

for all  $1 \leq i \leq n$ . An offset  $o_i$  does not change anything from a structural viewpoint, in fact, it could be integrated into the feature  $\mathbf{x}_i$  with a regression parameter that is identically equal to 1.

Offsets are frequently used in Poisson models with the (canonical) log-link choice to model multiplicative time exposures in claim frequency modeling. Under the log-link choice we receive from (5.25) the following mean function

$$(\mathbf{x}_i, o_i) \mapsto \mu(\mathbf{x}_i, o_i) = \exp\{\eta(\mathbf{x}_i, o_i)\} = \exp\{o_i + \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle\} = \exp\{o_i\} \exp\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle.$$

In this version, the offset  $o_i$  provides us with an exposure  $\exp\{o_i\}$  that acts multiplicatively on the regression function. If  $w_i = \exp\{o_i\}$  measures time, then  $w_i$  is a so-called pro-rata temporis (proportional in time) exposure.

*Remark 5.14 (Boosting)* A popular machine learning technique in statistical modeling is boosting. Boosting tries to step-wise adaptively improve a regression model. Offsets (5.25) are a simple way of constructing boosted models. Assume we have constructed a predictive model using any statistical model, and denote the resulting estimated means of  $Y_i$  by  $\hat{\mu}_i^{(0)}$ . The idea of boosting is that we select another statistical model and we try to see whether this second model can still find systematic structure in the data which has not been found by the first model. In view

of (5.25), we include the first model into the offset and we build a second model around this offset, that is, we may explore a GLM

$$\widehat{\mu}_i^{(1)} = g^{-1} \left( g(\widehat{\mu}_i^{(0)}) + \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle \right).$$

If the first model is perfect we come up with a regression parameter  $\boldsymbol{\beta} = 0$ , otherwise the linear predictor  $\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$  of the second model starts to compensate for weaknesses in  $\widehat{\mu}_i^{(0)}$ . Of course, this boosting procedure can then be iterated and one should stop boosting before the resulting model starts to over-fit to the data. Typically, this approach is applied to regression trees instead of GLMs, see Ferrario–Hämmerli [125], Section 7.4 in Wüthrich–Buser [392], Lee–Lin [241] and Denuit et al. [100].

### 5.2.4 Lab: Poisson GLM for Car Insurance Frequencies

We present a first GLM example. This example is based on French motor third party liability (MTPL) insurance claim counts data. The data is described in detail in Chap. 13.1; an excerpt of the available MTPL data is given in Listing 13.2. For the moment we only consider claim frequency modeling. We use the following data:  $N_i$  describes the number of claims,  $v_i \in (0, 1]$  describes the duration of the insurance policy, and  $\tilde{\mathbf{x}}_i$  describes the available raw feature information of insurance policy  $i$ , see Listing 13.2.

We are going to model the claim counts  $N_i$  with a Poisson GLM using the canonical link function of the Poisson model. In the Poisson approach there are two different ways to account for the duration of the insurance policy. Either we model  $Y_i = N_i / v_i$  with the Poisson model of the EDF, see Sect. 2.2.2 and Remarks 2.13 (reproductive form), or we directly model  $N_i$  with the Poisson distribution from the EF and treat the log-duration as an offset variable  $o_i = \log v_i$ . In the first approach we have for the log-link choice  $g(\cdot) = h(\cdot) = \log(\cdot)$  and dispersion  $\varphi = 1$

$$Y_i = N_i / v_i \sim f(y_i; \theta_i, v_i) = \exp \left\{ \frac{y_i \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle - e^{\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle}}{1/v_i} + a(y_i; v_i) \right\}, \quad (5.26)$$

where  $\mathbf{x}_i \in \mathcal{X}$  is the suitably pre-processed feature information of insurance policy  $i$ , and with canonical parameter  $\theta_i = \eta(\mathbf{x}_i) = \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$ . In the second approach we include the log-duration as offset into the regression function and model  $N_i$  with



the Poisson distribution from the EF. Using notation (2.2) this gives us

$$\begin{aligned} N_i &\sim f(n_i; \theta_i) = \exp \left\{ n_i (\log v_i + \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle) - e^{\log v_i + \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle} + a(n_i) \right\} \quad (5.27) \\ &= \exp \left\{ \frac{\frac{n_i}{v_i} \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle - e^{\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle}}{1/v_i} + a(n_i) + n_i \log v_i \right\}, \end{aligned}$$

with canonical parameter  $\theta_i = \eta(\mathbf{x}_i, o_i) = o_i + \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle = \log v_i + \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$  for observation  $n_i = v_i y_i$ . That is, we receive the same model in both cases (5.26) and (5.27) under the canonical log-link choice for the Poisson GLM.

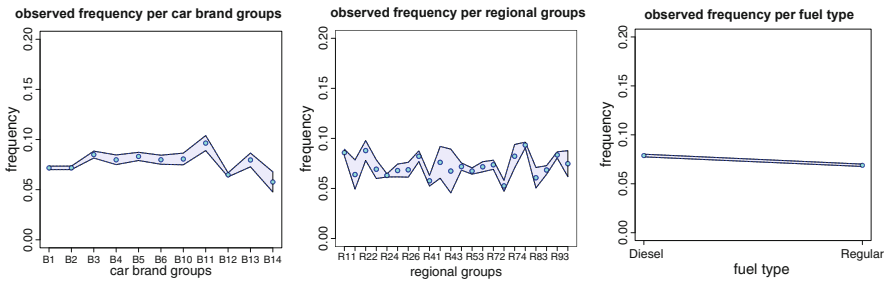
Finally, we make the assumption that all observations  $N_i$  are independent. There remains the pre-processing of the raw features  $\tilde{\mathbf{x}}_i$  to features  $\mathbf{x}_i$  so that they can be used in a sensible way in the linear predictors  $\eta_i = \eta(\mathbf{x}_i, o_i) = o_i + \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$ .

## Feature Engineering

### Categorical and Binary Variables: Dummy Coding

For categorical and binary variables we use dummy coding as described in Sect. 5.2.2. We have two categorical variables VehBrand and Region, as well as a binary variable VehGas, see Listing 13.2. We choose the first level as reference level, and the remaining levels are characterized by  $(K - 1)$ -dimensional embeddings (5.21). This provides us with  $K - 1 = 10$  parameters for VehBrand,  $K - 1 = 21$  parameters for Region and  $K - 1 = 1$  parameter for VehGas.

Figure 5.3 shows the empirical marginal frequencies  $\bar{\lambda} = \sum N_i / \sum v_i$  on all levels of the categorical feature components VehBrand, Region and VehGas. Moreover, the blue areas (in the colored version) give confidence bounds of  $\pm 2\sqrt{\bar{\lambda} / \sum v_i}$  (under a Poisson assumption), see Example 3.22. The more narrow these confidence bounds, the bigger the volumes  $\sum v_i$  behind these empirical marginal estimates.



**Fig. 5.3** Empirical marginal frequencies on each level of the categorical variables (lhs) VehBrand, (middle) Region, and (rhs) VehGas

## Continuous Variables

We consider feature engineering of the continuous variables `Area`, `VehPower`, `VehAge`, `DriveAge`, `BonusMalus` and `log-Density` (`Density` on the log scale); note that we map the `Area` codes  $(A, \dots, F) \mapsto (1, \dots, 6)$ . Some of these variables do not show any monotonicity nor log-linearity in the empirical marginal frequency plots, see Fig. 5.4.

These non-monotonicity and non-log-linearity suggest in a first step to build homogeneous classes for these feature components and use dummy coding for the resulting classes. We make the following choices here (motivated by the marginal graphs of Fig. 5.4):

- `Area`: continuous log-linear feature component for  $\{A, \dots, F\} \mapsto \{1, \dots, 6\}$ ;
- `VehPower`: discretize into categorical classes where we merge vehicle power groups bigger and equal to 9 (totally  $K = 6$  levels);
- `VehAge`: we build categorical classes  $[0, 6)$ ,  $[6, 13)$ ,  $[13, \infty)$  (totally  $K = 3$  levels);
- `DriveAge`: we build categorical classes  $[18, 21)$ ,  $[21, 26)$ ,  $[26, 31)$ ,  $[31, 41)$ ,  $[41, 51)$ ,  $[51, 71)$ ,  $[71, \infty)$  (totally  $K = 7$  levels);
- `BonusMalus`: continuous log-linear feature component (we censor at 150);
- `Density`: log-density is chosen as continuous log-linear feature component.

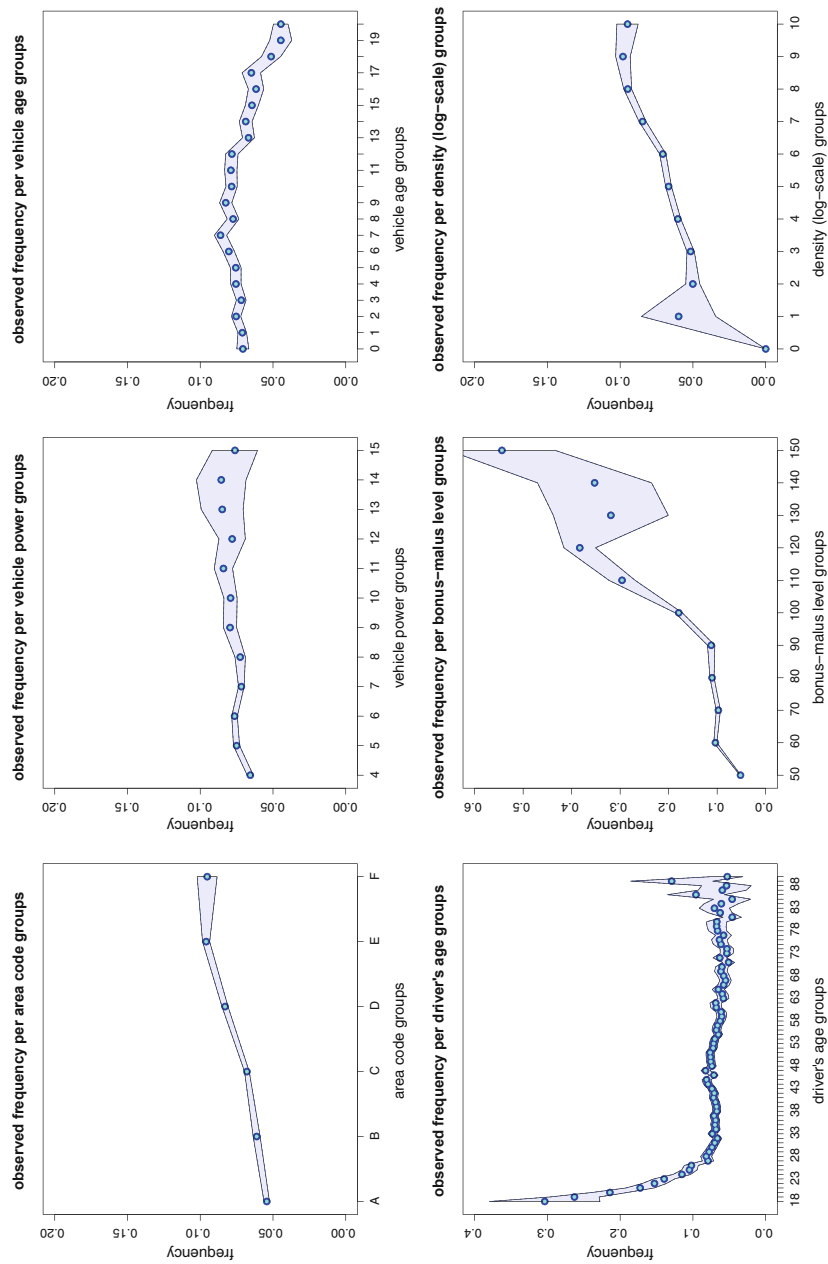
This encoding is slightly different from Noll et al. [287] because of different data cleaning. The discretization has been chosen quite ad-hoc by just looking at the empirical plots; as illustrated in Section 6.1.6 of Wüthrich–Buser [392] regression trees may provide an algorithmic way of choosing homogeneous classes of sufficient volume. This provides us with a feature space (the initial component stands for the intercept  $x_{i,0} = 1$  and the order of the terms is the same as in Listing 13.2)

$$\mathcal{X} \subset \{1\} \times \mathbb{R} \times \{0, 1\}^5 \times \{0, 1\}^2 \times \{0, 1\}^6 \times \mathbb{R} \times \{0, 1\}^{10} \times \{0, 1\} \times \mathbb{R} \times \{0, 1\}^{21},$$

of dimension  $q + 1 = 1 + 1 + 5 + 2 + 6 + 1 + 10 + 1 + 1 + 21 = 49$ . The R code [307] for this pre-processing of continuous variables is shown in Listing 5.1, categorical variables do not need any special treatment because variables of `factor` type are considered internally in R by dummy coding; we call this model Poisson GLM1.

## Choice of Learning and Test Samples

To measure predictive performance we follow the generalization approach as proposed in Chap. 4. This requires that we partition our entire data into learning sample  $\mathcal{L}$  and test sample  $\mathcal{T}$ , see Fig. 4.1. Model selection and model fitting will be done on the learning sample  $\mathcal{L}$ , only, and the test sample  $\mathcal{T}$  is used to analyze the generalization of the fitted models to unseen data. We partition the data at random (non-stratified) in a ratio of 9 : 1, and we are going to hold on to the same partitioning throughout this monograph whenever we study this example. The R code used is given in Listing 5.2.



**Fig. 5.4** Empirical marginal frequencies of the continuous variables: top row (lhs) Area, (middle) VehPower, and bottom row (lhs) DriveAge, (middle) BonusMalus, (rhs) log-Density, i.e., Density on the log scale; note that DriveAge and BonusMalus have a different y-scale in these plots

**Listing 5.1** Pre-processing of features for model Poisson GLM1 in R

---

```

1 dat$AreaGLM      <- as.integer(dat$Area)
2 dat$VehPowerGLM  <- as.factor(pmin(dat$VehPower, 9))
3 dat$VehAgeGLM    <- as.factor(cut(dat$VehAge, c(0,5,12,101),
4                                labels = c("0-5", "6-12", "12+"),
5                                include.lowest = TRUE))
6 dat$DrivAgeGLM   <- as.factor(cut(dat$DrivAge, c(18,20,25,30,40,50,70,101),
7                                labels = c("18-20", "21-25", "26-30", "31-40", "41-50",
8                                "51-70", "71+"), include.lowest = TRUE))
9 dat$BonusMalusGLM <- pmin(dat$BonusMalus, 150)
10 dat$DensityGLM   <- log(dat$Density)

```

---

Table 5.2 shows the summary of the chosen partition into learning and test samples

$$\mathcal{L} = \{(Y_i = N_i/v_i, \mathbf{x}_i, v_i) : i = 1, \dots, n = 610'206\},$$

and

$$\mathcal{T} = \{(Y_t^\dagger = N_t^\dagger/v_t^\dagger, \mathbf{x}_t^\dagger, v_t^\dagger) : t = 1, \dots, T = 67'801\}.$$

In contrast to Sect. 4.2 we also include feature information and exposure information to  $\mathcal{L}$  and  $\mathcal{T}$ .

**Listing 5.2** Partition of the data to learning sample  $\mathcal{L}$  and test sample  $\mathcal{T}$ 


---

```

1 RNGversion("3.5.0")      # we use R version 3.5.0 for this partition
2 set.seed(500)
3 ll  <- sample(c(1:nrow(dat)), round(0.9*nrow(dat)), replace = FALSE)
4 learn <- dat[ll,]
5 test  <- dat[-ll,]

```

---

**Table 5.2** Choice of learning data set  $\mathcal{L}$  and test data set  $\mathcal{T}$ ; the empirical frequency on both data sets is similar (last column), and the split of the policies w.r.t. the numbers of claims is also rather similar

	Numbers of observed claims						Empirical frequency
	0	1	2	3	4	5	
Learning sample $\mathcal{L}$	96.32%	3.47%	0.19%	0.01%	0.0006%	0.0002%	7.36%
Test sample $\mathcal{T}$	96.31%	3.50%	0.18%	0.01%	0.0015%	0.0015%	7.35%

## Maximum-Likelihood Estimation and Results

The remaining step is to perform MLE to estimate regression parameter  $\beta \in \mathbb{R}^{q+1}$ . This can be done either by maximizing the Poisson log-likelihood function or by minimizing the Poisson deviance loss. In view of (4.9) and Example 4.27, the Poisson deviance loss on the learning data  $\mathcal{L}$  is given by

$$\beta \mapsto \mathfrak{D}(\mathcal{L}, \beta) = \frac{2}{n} \sum_{i=1}^n v_i \left( \mu(\mathbf{x}_i) - Y_i - Y_i \log \left( \frac{\mu(\mathbf{x}_i)}{Y_i} \right) \right) \geq 0, \quad (5.28)$$

where the terms under the summation are set equal to  $v_i \mu(\mathbf{x}_i)$  for  $Y_i = 0$ , see (4.8), and we have GLM regression function

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = \mu_\beta(\mathbf{x}) = \exp\langle \beta, \mathbf{x} \rangle.$$

That is, we work under the canonical link with the canonical parameter being equal to the linear predictor. The MLE of  $\beta$  is found by minimizing (5.28). This is done with Fisher's scoring method. In order to receive a non-degenerate solution we need to ensure that we have sufficiently many claims  $Y_i > 0$ , otherwise it might happen that the MLE provides a (degenerate) solution at the boundary of the effective domain  $\Theta$ . We denote the MLE by  $\hat{\beta}_{\mathcal{L}}^{\text{MLE}} = \hat{\beta}^{\text{MLE}}$ , because it has been estimated on the learning data  $\mathcal{L}$ , only. This gives us estimated regression function

$$\mathbf{x} \mapsto \hat{\mu}(\mathbf{x}) = \mu_{\hat{\beta}^{\text{MLE}}}(\mathbf{x}) = \exp\langle \hat{\beta}_{\mathcal{L}}^{\text{MLE}}, \mathbf{x} \rangle.$$

We emphasize that we only use the learning data  $\mathcal{L}$  for this model fitting. In view of Definition 4.24 we receive in-sample and out-of-sample Poisson deviance losses

$$\begin{aligned} \mathfrak{D}(\mathcal{L}, \hat{\beta}_{\mathcal{L}}^{\text{MLE}}) &= \frac{2}{n} \sum_{i=1}^n v_i \left( \hat{\mu}(\mathbf{x}_i) - Y_i - Y_i \log \left( \frac{\hat{\mu}(\mathbf{x}_i)}{Y_i} \right) \right) \geq 0, \\ \mathfrak{D}(\mathcal{T}, \hat{\beta}_{\mathcal{L}}^{\text{MLE}}) &= \frac{2}{T} \sum_{t=1}^T v_t^\dagger \left( \hat{\mu}(\mathbf{x}_t^\dagger) - Y_t^\dagger - Y_t^\dagger \log \left( \frac{\hat{\mu}(\mathbf{x}_t^\dagger)}{Y_t^\dagger} \right) \right) \geq 0. \end{aligned}$$

We implement this GLM on the data of Listing 5.1 (and including the categorical features) in R using the function `glm` [307], a short overview of the results is presented in Listing 5.3. This overview presents the regression model implemented, an excerpt of the parameter estimates  $\hat{\beta}_{\mathcal{L}}^{\text{MLE}}$ , standard errors which are received from the square-rooted diagonal entries of the inverse of the estimated Fisher's information matrix  $\mathcal{I}_n(\hat{\beta}_{\mathcal{L}}^{\text{MLE}})$ , see (5.17); the remaining columns will be described in Sect. 5.3.2 on the Wald test (5.33). The bottom line of the output says that Fisher's scoring algorithm has converged in 6 iterations, it gives the in-sample deviance loss  $n\mathfrak{D}(\mathcal{L}, \hat{\beta}_{\mathcal{L}}^{\text{MLE}})$  called `Residual deviance` (not being scaled by the number of

**Listing 5.3** Results in model Poisson GLM1 using the R command glm

```
1 Call:
2 glm(formula = ClaimNb ~ VehPowerGLM + VehAgeGLM + DrivAgeGLM +
3     BonusMalusGLM + VehBrand + VehGas + DensityGLM + Region +
4     AreaGLM, family = poisson(), data = learn, offset = log(Exposure))
5
6 Deviance Residuals:
7     Min       1Q   Median       3Q      Max
8 -1.4728  -0.3256  -0.2456  -0.1383   7.7971
9
10 Coefficients:
11             Estimate Std. Error z value Pr(>|z|)
12 (Intercept)  -4.8175439   0.0579296  -83.162  < 2e-16 ***
13 VehPowerGLM5   0.0604293   0.0229841    2.629  0.008559 **
14 VehPowerGLM6   0.0868252   0.0225509    3.850  0.000118 ***
15 .              .              .          .
16 .              .              .          .
17 RegionR93      0.1388160   0.0294901    4.707  2.51e-06 ***
18 RegionR94      0.1918538   0.0938250    2.045  0.040874 *
19 AreaGLM        0.0407973   0.0200818    2.032  0.042199 *
20 ---
21 Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
22
23 (Dispersion parameter for poisson family taken to be 1)
24
25 Null deviance: 153852  on 610205  degrees of freedom
26 Residual deviance: 147069  on 610157  degrees of freedom
27 AIC: 192818
28
29 Number of Fisher Scoring iterations: 6
```

**Table 5.3** Run times, number of parameters, AICs, in-sample and out-of-sample deviance losses, tenfold cross-validation losses with empirical standard deviation in brackets, see also (4.36), (units are in  $10^{-2}$ ) and the in-sample average frequency of the null model (Poisson intercept model, see Example 4.27) and of model Poisson GLM1

	Run time	# Param.	AIC	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Tenfold CV loss $\widehat{\mathcal{D}}^{CV}$	Aver. freq.
Poisson null	–	1	199’506	25.213	25.445	25.213(0.234)	7.36%
Poisson GLM1	16 s	49	192’818	24.101	24.146	24.121(0.245)	7.36%

observations), as well as Akaike’s Information Criterion (AIC), see Sect. 4.2.3 for AIC. Note that we have implemented Poisson version (5.27) with the exposures entering the offset, see lines 2–4 of Listing 5.3; this is important for understanding AIC being calculated on the (unscaled) claim counts  $N_i$ .

Table 5.3 summarizes the results of model Poisson GLM1 and it compares the figures to the null model (only having an intercept  $\beta_0$ ); the null model has already been introduced in Example 4.27. We present the run time needed to fit the model,<sup>3</sup> the number of regression parameters  $q + 1$  in  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ , AIC, in-sample and out-of-sample deviance losses, as well as tenfold cross-validation losses on the

<sup>3</sup> All run times are measured on a personal laptop Intel(R) Core(TM) i7-8550U CPU @ 1.80 GHz 1.99 GHz with 16 GB RAM, and they only correspond to fitting the model (or the corresponding step) once, i.e., they do not account for multiple runs, for instance, for  $K$ -fold cross-validation.

learning data  $\mathcal{L}$ . For tenfold cross-validation we always use the same (non-stratified) partition of  $\mathcal{L}$  (in all examples in this monograph), and in bracket we show the empirical standard deviation received by (4.36). Tenfold cross-validation would not be necessary in this case because we have test data  $\mathcal{T}$  on which we can evaluate the out-of-sample deviance GL. We present both figures to back-test whether tenfold cross-validation works properly in our example. We observe that the out-of-sample deviance losses  $\mathfrak{D}(\mathcal{T}, \hat{\beta}_{\mathcal{L}}^{\text{MLE}})$  are within one empirical standard deviation of the tenfold cross-validation losses  $\hat{\mathfrak{D}}^{\text{CV}}$ , which supports this methodology of model comparison.

From Table 5.3 we conclude that we should prefer model Poisson GLM1 over the null model, this decision is supported by a smaller AIC, a smaller out-of-sample deviance loss  $\mathfrak{D}(\mathcal{T}, \hat{\beta}_{\mathcal{L}}^{\text{MLE}})$  as well as a smaller cross-validation loss  $\hat{\mathfrak{D}}^{\text{CV}}$ . The last column of Table 5.3 confirms that the estimated model meets the balance property (we work with the canonical link here). Note that this balance property should be fulfilled for two reasons. Firstly, we would like to have the overall portfolio price on the right level, and secondly, deviance losses should only be compared on the same overall frequency, see Example 4.10.

Before we continue to introduce more models to challenge model Poisson GLM1, we are going to discuss statistical tools for model evaluation. Of course, we would like to know whether model Poisson GLM1 is a good model for this data or whether it is just the better model of two bad options.

*Remark 5.15 (Prior and Posterior Information)* Pricing literature distinguishes between prior feature information and posterior feature information, see Verschuren [372]. Prior feature information is available at the inception of the (new) insurance contract before having any claims history. This includes, for instance, age of driver, vehicle brand, etc. For policy renewals, past claims history is available and prices of policy renewals can also be based on such posterior information. Past claims history has led to the development of so-called bonus-malus systems (BMS) which often are in the form of multiplicative factors to the base premium to reward and punish good and bad past experience, respectively. One stream of literature studies optimal designs of BMS, we refer to Loimaranta [255], De Pril [91], Lemaire [245], Denuit et al. [102], Brouhns et al. [57] Pinquet [304], Pinquet et al. [305], Tzougas et al. [360] or Ágoston–Gyetvai [4]. Another stream of literature studies how one can optimally extract predictive information from an existing BMS, see Boucher–Inoussa [46], Boucher–Pigeon [47] and Verschuren [372].

The latter is basically what we also do in the above example: note that we include the variable `BonusMalus` into the feature information and, thus, we use past claims information to predict future claims. For new policies, the bonus-malus level is at 100%, and our information does not allow to clearly distinguish between new

policies and policy renewals for drivers that have posterior information reflected by a bonus-malus level of 100%. Since young drivers are more likely new customers we expect interactions between the driver's age variable and the bonus-malus level, this intuition is supported by Fig. 13.12 (lhs). In order to improve our model, we would require more detailed information about past claims history. Remark that we do not strictly distinguish between prior and posterior information, here. If we go over to a time-series consideration, where more and more claims experience becomes available of an individual driver, we should clearly distinguish the different sets of information, because otherwise it may happen that in prior and posterior pricing factors we correct twice for the same factor; an interesting paper is Corradin et al. [82].

We also mention that a new source of posterior information is emerging through the collection of telematics car driving data. Telematics car driving data leads to a completely new way of posterior information rate making (experience rating), we refer to Ayuso et al. [17–19], Boucher et al. [42], Lemaire et al. [246] and Denuit et al. [98]. We mention the papers of Gao et al. [152, 154] and Meng et al. [271] who directly extract posterior feature information from telematics car driving data in order to improve rate making. This approach combines a Poisson GLM with a network extractor for the telematics car driving data.

## 5.3 Model Validation

One of the purposes of Chap. 4 has been to describe measures to analyze how well a fitted model generalizes to unseen data. In a proper generalization analysis this requires learning data  $\mathcal{L}$  for in-sample model fitting and a test sample  $\mathcal{T}$  for an out-of-sample generalization analysis. In many cases, one is not in the comfortable situation of having a test sample. In such situations one can use AIC that tries to correct the in-sample figure for model complexity or, alternatively,  $K$ -fold cross-validation as used in Table 5.3.

The purpose of this section is to introduce diagnostic tools for fitted models; these are often based on unit deviances  $\mathfrak{d}(Y_i, \mu_i)$ , which play the role of squared residuals in classical linear regression. Moreover, we discuss parameter and model selection, for instance, by step-wise backward elimination or forward selection using the analysis of variance (ANOVA) or the likelihood ratio test (LRT).

### 5.3.1 Residuals and Dispersion

Within the EDF we distinguish two different types of residuals. The first type of residuals are based on the unit deviances  $\mathfrak{d}(Y_i, \mu_i)$  studied in (4.7). The *deviance*



*residuals* are given by

$$r_i^D = \text{sign}(Y_i - \mu_i) \sqrt{\frac{v_i}{\varphi}} \mathfrak{d}(Y_i, \mu_i).$$

Secondly, *Pearson's residuals* are given by, see also (4.12),

$$r_i^P = \sqrt{\frac{v_i}{\varphi}} \frac{Y_i - \mu_i}{\sqrt{V(\mu_i)}}.$$

In the Gaussian case the two residuals coincide. This indicates that Pearson's residuals are most appropriate in the Gaussian case because they respect the distributional properties in that case. For other distributions, Pearson's residuals can be markedly skewed, as stated in Section 2.4.2 of McCullagh–Nelder [265], and therefore may fail to have properties similar to Gaussian residuals. An other issue occurs in Pearson's residuals when the denominator involves an estimated standard deviation  $\sqrt{V(\hat{\mu}_i)}$ , for instance, if we work in a small frequency Poisson problem. Estimation uncertainty in small denominators of Pearson's residuals may substantially distort the estimated residuals. For this reason, we typically work with (the more robust) deviance residuals; this is related to the discussion in Chap. 4 on MSEPs versus expected deviance GLs, see Remarks 4.6.

The squared residuals provide unit deviance and weighted square loss, respectively,

$$(r_i^D)^2 = \frac{v_i}{\varphi} \mathfrak{d}(Y_i, \mu_i) \quad \text{and} \quad (r_i^P)^2 = \frac{v_i}{\varphi} \frac{(Y_i - \mu_i)^2}{V(\mu_i)},$$

the latter corresponds to Pearson's  $\chi^2$ -statistic, see (4.12).

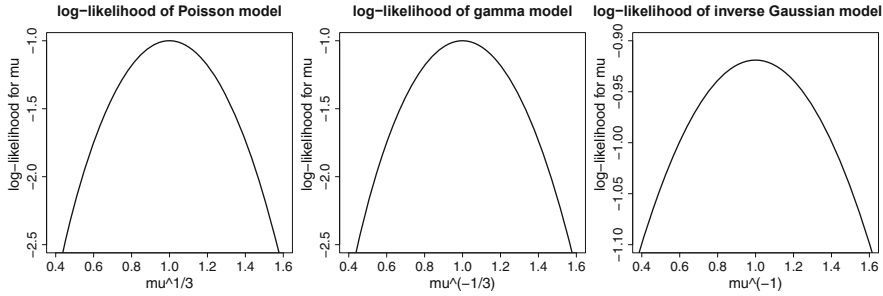
*Example 5.16 (Residuals in the Poisson Case)* In the Poisson case, Pearson's  $\chi^2$ -statistic is for  $v_i = \varphi = 1$  given by

$$(r_i^P)^2 = \frac{(Y_i - \mu_i)^2}{\mu_i},$$

because we have variance function  $V(\mu) = \mu$ . A second order Taylor expansion around  $Y_i$  on the scale  $\mu_i^{1/3}$  (for  $\mu_i$ ) provides approximation to the unit deviances in the Poisson case, see formula (6.4) and Figure 6.2 in McCullagh–Nelder [265],

$$\mathfrak{d}(Y_i, \mu_i) \approx 9Y_i^{1/3} \left( Y_i^{1/3} - \mu_i^{1/3} \right)^2. \quad (5.29)$$

This emphasizes the different behaviors around the observation  $Y_i$  of the two types of residuals in the Poisson case. The scale  $\mu_i^{1/3}$  has been motivated in McCullagh–



**Fig. 5.5** Log-likelihoods  $\ell_Y(\mu)$  in  $Y = 1$  as a function of  $\mu$  plotted against (lhs)  $\mu^{1/3}$  in the Poisson case, (middle)  $\mu^{-1/3}$  in the gamma case with shape parameter  $\alpha = 1$ , and (rhs)  $\mu^{-1}$  in the inverse Gaussian case with  $\alpha = 1$

Nelder [265] by providing a symmetric behavior around the mode in  $Y_i = 1$  of the resulting log-likelihood function, see Fig. 5.5 (lhs). ■

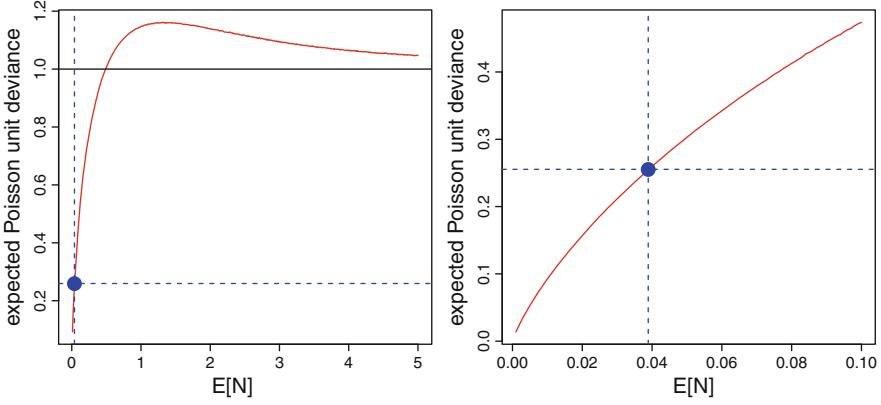
The explicit calculation of the residuals requires knowledge of the dispersion parameter  $\varphi > 0$ . In the Poisson Example 5.16 this dispersion parameter has been set equal to 1 because the Poisson model does neither allow for under- nor for over-dispersion. Typically, this is not the case for other models, and this requires determination of the dispersion parameter if we want to simulate from these other models. So far, this dispersion parameter has been treated as a nuisance parameter and, in fact, it canceled in MLE (because it was assumed to be constant), see Proposition 5.1.

If we need to estimate the dispersion parameter, we can either do this within MLE, see Remarks 5.2, or we can use Pearson's or the deviance estimates, respectively,

$$\hat{\varphi}^P = \frac{1}{n - (q + 1)} \sum_{i=1}^n \frac{(Y_i - \hat{\mu}_i)^2}{V(\hat{\mu}_i)/v_i} \quad \text{and} \quad \hat{\varphi}^D = \frac{1}{n - (q + 1)} \sum_{i=1}^n v_i \mathfrak{d}(Y_i, \hat{\mu}_i), \quad (5.30)$$

where  $\hat{\mu}_i = \hat{\mu}(\mathbf{x}_i)$  are the MLE estimated means involving  $q + 1$  estimated parameters  $\hat{\boldsymbol{\beta}}^{\text{MLE}} \in \mathbb{R}^{q+1}$ . We briefly motivate these choices. Firstly, Pearson's estimate  $\hat{\varphi}^P$  is consistent for  $\varphi$ . Note that in the Gaussian case this is just the standard estimate for the variance parameter. Justification of the deviance dispersion estimate is more challenging. Consider the unscaled deviance with  $\hat{\boldsymbol{\mu}}_n = (\hat{\mu}_1, \dots, \hat{\mu}_n)^\top$ , see (4.9),

$$n\varphi \mathfrak{D}(Y_n, \hat{\boldsymbol{\mu}}_n) = \sum_{i=1}^n v_i \mathfrak{d}(Y_i, \hat{\mu}_i).$$



**Fig. 5.6** Expected unit deviance  $v\mathbb{E}_\mu[\mathfrak{d}(Y, \mu)]$  in the Poisson case as a function of  $\mathbb{E}[N] = \mathbb{E}[vY] = v\mu$ ; the two plots only differ in the scale on the  $x$ -axis

This statistic is under *certain* assumptions asymptotically  $\varphi \chi_{n-(q+1)}^2$ -distributed, where  $\chi_{n-(q+1)}^2$  denotes a  $\chi^2$ -distribution with  $n - (q + 1)$  degrees of freedom. Thus, this approximation gives us an expected value of  $\varphi(n - (q + 1))$ . This exactly justifies the deviance dispersion estimate (5.30) in these cases. However, as stated in the last paragraph of Section 2.3 of McCullagh–Nelder [265], often a  $\chi^2$ -approximation is not suitable even as  $n \rightarrow \infty$ . We give an example.

*Example 5.17 (Poisson Unit Deviances)* The deviance statistics in the Poisson model with means  $\boldsymbol{\mu}_n = (\mu_1, \dots, \mu_n)^\top$  is given by

$$\mathfrak{D}(\mathbf{Y}_n, \boldsymbol{\mu}_n) = \frac{1}{n} \sum_{i=1}^n v_i \mathfrak{d}(Y_i, \mu_i) = \frac{1}{n} \sum_{i=1}^n 2v_i \left( \mu_i - Y_i - Y_i \log \left( \frac{\mu_i}{Y_i} \right) \right),$$

note that in the Poisson model we have (by definition)  $\varphi = 1$ . We evaluate the expected value of this deviance statistics. It is given by

$$\mathbb{E}_{\boldsymbol{\mu}_n}[\mathfrak{D}(\mathbf{Y}_n, \boldsymbol{\mu}_n)] = \frac{1}{n} \sum_{i=1}^n 2v_i \mathbb{E}_{\mu_i} \left[ \mu_i - Y_i - Y_i \log \left( \frac{\mu_i}{Y_i} \right) \right] = \frac{1}{n} \sum_{i=1}^n 2\mathbb{E}_{\mu_i} \left[ N_i \log \left( \frac{N_i}{v_i \mu_i} \right) \right],$$

with  $N_i \stackrel{\text{ind.}}{\sim} \text{Poi}(v_i \mu_i)$ .

In Fig. 5.6 we plot the expected unit deviance  $v\mu \mapsto v\mathbb{E}_\mu[\mathfrak{d}(Y, \mu)]$  in the Poisson model. In our example of Table 5.3, we have  $\mathbb{E}_\mu[vY] = v\mu \approx 3.89\%$ , which results in an expected unit deviance of  $v\mathbb{E}_\mu[\mathfrak{d}(Y, \mu)] \approx 25.52 \cdot 10^{-2} < 1$ . This is in line with the losses in Table 5.3. Thus, the expected deviance  $n\mathbb{E}_{\boldsymbol{\mu}_n}[\mathfrak{D}(\mathbf{Y}_n, \boldsymbol{\mu}_n)] \approx n/4 < n$ . Therefore it is substantially smaller than  $n$ . But this implies that  $n\mathfrak{D}(\mathbf{Y}_n, \boldsymbol{\mu}_n)$  cannot be asymptotically  $\chi_{n-(q+1)}^2$ -distributed because the latter has an expected of value  $n - (q + 1) \approx n$  for  $n \rightarrow \infty$ . In fact, the deviance dispersion estimate is not consistent

in this example, and for a consistent estimate one should rely on Pearson's deviance estimate.

In order to have an asymptotic  $\chi^2$ -distribution we need to have large volumes  $v$  because then a saddlepoint approximation holds that allows to approximate the (scaled) unit deviances by  $\chi^2$ -distributions, see Sect. 5.5.2, below. ■

### 5.3.2 Hypothesis Testing

Consider a sub-vector  $\beta_r \in \mathbb{R}^r$  of the GLM parameter  $\beta \in \mathbb{R}^{q+1}$ , for  $r < q + 1$ . We would like to understand if we can set this sub-vector  $\beta_r = 0$ , and at the same time we do not lose any generalization power. Thus, we investigate whether there is a simpler *nested* GLM that provides a similar prediction accuracy. If this is the case, preference should be given to the simpler model because the bigger model seems over-parametrized (has redundancy, is not parsimonious). This section is based on Section 2.2.2 of Fahrmeir–Tutz [123].

**Geometric Interpretation** We begin by giving a geometric interpretation. We start from the full model being expressed by the design matrix  $\mathfrak{X} \in \mathbb{R}^{n \times (q+1)}$ . This design matrix together with the link function  $g$  generates a  $(q + 1)$ -dimensional manifold  $\mathfrak{M} \subset \mathbb{R}^n$  given by, see (5.19) and Fig. 5.2,

$$\mathfrak{M} = \left\{ \mu = g^{-1}(\mathfrak{X}\beta) = (g^{-1}(\beta, x_1), \dots, g^{-1}(\beta, x_n))^T \in \mathbb{R}^n \mid \beta \in \mathbb{R}^{q+1} \right\} \subset \mathbb{R}^n.$$

The MLE  $\hat{\beta}^{\text{MLE}}$  is determined by the point in  $\mathfrak{M}$  that minimizes the distance to  $Y$ , where distance between  $Y$  and  $\mathfrak{M}$  is measured component-wise by  $\frac{\psi_i}{\varphi}(Y_i, \mu_i)$  with  $\mu \in \mathfrak{M}$ , i.e., w.r.t. the KL divergence.

Assume, now, that we want to drop the components  $\beta_r$  in  $\beta$ , i.e., we want to drop these columns from the design matrix resulting in a smaller design matrix  $\mathfrak{X}_r \in \mathbb{R}^{n \times (q+1-r)}$ . This generates a  $(q + 1 - r)$ -dimensional *nested* manifold  $\mathfrak{M}_r \subset \mathfrak{M}$  described by

$$\mathfrak{M}_r = \left\{ \mu = g^{-1}(\mathfrak{X}_r\beta) \in \mathbb{R}^n \mid \beta \in \mathbb{R}^{q+1-r} \right\} \subset \mathfrak{M}.$$

If the distance of  $Y$  to  $\mathfrak{M}_r$  and  $\mathfrak{M}$  is roughly the same, we should go for the smaller model. In the Gaussian case of Example 5.9 this can be explained by the Pythagorean theorem applied to successive orthogonal projections. In the general unit deviance case, this has to be studied in terms of information geometry considering the KL divergence, see Sect. 2.3.

**Likelihood Ratio Test (LRT)** We consider the testing problem of the null hypothesis  $H_0$  against the alternative hypothesis  $H_1$

$$H_0 : \beta_r = 0 \quad \text{against} \quad H_1 : \beta_r \neq 0. \quad (5.31)$$

Denote by  $\hat{\beta}^{\text{MLE}}$  the MLE under the full model and by  $\hat{\beta}_{(-r)}^{\text{MLE}}$  the MLE under the null hypothesis model. Define the (log-)likelihood ratio test (LRT) statistics

$$\Lambda = -2 \left( \ell_Y(\hat{\beta}_{(-r)}^{\text{MLE}}) - \ell_Y(\hat{\beta}^{\text{MLE}}) \right) \geq 0.$$

The inequality holds because the null hypothesis model is nested in the full model, henceforth, the latter needs to have a bigger log-likelihood value in the MLE. If the LRT statistics  $\Lambda$  is large, the null hypothesis should be rejected because the reduced model is not competitive compared to the full model. More mathematically, under similar conditions as for the asymptotic normality results of the MLE of  $\beta$  in (5.17), we have that under the null hypothesis  $H_0$  the LRT statistics  $\Lambda$  is asymptotically  $\chi^2$ -distributed with  $r$  degrees of freedom. Therefore, we should reject the null hypothesis in favor of the full model if the resulting  $p$ -value of  $\Lambda$  under the  $\chi_r^2$ -distribution is too small. These results remain true if the unknown dispersion parameter  $\varphi$  is replaced by a consistent estimator  $\hat{\varphi}$ , e.g., Pearson's dispersion estimate  $\hat{\varphi}^{\text{P}}$  (from the bigger model).

The LRT statistics  $\Lambda$  may not be properly defined in over-dispersed situations where the distributional assumptions are not fully specified, for instance, in an over-dispersed Poisson model. In such situations, one usually divides the log-likelihood (of the Poisson model) by the estimated over-dispersion and then uses the resulting scaled LRT statistics  $\Lambda$  as an approximation to the unspecified model.

**Wald Test** Alternatively, we can use the Wald statistics. The Wald statistics uses a second order approximation to the log-likelihood and, therefore, is only based on the first two moments (and not on the entire distribution). Define the matrix  $I_r \in \mathbb{R}^{r \times (q+1)}$  such that  $\beta_r = I_r \beta$ , i.e., matrix  $I_r$  selects exactly the components of  $\beta$  that are included in  $\beta_r$  (and which are set to 0 under the null hypothesis  $H_0$  given in (5.31)).

Asymptotic normality (5.17) motivates consideration of the Wald statistics

$$W = (I_r \hat{\beta}^{\text{MLE}} - 0)^\top \left( I_r \mathcal{I}(\hat{\beta}^{\text{MLE}})^{-1} I_r^\top \right)^{-1} (I_r \hat{\beta}^{\text{MLE}} - 0). \quad (5.32)$$

The Wald statistics measures the distance between the MLE in the full model  $I_r \hat{\beta}^{\text{MLE}}$  restricted to the components of  $\beta_r$  and the null hypothesis  $H_0$  (being  $\beta_r = 0$ ). The estimated Fisher's information matrix  $\mathcal{I}(\hat{\beta}^{\text{MLE}})$  is used to bring all components onto the same unit scale (and to account for collinearity). The Wald statistics  $W$  is asymptotically  $\chi_r^2$ -distributed under the same assumptions as for (5.17) to hold. Thus, the null hypothesis  $H_0$  should be rejected if the resulting  $p$ -

value of  $W$  under the  $\chi_r^2$ -distribution is too small. Note that this test does not require calculation of the MLE in the null hypothesis model, i.e., this test is computationally more attractive than the LRT because we only need to fit one model. Again, an unknown dispersion parameter  $\varphi$  in Fisher's information matrix  $\mathcal{I}(\boldsymbol{\beta})$  is replaced by a consistent estimator  $\hat{\varphi}$  (from the bigger model).

In the special case of considering only one component of  $\boldsymbol{\beta}$ , i.e., if  $\boldsymbol{\beta}_r = \beta_k$  with  $r = 1$  and for one selected component  $0 \leq k \leq q$ , the Wald statistics reduces to

$$W_k = \frac{(\hat{\beta}_k^{\text{MLE}})^2}{\hat{\sigma}_k^2} \quad \text{or} \quad T_k = W_k^{1/2} = \frac{\hat{\beta}_k^{\text{MLE}}}{\hat{\sigma}_k}, \quad (5.33)$$

with diagonal entries of the inverse of the estimated Fisher's information matrix given by  $\hat{\sigma}_k^2 = (\mathcal{I}(\hat{\boldsymbol{\beta}}^{\text{MLE}})^{-1})_{k,k}$ ,  $0 \leq k \leq q$ . The square-roots of these estimates are provided in column `Std. Error` of the R output in Listing 5.3.

In this case the Wald statistics  $W_k$  is equal to the square of the  $t$ -statistics  $T_k$ ; this  $t$ -statistics is provided in column `z value` of the R output of Listing 5.3. Remark that Fisher's information matrix involves the dispersion parameter  $\varphi$ . If this dispersion parameter is estimated with a consistent estimator  $\hat{\varphi}$  we have a  $t$ -statistics. For known dispersion parameter the  $t$ -statistics reduces to a  $z$ -statistics, i.e., the corresponding  $p$ -values can be calculated from a normal distribution instead of a  $t$ -distribution. In the Poisson case, the dispersion  $\varphi = 1$  is known, and for this reason, we perform a  $z$ -test (and not a  $t$ -test) in the last column of Listing 5.3; and we call  $T_k$  a  $z$ -statistics in that case.

### 5.3.3 Analysis of Variance

In the previous section, we have presented tests that allow for model selection in the case of nested models. More generally, if we have a full model, say, based on regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$  we would like to select the “best” sub-model according to some selection criterion. In most cases, it is computationally not feasible to fit all sub-models if  $q$  is large, therefore, this is not a practical solution. For large models and data sets step-wise procedures are a feasible tool. *Backward elimination* starts from the full model, and then recursively drops feature components which have high  $p$ -values in the corresponding Wald statistics (5.32) and (5.33). Performing this recursively will provide us with hierarchy of nested models. *Forward selection* works just in the opposite direction, that is, we start with the null model and we include feature components one after the other that have a low  $p$ -value in the corresponding Wald statistics.

*Remarks 5.18*

- The order of the inclusion/exclusion of the feature components matters in this selection algorithms because we do not have additivity in this selection process. For this reason, often backward elimination and forward selection is combined in an alternating way.
- This process as well as the tests from Sect. 5.3.2 are based on a fixed pre-processing of features. If the feature pre-processing is done differently, all analysis needs to be repeated for this new model. Moreover, between two different models we need to apply different tools for model selection (if they are not nested), for instance, AIC, cross-validation or an out-of-sample generalization analysis.
- For categorical variables with dummy coding we should apply the forward selection or the backward elimination simultaneously on the entire dummy coded vector of a categorical variable. This will include or exclude this variable; if we only apply the Wald test to one component of the dummy vector, then we test whether this level should be merged with the reference level.

Typically, in practice, a so-called analysis of variance (ANOVA) table is studied. The ANOVA table is mainly motivated by the Gaussian model with orthogonal data. The Gaussian assumption implies that the deviance loss is equal to the square loss and the orthogonality implies that the square loss decouples in an additive way w.r.t. the feature components. This implies that one can explicitly study the contribution of each feature component to the decrease in square loss; an example is given in Section 2.3.2 of McCullagh–Nelder [265]. In non-Gaussian and non-orthogonal situations one loses this additivity property and, as mentioned in Remarks 5.18, the order of inclusion matters. Therefore, for the ANOVA table we pre-specify the order in which the components are included and then we analyze the decrease of deviance loss by the inclusion of additional components.

*Example 5.19 (Poisson GLMI, Revisited)* We revisit the MTPL claim frequency example of Sect. 5.2.4 to illustrate the variable selection procedures. Based on the model presented in Listing 5.3 we run an ANOVA analysis using the R command `anova`, the results are presented in Listing 5.4.

Listing 5.4 shows the hierarchy of models starting from the null model by sequentially including feature components one by one. The column `Df` gives the number of regression parameters involved and the column `Deviance` the decrease of deviance loss by the inclusion of this feature component. The biggest model improvements are provided by the bonus-malus level and driver's age, this is not surprising in view of the empirical analysis in Figs. 5.3 and 5.4, and in Chap. 13.1. At the other end we have the `Area` code which only seems to improve the model marginally. However, this does not imply, yet, that this variable should be dropped. There are two points that need to be considered: (1) maybe feature pre-processing of `Area` has not been done in an appropriate way and the variable is not in the right functional form for the chosen link function; and (2) `Area` is the last variable included in the model in Listing 5.4 and, maybe, there are already other variables

**Listing 5.4** ANOVA table of model Poisson GLM1

---

```

1 Analysis of Deviance Table
2
3 Model: poisson, link: log
4
5 Response: ClaimNb
6
7 Terms added sequentially (first to last)
8
9
10
11      Df Deviance Resid. Df Resid. Dev
12 NULL                      610205    153852
13 VehPowerGLM      5      73.7    610200    153779
14 VehAgeGLM        2     179.7    610198    153599
15 DrivAgeGLM        6    1199.4    610192    152400
16 BonusMalusGLM    1    4300.6    610191    148099
17 VehBrand         10     240.3    610181    147859
18 VehGas            1      82.4    610180    147776
19 DensityGLM        1     512.1    610179    147264
20 Region            21     191.3    610158    147073
21 AreaGLM           1        4.1    610157    147069

```

---

that take over the role of Area in smaller models which is possible if we have correlations between the feature components. In our data, Area and Density are highly correlated. For this reason, we exchange the order of these two components and run the same analysis again, we call this model Poisson GLM1B (which of course provides the same predictive model as Poisson GLM1).

**Listing 5.5** ANOVA table of model Poisson GLM1B

---

```

1 Analysis of Deviance Table
2
3 Model: poisson, link: log
4
5 Response: ClaimNb
6
7 Terms added sequentially (first to last)
8
9
10
11      Df Deviance Resid. Df Resid. Dev
12 NULL                      610205    153852
13 VehPowerGLM      5      73.7    610200    153779
14 VehAgeGLM        2     179.7    610198    153599
15 DrivAgeGLM        6    1199.4    610192    152400
16 BonusMalusGLM    1    4300.6    610191    148099
17 VehBrand         10     240.3    610181    147859
18 VehGas            1      82.4    610180    147776
19 AreaGLM           1     505.0    610179    147271
20 Region            21     192.4    610158    147079
21 DensityGLM        1      10.1    610157    147069

```

---

Listing 5.5 shows the ANOVA table if we exchange the order of these two variables. We observe that the magnitudes of the decrease of the deviance loss has switched between the two variables. Overall, Density seems slightly more



predictive, and we may consider dropping Area from the model, also because the correlation between Density and Area is very high.

If we want to perform backward elimination (sequentially drop one variable after the other) we can use the R command `drop1`. For small models this is doable, for larger models it is computationally demanding.

**Listing 5.6** `drop1` analysis of model Poisson GLM1

---

```

1 Single term deletions
2
3 Model:
4 ClaimNb ~ VehPowerGLM + VehAgeGLM + DrivAgeGLM + BonusMalusGLM +
5   VehBrand + VehGas + DensityGLM + Region + AreaGLM
6
7           Df Deviance      AIC      LRT   Pr(>Chi)
8 <none>                147069 192818
9 VehPowerGLM      5   147152 192892    83.4 < 2.2e-16 ***
10 VehAgeGLM       2   147283 193028   214.1 < 2.2e-16 ***
11 DrivAgeGLM      6   147603 193341   534.5 < 2.2e-16 ***
12 BonusMalusGLM  1   150970 196718 3901.5 < 2.2e-16 ***
13 VehBrand       10   147298 193027   228.9 < 2.2e-16 ***
14 VehGas         1   147213 192961   144.5 < 2.2e-16 ***
15 DensityGLM     1   147079 192826    10.1 0.001459 **
16 Region        21   147259 192967   190.7 < 2.2e-16 ***
17 AreaGLM        1   147073 192820     4.1 0.042180 *
18 ---
19 Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

---

In Listing 5.6 we present the results of this `drop1` analysis. Both, according to AIC and according to the LRT, we should keep all variables in the model. Again, Area and Density provide the smallest LRT statistics  $\Lambda$  which illustrates the high collinearity between these two variables (note that the values in Listing 5.6 are identical to the ones in Listings 5.4 and 5.5, respectively).

We conclude that in model Poisson GLM1 we should keep all feature components, and a model improvement can only be obtained by a different feature pre-processing, by a different regression function or by a different distributional model. ■

### 5.3.4 Lab: Poisson GLM for Car Insurance Frequencies, Revisited

#### Continuous Coding of Non-monotone Feature Components

We revisit model Poisson GLM1 studied in Sect. 5.2.4 for MTPL claim frequency modeling, and we consider additional competing models by using different feature pre-processing. From Example 5.19, above, we conclude that we should keep all variables in the model if we work with model Poisson GLM1.

**Table 5.4** Contingency table of observed number of policies against predicted number of policies with given claim counts `ClaimNb`

	Numbers of claims <code>ClaimNb</code>					
	0	1	2	3	4	5
Observed number of policies	587'772	21'198	1'174	57	4	1
Predicted number of policies	587'325	22'064	779	34	3	0.3

We calculate Pearson's dispersion estimate which provides  $\hat{\varphi}^P = 1.6697 > 1$ . This indicates that the model is not fully suitable for our data because in a Poisson model the dispersion parameter should be equal to 1. There may be two reasons for this over-dispersion: (1) the Poisson assumption is not appropriate because, for instance, the tail of the observations is more heavy-tailed, or (2) the Poisson assumption is appropriate but the regression function has not been chosen in a fully suitable way (maybe also due to missing feature information).

We believe that in our example the observed over-dispersion is a mixture of the two reasons (1) and (2). Surely, the regression structure can be improved since our feature pre-processing is non-optimal and since the chosen regression function only considers multiplicative interactions between the feature components (we have chosen the log-link regression function without adding interaction terms to the regression function).

Table 5.4 gives a contingency table. We observe that we have much more policies with more than 1 claim compared to what is predicted by the fitted model. As a result, a  $\chi^2$ -test rejects this Poisson model because the resulting  $p$ -value is close to 0.

In our data, we have a rather large number of policies with short exposures  $v_i$ , and further analysis suggests that these short exposures are not suitably modeled. We will not invest more time into improving the exposure modeling. As mentioned in the appendix, there seem to be a couple of issues how the exposures are displayed and how policy renewals are accounted for in this data. However, it is difficult (almost impossible) to clean the data for better exposure measures without more detailed information about the data collection process.

Our next aim is to model continuous feature components differently, if their raw form does not match the linear predictor assumption. In Poisson GLM1 we have categorized such components and then used dummy coding for the resulting classes, see Sect. 5.2.4. Alternatively, we can use different functional forms, for instance, we can use for `DrivAge` the following pre-processing

$$\text{DrivAge} \mapsto \beta_l \text{DrivAge} + \beta_{l+1} \log(\text{DrivAge}) + \sum_{j=2}^4 \beta_{l+j} (\text{DrivAge})^j. \quad (5.34)$$

**Table 5.5** Run times, number of parameters, AICs, in-sample and out-of-sample deviance losses, tenfold cross-validation losses (units are in  $10^{-2}$ ) and in-sample average frequency of the null model (intercept model) and of different Poisson GLMs

	Run time	# Param.	AIC	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Tenfold CV loss $\widehat{\mathcal{D}}^{\text{CV}}$	Aver. freq.
Poisson null	–	1	199'506	25.213	25.445	25.213	7.36%
Poisson GLM1	16s	49	192'818	24.101	24.146	24.121	7.36%
Poisson GLM2	15s	48	192'753	24.091	24.113	24.110	7.36%
Poisson GLM3	15s	50	192'716	24.084	24.102	24.104	7.36%

This replaces the  $K = 7$  categorical age classes of model Poisson GLM1 by 5 continuous functions of the variable `DrivAge`, and the number of regression parameters is reduced from  $K - 1 = 6$  to 5. We call this model Poisson GLM2.

Besides improving the modeling of the feature components we can also start to add interactions beyond the multiplicative ones. For instance, Fig. 13.12 in Chap. 13 may indicate that there is an interaction term between `BonusMalus` and `DrivAge`. New young drivers enter the bonus-malus system at level 100, and it takes some years free of accidents to reach the lowest bonus-malus level of 50. Whereas for senior drivers a bonus-malus level of 100 may indicate that they have had a bad claim experience because otherwise they would be on the lowest bonus-malus level, see also Remark 5.15. We are adding the following interaction to Poisson GLM2 and we call the resulting model Poisson GLM3

$$\beta_{l'} \text{BonusMalus} \cdot \text{DrivAge} + \beta_{l'+1} \text{BonusMalus} \cdot (\text{DrivAge})^2. \quad (5.35)$$

From Table 5.5 we observe that this leads to a further small model improvement. We mention that this model improvement can also be observed in a decrease of Pearson's dispersion estimate to  $\widehat{\varphi}^P = 1.6644$ . Noteworthy, all model selection criteria AIC, out-of-sample generalization loss and cross-validation come to the same conclusion in this example.

The tedious task of the modeler now is to find all these systematic effects and bring them in an appropriate form into the model. Here, this is still possible because we have a comparably small model. However, if we have hundreds of feature components, such a manual analysis becomes intractable. Other regression models such as network regression models should be preferred, or at least should be used to find systematic effects. But, one should also keep in mind that the (final) chosen model should be as simple as possible (parsimonious).

#### Remarks 5.20

- An advantage of GLMs is that these regression models can deal with collinearity in feature components. Nevertheless, the results should be carefully checked if the collinearity in feature components is very high. If we have a high collinearity between two feature components then we may observe large values with opposite signs in the corresponding regression parameters compensating each other. The

**Listing 5.7** drop1 analysis of model Poisson GLM2

---

```

1 Single term deletions
2
3 Model:
4 ClaimNb ~ VehPowerGLM + VehAgeGLM + DrivAge + log(DrivAge) +
5       I(DrivAge^2) + I(DrivAge^3) + I(DrivAge^4) + BonusMalusGLM +
6       VehBrand + VehGas + DensityGLM + Region + AreaGLM
7           Df Deviance    AIC    LRT  Pr(>Chi)
8 <none>             147005 192753
9 VehPowerGLM      5   147087 192825    82.4 2.671e-16 ***
10 VehAgeGLM        2   147225 192969   220.3 < 2.2e-16 ***
11 DrivAge          1   147157 192902   151.9 < 2.2e-16 ***
12 log(DrivAge)     1   147190 192935   184.8 < 2.2e-16 ***
13 I(DrivAge^2)     1   147123 192869   118.1 < 2.2e-16 ***
14 I(DrivAge^3)     1   147094 192840    89.0 < 2.2e-16 ***
15 I(DrivAge^4)     1   147071 192816    65.5 5.687e-16 ***
16 BonusMalusGLM   1   150907 196653  3902.0 < 2.2e-16 ***
17 VehBrand        10   147232 192959   226.5 < 2.2e-16 ***
18 VehGas          1   147148 192893   142.8 < 2.2e-16 ***
19 DensityGLM      1   147015 192761    10.1 0.001498 **
20 Region          21   147193 192899   188.0 < 2.2e-16 ***
21 AreaGLM         1   147009 192755     4.1 0.043123 *
22 ---
23 Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

---

resulting GLM will not be very robust, and a slight change in the observations may change these regression parameters completely. In this case one should drop one of the two highly collinear feature components. This problem may also occur if we include too many terms in functional forms like in (5.34).

- A tool to find suitable functional forms of regression functions in continuous feature components are the partial residual plots of Cook–Croos-Dabrera [80]. If we want to analyze the first feature component  $x_1$  of  $\mathbf{x}$ , we can fit a GLM to the data using the entire feature vector  $\mathbf{x}$ . The partial residuals for component  $x_1$  are defined by, see formula (8) in Cook–Croos-Dabrera [80],

$$r_i^{\text{partial}} = (Y_i - \mu(\mathbf{x}_i))g'(\mu(\mathbf{x}_i)) + \beta_1 x_{i,1} \quad \text{for } 1 \leq i \leq n,$$

where  $g$  is the chosen link function and  $g(\mu(\mathbf{x}_i)) = \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$ . These partial residuals offset the effect of feature component  $x_1$ . The partial residual plot shows  $r_i^{\text{partial}}$  against  $x_{i,1}$ . If this plot shows a linear structure then including  $x_1$  linearly is justified, and any other functional form may be detected from that plot.

## Under-Sampling and Over-Sampling

Often run times are an issue in model fitting, in particular, if we want to experiment with different models, different feature codings, etc. Under-sampling is an interesting approach that can be applied in imbalanced situations (like in our claim frequency data situation) to speed up calculations, and still receiving accurate approximations. We briefly describe under-sampling in this subsection.

Under-sampling is based on the idea that we do not need to consider all  $n = 610'206$  insurance policies for model fitting, and we can still receive accurate results. For this we select all insurance policies that have at least 1 claim; in our data these are 22'434 insurance policies, we call this data set  $\mathcal{L}_{\geq 1}^*$ . The motivation for selecting these insurance policies is that these are exactly the policies that have information about the drivers causing claims. These selected insurance policies need to be complemented with policies that do not cause any claims. We select at random (under-sample) 22'434 insurance policies of drivers without claims, we call this data set  $\mathcal{L}_0^*$ . Merging the two sets we receive data  $\mathcal{L}^* = \mathcal{L}_0^* \cup \mathcal{L}_{\geq 1}^*$  comprising 44'868 insurance policies. This data is balanced from the viewpoint of claim causing policies because exactly half of the policies in  $\mathcal{L}^*$  suffers a claim and the other half does not. The idea now is to fit a GLM only on this learning data  $\mathcal{L}^*$ , and because we only consider 44'868 insurance policies the fitting should be fast.

There is still one point to be considered, namely, in the new learning data  $\mathcal{L}^*$  policies with claims are over-represented (because we work in a low frequency problem). This motivates that we adjust the time exposures  $v_i$  in  $\mathcal{L}_0^*$  accordingly by multiplying as follows

$$v_i \mapsto v_i^* = v_i \frac{\sum_{j=1}^n v_j \mathbb{1}_{\{N_j=0\}}}{\sum_{v_j \in \mathcal{L}_0^*} v_j}.$$

Thus, we stretch the exposures of the policies without claims in  $\mathcal{L}^*$ ; for our data this factor is 26.17. This then provides us with an empirical frequency on  $\mathcal{L}^*$  of 7.36% which is identical to the observed frequency on the entire learning data  $\mathcal{L}$ .

We fit model Poisson GLM3 on this reduced (and exposure adjusted) learning data  $\mathcal{L}^*$ , the results are presented on the last line of Table 5.6. This model can be fitted in 1s, and by construction it fulfills the balance property. The resulting in-sample and out-of-sample losses (evaluated on the entire data  $\mathcal{L}$  and  $\mathcal{T}$ ) are very close to model Poisson GLM3 which verifies that the model fitted only on the learning data  $\mathcal{L}^*$  gives a good approximation. We do not provide AIC because the data used is not identical to the data used to fit the other models. The tenfold cross-

**Table 5.6** Run times, number of parameters, AICs, in-sample and out-of-sample deviance losses, tenfold cross-validation losses (units are in  $10^{-2}$ ) and in-sample average frequency of the null model (intercept model) and of different Poisson GLMs, the last row uses under-sampling in model Poisson GLM3

	Run time	# param.	AIC	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Tenfold CV loss $\widehat{\mathcal{D}}^{\text{CV}}$	Aver. freq.
Poisson null	–	1	199'506	25.213	25.445	25.213	7.36%
Poisson GLM1	16 s	49	192'818	24.101	24.146	24.121	7.36%
Poisson GLM2	15 s	48	192'753	24.091	24.113	24.110	7.36%
Poisson GLM3	15 s	50	192'716	24.084	24.102	24.104	7.36%
under-sampling	1 s	50	–	24.098	24.108	24.120	7.36%

validation loss is a little bit bigger which seems to be a consequence of applying the non-stratified version to only 44'868 insurance policies, i.e., this higher cross-validation loss shows that we fit the model on less data which provides higher uncertainty in model fitting. This finishes this example.

The presented method is called under-sampling because we under-sample from the insurance policies without claims to make both classes (policies with claims and policies without claims) equally large. Alternatively, to achieve a class balance we could also over-sample from the minority class by duplicating policies. This has a similar effect, but it increases run times. Importantly, if we under- or over-sample we *have* to adjust the exposures correspondingly. Otherwise we obtain a biased model that is not useful for pricing, the same applies to methods such as the synthetic minority oversampling technique (SMOTE) and similar techniques.

Alternatively, to under-sampling we could also fit a so-called zero-truncated Poisson (ZTP) model to the data by only fitting a model on the insurance policies that suffer at least one claim, and adjusting the distribution to the observations  $N_i | \{N_i \geq 1\}$ . This is rather similar to a hurdle Poisson model and we come back to this in Example 6.19, below.

### 5.3.5 Over-Dispersion in Claim Counts Modeling

#### Mixed Poisson Distribution

In the previous example we have seen that the considered Poisson GLMs do not fully fit our data, at least not with the chosen feature engineering, because there is over-dispersion in the data (relative to the chosen models). This may give rise to consider models that allow for over-dispersion. Typically, such over-dispersed models are constructed starting from the Poisson model, because the Poisson model enjoys many nice properties as we have seen above. A natural extension is to introduce the family of mixed Poisson models, where the frequency is not modeled with a single parameter but rather with a whole family of parameters described by an underlying mixing distribution.

In the dual mean parametrization the Poisson distribution for  $Y = N/v$  reads as

$$Y \sim f(y; \lambda, v) = e^{-v\lambda} \frac{(v\lambda)^{vy}}{(vy)!} \quad \text{for } y \in \mathbb{N}_0/v,$$

where the mean parameter is given by  $\lambda = \kappa'(\theta) = \exp\{\theta\}$ , and  $\theta$  denotes the canonical parameter; on purpose we use for the mean notation  $\lambda$  instead of  $\mu$ , here, the reason will become clear below. This model satisfies for the first two moments of  $N = vY$

$$\mathbb{E}_\lambda [N] = v\kappa'(\theta) = v\lambda \quad \text{and} \quad \text{Var}_\lambda (N) = v\kappa''(\theta) = v\lambda = \mathbb{E}_\lambda [N],$$

with dispersion parameter  $\varphi = 1$ . A mixed Poisson distribution is obtained by mixing/integrating over different frequency parameters  $\lambda > 0$ . We choose a

distribution  $\pi$  on  $\mathbb{R}_+$  (strictly positively supported), and define the new distribution

$$Y = N/v \sim f_\pi(y; v) = \int_{\mathbb{R}_+} f(y; \lambda, v) d\pi(\lambda) = \int_{\mathbb{R}_+} e^{-v\lambda} \frac{(v\lambda)^{vy}}{(vy)!} d\pi(\lambda). \quad (5.36)$$

If  $\pi$  is not concentrated in a single point, the tower property immediately implies

$$\mathbb{E}_\pi [N] < \text{Var}_\pi (N), \quad (5.37)$$

supposed that the moments exist, we refer to Lemma 2.18 in Wüthrich [387]. Hence, mixing over different frequency parameters allows us to receive over-dispersion. Of course, this concept can also be applied to mixing over the canonical parameter  $\theta$  in the EF (instead of the mean parameter).

This leads to the framework of Bayesian credibility models which are widely used and studied in actuarial science, we refer to the textbook of Bühlmann–Gisler [58]. We have already met this idea in the Bayesian decision rule of Example 3.3 which has led to the Bayesian estimator in Definition 3.6.

### Negative-Binomial Model

In the case of the Poisson model, the gamma distribution is a particularly attractive mixing distribution for  $\lambda$  because it allows for a closed-form solution in (5.36), and  $f_{\pi=\Gamma}(y; v)$  will be a negative-binomial distribution.<sup>4</sup> One can choose different parametrizations of this mixing distribution, and they will provide different scalings in the resulting negative-binomial distribution. We choose the following parametrization  $\pi(\lambda) \stackrel{(d)}{=} \Gamma(v\alpha, v\alpha/\mu)$  for mean parameter  $\mu > 0$  and shape parameter  $v\alpha > 0$ . This implies, see (5.36),

$$\begin{aligned} f_{\text{NB}}(y; \mu, v, \alpha) &= \int_{\mathbb{R}_+} e^{-v\lambda} \frac{(v\lambda)^{vy}}{(vy)!} \frac{(v\alpha/\mu)^{v\alpha}}{\Gamma(v\alpha)} \lambda^{v\alpha-1} e^{-v\alpha\lambda/\mu} d\lambda \\ &= \frac{\Gamma(vy + v\alpha)}{(vy)! \Gamma(v\alpha)} \frac{v^{vy} (v\alpha/\mu)^{v\alpha}}{(v + v\alpha/\mu)^{vy+v\alpha}} \\ &= \binom{vy + v\alpha - 1}{vy} (e^\theta)^{vy} (1 - e^\theta)^{v\alpha}, \end{aligned}$$

---

<sup>4</sup> The gamma distribution is the conjugate prior to the Poisson distribution. As a result, the posterior distribution, given observations, will again be a gamma distribution with posterior parameters, see Section 8.1 of Wüthrich [387]. This Bayesian model has been introduced to the actuarial literature by Bichsel [32].

setting for canonical parameter  $\theta = \log(\mu/(\mu + \alpha)) < 0$ . This is the negative-binomial distribution we have already met in (2.5). A single-parameter linear EDF representation is given by, we set unit dispersion parameter  $\varphi = 1$ ,

$$Y \sim f_{\text{NB}}(y; \theta, v, \alpha) = \exp \left\{ \frac{y\theta + \alpha \log(1 - e^\theta)}{1/v} + \log \binom{vy + v\alpha - 1}{vy} \right\}, \quad (5.38)$$

where this is a density w.r.t. the counting measure on  $\mathbb{N}_0/v$ . The cumulant function and the canonical link, respectively, are given by

$$\kappa(\theta) = -\alpha \log(1 - e^\theta) \quad \text{and} \quad \theta = h(\mu) = \log \left( \frac{\mu}{\mu + \alpha} \right) \in \Theta = (-\infty, 0).$$

Note that  $\alpha > 0$  is treated as nuisance parameter (which is a fixed part of the cumulant function, here). The first two moments of the claim count  $N = vY$  are given by

$$v\mu = \mathbb{E}_\theta[N] = v\alpha \frac{e^\theta}{1 - e^\theta}, \quad (5.39)$$

$$\text{Var}_\theta(N) = \mathbb{E}_\theta[N] \left( 1 + \frac{e^\theta}{1 - e^\theta} \right) = \mathbb{E}_\theta[N] \left( 1 + \frac{\mu}{\alpha} \right) > \mathbb{E}_\theta[N]. \quad (5.40)$$

This shows that we receive a fixed over-dispersion of size  $\mu/\alpha$ , which (in this parametrization) does not depend on the exposure  $v$ ; this is the reason for choosing a mixing distribution  $\pi(\lambda) \stackrel{(d)}{=} \Gamma(v\alpha, v\alpha/\mu)$ . This parametrization is called NB2 parametrization.

#### Remarks 5.21

- We emphasize that the effective domain  $\Theta = (-\infty, 0)$  is one-sided bounded. Therefore, the canonical link for the linear predictor will not work in general because the linear predictor  $\mathbf{x} \mapsto \eta(\mathbf{x})$  can be both-sided unbounded in a GLM setting. Instead, we use the log-link for  $g(\cdot)$  in our example below, with the downside that one loses the balance property.
- The unit deviance in this negative-binomial EDF model is given by

$$(y, \mu) \mapsto \mathfrak{d}(y, \mu) = 2 \left[ y \log \left( \frac{y}{\mu} \right) - (y + \alpha) \log \left( \frac{y + \alpha}{\mu + \alpha} \right) \right],$$

we also refer to Table 4.1 for  $\alpha = 1$ . We emphasize that this is the unit deviance in a single-parameter linear EDF, and we only aim at estimating canonical parameter  $\theta \in \Theta$  and mean parameter  $\mu \in \mathcal{M}$ , respectively, whereas  $\alpha > 0$  is treated as a given nuisance parameter. This is important because the unit deviance relies on the saturated model which, in general, estimates a one-dimensional



parameter  $\theta$  and  $\mu$ , respectively, from the one-dimensional observation  $Y$ . The nuisance parameter is not affected by the consideration of the saturated model, and it is treated as a fixed part of the cumulant function, which is not estimated at this stage. An important consequence of this is that model comparison using deviance residuals only works for identical nuisance parameters.

- We mention that we receive over-dispersion in (5.40) though we have dispersion parameter  $\varphi = 1$  in (5.38). Alternatively, we could do the duality transformation  $y \mapsto \tilde{y} = y/\alpha$  for nuisance parameter  $\alpha > 0$ ; this gives the reproductive form of the negative-binomial model NB2, see also Remarks 2.13. This provides us with a density on  $\mathbb{N}_0/(v\alpha)$ , set  $\tilde{\varphi} = 1/\alpha$ ,

$$\tilde{Y} \sim f_{\text{NB}}(\tilde{y}; \theta, v/\tilde{\varphi}) = \exp \left\{ \frac{\tilde{y}\theta + \log(1 - e^\theta)}{1/(v\alpha)} + \log \binom{v\alpha\tilde{y} + v\alpha - 1}{v\alpha\tilde{y}} \right\}.$$

The cumulant function and the canonical link, respectively, are now given by

$$\kappa(\theta) = -\log(1 - e^\theta) \quad \text{and} \quad \theta = h(\tilde{\mu}) = \log \left( \frac{\tilde{\mu}}{\tilde{\mu} + 1} \right) \in \Theta = (-\infty, 0).$$

The first two moments are for  $\theta \in \Theta$  given by

$$\begin{aligned} \tilde{\mu} &= \mathbb{E}_\theta[\tilde{Y}] = \frac{e^\theta}{1 - e^\theta}, \\ \text{Var}_\theta(\tilde{Y}) &= \frac{\tilde{\varphi}}{v} \kappa''(\theta) = \frac{1}{v\alpha} \tilde{\mu} (1 + \tilde{\mu}). \end{aligned}$$

Thus, we receive the reproductive EDF representation with dispersion parameter  $\tilde{\varphi} = 1/\alpha$  and variance function  $V(\tilde{\mu}) = \tilde{\mu}(1 + \tilde{\mu})$ . Moreover,  $N = vY = v\alpha\tilde{Y}$ .

- The negative-binomial model with the NB1 parametrization uses the mixing distribution  $\pi(\lambda) \stackrel{(d)}{=} \Gamma(\mu v/\alpha, v/\alpha)$ . This leads to mean  $\mathbb{E}_\theta[N] = v\mu$  and variance  $\text{Var}_\theta(N) = \mathbb{E}_\theta[N](1 + \alpha)$ . In this parametrization,  $\mu$  enters the gamma function as  $\Gamma(\mu v/\alpha)$  in the gamma density which does not allow for an EDF representation. This parametrization has been called NB1 by Cameron–Trivedi [63] because both terms in the variance  $\text{Var}_\theta(N) = v\mu + v\mu\alpha$  are linear in  $\mu$ . In contrast, in the NB2 parametrization the second term has a square  $v\mu^2/\alpha$  in  $\mu$ , see (5.40). Further discussion is provided in Greene [171].

### Nuisance Parameter Estimation

All previous statements have been based on the assumption that  $\alpha > 0$  is a *given* nuisance parameter. If  $\alpha$  needs to be estimated too, then, we drop out of the EF. In this case, an iterative estimation procedure is applied to the EDF representation (5.38). One starts with a fixed nuisance parameter  $\alpha^{(0)}$  and fits the

negative-binomial GLM with MLE which provides a first set of MLE  $\widehat{\boldsymbol{\beta}}^{(1)} = \widehat{\boldsymbol{\beta}}^{(1)}(\alpha^{(0)})$ . Based on this estimate the nuisance parameter is updated  $\alpha^{(0)} \mapsto \alpha^{(1)}$  by maximizing the log-likelihood in  $\alpha$  for given  $\widehat{\boldsymbol{\beta}}^{(1)}$ . Iteration of this procedure then leads to a joint estimation of regression parameter  $\boldsymbol{\beta}$  and nuisance parameter  $\alpha$ . Both MLE steps in this algorithm increase the joint log-likelihood.

*Remark 5.22 (Implementation of the Negative-Binomial GLM in R)* Implementation of the negative-binomial model needs some care. There are two R procedures `glm` and `glm.nb` that can be used to fit negative-binomial GLMs, the latter being built on the former. The procedure `glm` is just the classical R procedure [307] that is usually used to fit GLMs within the EDF, it requires to set

```
family=negative.binomial(theta, link="log").
```

This parametrization considers the single-parameter linear EF on  $\mathbb{N}$  (for mean  $\mu \in \mathcal{M}$ )

$$f_{\text{NB}}(n; \mu, \text{theta}) = \binom{n + \text{theta} - 1}{n} \left( \frac{\mu}{\mu + \text{theta}} \right)^n \left( 1 - \frac{\mu}{\mu + \text{theta}} \right)^{\text{theta}},$$

where  $\text{theta} > 0$  denotes the nuisance parameter. The tricky part now is that we have to bring in the different exposures  $v_i$  of all policies  $1 \leq i \leq n$ . That is, we would like to have for claim counts  $n_i = v_i y_i$ , see (5.38),

$$\begin{aligned} f_{\text{NB}}(y_i; \mu_i, v_i, \alpha) &= \binom{v_i y_i + v_i \alpha - 1}{v_i y_i} \left( \frac{v_i \mu_i}{v_i \mu_i + v_i \alpha} \right)^{v_i y_i} \left( 1 - \frac{v_i \mu_i}{v_i \mu_i + v_i \alpha} \right)^{v_i \alpha} \\ &= \binom{v_i y_i + v_i \alpha - 1}{v_i y_i} \left[ \left( \frac{\mu_i}{\mu_i + \alpha} \right)^{y_i} \left( 1 - \frac{\mu_i}{\mu_i + \alpha} \right)^{\alpha} \right]^{v_i}. \end{aligned}$$

The square bracket can be implemented in `glm` as a scaled and weighted regression problem, see Listing 5.8 with  $\text{theta} = \alpha$ . This approach provides the correct GLM parameter estimates  $\widehat{\boldsymbol{\beta}}^{\text{MLE}}$  for given  $\alpha$ , however, the outputted AIC values cannot be compared to the Poisson case. Note that the Poisson case of Table 5.5 considers observations  $N_i$  whereas Listing 5.8 uses  $Y_i = N_i/v_i$ . For this reason we calculate the log-likelihood and AIC by an own implementation.

The same remark applies to `glm.nb`, and also nuisance parameter estimation cannot be performed by that routine under different exposures  $v_i$ . Therefore, we have implemented an iterative estimation algorithm ourselves, alternating `glm` of Listing 5.8 for given  $\alpha$  and a maximization routine `optimize` to find the optimal  $\alpha$  for given  $\boldsymbol{\beta}$  using (5.38). We have applied this iteration in Example 5.23, below, and it has converged in 5 iterations.

*Example 5.23 (Negative-Binomial Distribution for Claim Counts)* We revisit the MTPL claim frequency GLM example of Sect. 5.3.4, but we replace the Poisson distribution by the negative-binomial one. We start with the negative-binomial (NB)

**Listing 5.8** Implementation of model NB GLM3

```
d.glmnb <- glm(ClaimNb/Exposure ~ VehPowerGLM + VehAgeGLM
+ log(DrivAge) + I(DrivAge^3) + I(DrivAge^4)
+ BonusMalusGLM*DrivAge + BonusMalusGLM*I(DrivAge^2)
+ VehBrand + VehGas + DensityGLM + Region + AreaGLM,
data=learn, weights=Exposure,
family=negative.binomial(alpha, link="log"))
```

**Table 5.7** Run times, number of parameters, AICs, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the null models (Poisson and negative-binomial) and the Poisson and negative-binomial GLMs. The optimal model is highlighted in boldface

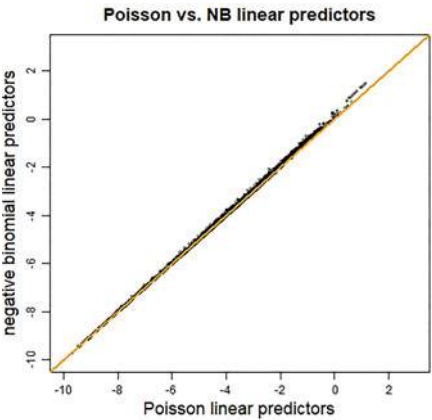
	Run time	# Param.	AIC	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	199’506	25.213	25.445	7.36%
Poisson GLM3	15 s	50	192’716	24.084	24.102	7.36%
NB null $\hat{\alpha}_{\text{null}}^{\text{MLE}} = 1.059$	–	2	198’466	20.357	20.489	7.36%
NB null $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$	–	1	198’564	21.796	21.948	7.36%
NB GLM3 $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$	85s	51	<b>192’113</b>	20.722	20.674	7.38%

null model. The NB null model has two parameters, the homogeneous (overall) frequency and the nuisance parameter. MLE of the homogeneous overall frequency is identical to the one in the Poisson null model, and MLE of the nuisance parameter provides  $\hat{\alpha}_{\text{null}}^{\text{MLE}} = 1.059$ . This is substantially smaller than infinity and suggests over-dispersion. The results are presented on the third line of Table 5.7. We observe a smaller AIC of the NB null model against the Poisson null model which says that we should allow for over-dispersion.

We now focus on the NB GLM. The feature pre-processing is done exactly as in model Poisson GLM3, and we choose the log-link for  $g$ . We call this model NB GLM3. The iterative estimation procedure outlined above provides a nuisance parameter estimate  $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$ . This is bigger than in the NB null model because the regression structure explains some part of the over-dispersion, however, it is still substantially smaller than infinity which justifies the inclusion of this over-dispersion parameter.

The last line of Table 5.7 gives the result of model NB GLM3. From AIC we conclude that we favor the negative-binomial GLM over the Poisson GLM since AIC decreases from 192’716 to 192’113. The in-sample and out-of-sample deviance losses can only be compared within the same models, i.e., the models that have the same cumulant function. This also applies to the negative-binomial models which have cumulant function  $\kappa(\theta) = -\alpha \log(1 - e^\theta)$ . Thus, to compare the NB null model and model NB GLM3, we need to choose the same nuisance parameter  $\alpha$ . For this reason we added this second NB null model to Table 5.7. This second NB null model no longer uses the MLE  $\hat{\alpha}_{\text{null}}^{\text{MLE}}$ , therefore, the corresponding AIC only includes one estimated parameter.

**Fig. 5.7** Poisson logged predictors vs. negative-binomial logged predictors



**Table 5.8** Out-of-sample deviance losses: forecast dominance. The optimal model is highlighted in boldface

Model	Poisson deviance	NB deviance $\hat{\alpha}_{\text{null}}^{\text{MLE}} = 1.059$	NB deviance $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$
Null model	25.445	20.489	21.948
Poisson GLM3	24.102	19.266	20.678
NB GLM3 $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$	<b>24.100</b>	<b>19.262</b>	<b>20.674</b>

As mentioned above, deviance losses can only be compared under exactly the same cumulant function (including the same nuisance parameters). If we want to have a more robust model selection, we can consider forecast dominance according to Definition 4.20. Being less ambitious, here, we consider forecast dominance only for the three considered cumulant functions Poisson, negative-binomial with  $\hat{\alpha}_{\text{null}}^{\text{MLE}} = 1.059$  and negative-binomial with  $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$ . The out-of-sample deviance losses are given in Table 5.8 in the different columns. According to this forecast dominance analysis we also give preference to model NB GLM3, but model Poisson GLM3 is pretty close.

Figure 5.7 compares the logged predictors  $\log(\hat{\mu}_i)$ ,  $1 \leq i \leq n$ , of the models Poisson GLM3 and NB GLM3. We see a huge similarity in these predictors, only high frequency policies are judged slightly differently by the NB model compared to the Poisson model.

Table 5.9 gives the predicted number of claims against the observed ones. We observe that model NB GLM3 predicts more accurately the number of policies with 2 or less claims, but it over-estimates the number of policies with more than 2 claims. This may also be related to the fact that the estimated in-sample frequency has a

**Table 5.9** Contingency table of observed number of policies against predicted number of policies with given claim counts ClaimNb

	Numbers of claims ClaimNb					
	0	1	2	3	4	5
Observed number of policies	587'772	21'198	1'174	57	4	1
Poisson predicted number of policies	587'325	22'064	779	34	3	0.3
NB predicted number of policies	587'902	20'982	1'200	100	15	4

positive bias in model NB GLM3, see Table 5.7. That is, since we do not work with the canonical link, we do not have the balance property.

**Listing 5.9** drop1 analysis of model NB GLM3

1	Single term deletions					
2						
3	Model:					
4	ClaimNb/Exposure ~ VehPowerGLM + VehAgeGLM + DrivAge + log(DrivAge) +					
5	I(DrivAge^2) + I(DrivAge^3) + I(DrivAge^4) + BonusMalusGLM *					
6	DrivAge + BonusMalusGLM * I(DrivAge^2) + BonusMalusGLM +					
7	VehBrand + VehGas + DensityGLM + Region + AreaGLM					
8		Df	Deviance	AIC	scaled dev.	Pr(>Chi)
9	<none>		126446	171064		
10	VehPowerGLM	5	126524	171102	48.266	3.134e-09 ***
11	VehAgeGLM	2	126655	171190	130.070	< 2.2e-16 ***
12	log(DrivAge)	1	126592	171153	91.057	< 2.2e-16 ***
13	I(DrivAge^3)	1	126527	171112	50.483	1.202e-12 ***
14	I(DrivAge^4)	1	126508	171100	38.381	5.820e-10 ***
15	VehBrand	10	126658	171176	132.098	< 2.2e-16 ***
16	VehGas	1	126583	171147	85.232	< 2.2e-16 ***
17	DensityGLM	1	126456	171068	6.137	0.01324 *
18	Region	21	126622	171132	109.838	5.042e-14 ***
19	AreaGLM	1	126450	171064	2.411	0.12049
20	DrivAge:BonusMalusGLM	1	126484	171085	23.481	1.262e-06 ***
21	I(DrivAge^2):BonusMalusGLM	1	126490	171089	27.199	1.836e-07 ***
22	---					
23	Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1					

We close this example by providing the drop1 analysis in Listing 5.9. From this analysis we conclude that the feature component Area should be dropped. Of course, this confirms the high collinearity between Density and Area which implies that we do not need both variables in the model. We remark that the AIC values in Listing 5.9 are not on our scale, as stated in Remark 5.22. ■

### 5.3.6 Zero-Inflated Poisson Model

In many applications it is the case that the Poisson distribution does not fully fit the claim counts data because there are too many policies with zero claims, i.e.,

policies with  $Y = 0$ , compared to a Poisson assumption. This topic has attracted some attention in the recent actuarial literature, see, e.g., Boucher et al. [43–45], Frees et al. [137], Calderín-Ojeda et al. [62] and Lee [239]. An obvious solution to this problem is to ‘artificially’ increase the probability of a zero claim compared to a Poisson model, this is the proposal introduced by Lambert [232].  $Y$  has a zero-inflated Poisson (ZIP) distribution if the probability weights of  $Y$  are given by (set  $v = 1$ )

$$f_{\text{ZIP}}(y; \theta, \pi_0) = \begin{cases} \pi_0 + (1 - \pi_0)e^{-\mu} & \text{for } y = 0, \\ (1 - \pi_0)e^{-\mu} \frac{\mu^y}{y!} & \text{for } y \in \mathbb{N}, \end{cases}$$

for  $\pi_0 \in (0, 1)$ ,  $\mu = e^\theta > 0$ , and for the Poisson probability weights we refer to (2.4). For  $\pi_0 > 0$  the weight of a zero claim  $Y = 0$  is increased (inflated) compared to the original Poisson distribution.

*Remarks 5.24*

- The ZIP distribution has different interpretations. It can be interpreted as a hierarchical model where we have a latent variable  $Z$  which indicates with probability  $\pi_0$  that we have an excess zero, and with probability  $1 - \pi_0$  we have an ordinary Poisson distribution, i.e. for  $y \in \mathbb{N}_0$

$$\mathbb{P}_\theta [Y = y | Z = z] = \begin{cases} \mathbb{1}_{\{y=0\}} & \text{for } z = 0, \\ e^{-\mu} \frac{\mu^y}{y!} & \text{for } z = 1, \end{cases} \quad (5.41)$$

with  $\mathbb{P}[Z = 0] = 1 - \mathbb{P}[Z = 1] = \pi_0$ .

The latter shows that we can also understand it as a mixture of two distributions, namely, of the Poisson distribution and of a single point measure in  $y = 0$  with mixing probability  $\pi_0$ . Mixture distributions are going to be discussed in Sect. 6.3.1, below. In this sense, we can also interpret the model as a mixed Poisson model with mixing distribution  $\pi(\lambda)$  being a Bernoulli distribution taking values 0 and  $\mu$  with probability  $\pi_0$  and  $1 - \pi_0$ , respectively, see (5.36), and the former parameter  $\lambda = 0$  leads to a degenerate Poisson model.

- We have introduced the ZIP model, but this approach is neither limited to the Poisson model nor the zeros. For instance, we could also consider an inflated negative-binomial model where both the zeros and the ones are inflated with probabilities  $\pi_0, \pi_1 > 0$  such that  $\pi_0 + \pi_1 < 1$ .
- Hurdle models are an alternative way to model excess zeros. Hurdle models have been introduced by Cragg [83], and they also allow for too little zeros. A hurdle (Poisson) model mixes a lower-truncated (Poisson) count distribution with a point mass in zero

$$f_{\text{hurdle Poisson}}(y; \theta, \pi_0) = \begin{cases} \pi_0 & \text{for } y = 0, \\ (1 - \pi_0) \frac{e^{-\mu} \mu^y}{1 - e^{-\mu}} & \text{for } y \in \mathbb{N}, \end{cases} \quad (5.42)$$

for  $\pi_0 \in (0, 1)$  and  $\mu > 0$ . For  $\pi_0 > e^{-\mu}$  the weight of a zero claim is increased and for  $\pi_0 < e^{-\mu}$  it is decreased. This distribution is called a hurdle distribution, because we first need to overcome the hurdle at zero to come to the Poisson model. Lower-truncated distributions are studied in Sect. 6.4, below, and mixture distributions are discussed in Sect. 6.3.1. In general, fitting lower-truncated distributions is challenging because the density and the distribution function should both have tractable forms to perform MLE for truncated distributions. The Expectation-Maximization (EM) algorithm is a useful tool to perform model fitting under truncation. We come back to the hurdle Poisson model in Example 6.19, below, and it is also closely related to the zero-truncated Poisson (ZTP) model discussed in Remarks 6.20.

The first two moments of a ZIP random variable  $Y \sim f_{\text{ZIP}}(\cdot; \theta, \pi_0)$  are given by

$$\begin{aligned}\mathbb{E}_{\theta, \pi_0}[Y] &= (1 - \pi_0)\mu, \\ \text{Var}_{\theta, \pi_0}(Y) &= (1 - \pi_0)\mu + (\pi_0 - \pi_0^2)\mu^2 = \mathbb{E}_{\theta, \pi_0}[Y](1 + \pi_0\mu),\end{aligned}$$

these calculations easily follow with the latent variable  $Z$  interpretation from above. As a consequence, we receive an over-dispersed model with over-dispersion  $\pi_0\mu$  (the latter also follows from the fact that we consider a mixed Poisson distribution with a Bernoulli mixing distribution having weights  $\pi_0$  in 0 and  $1 - \pi_0$  in  $\mu > 0$ , see (5.37)).

Unfortunately, MLE does not allow for explicit solutions in this model. The score equations of  $Y_i \stackrel{\text{i.i.d.}}{\sim} f_{\text{ZIP}}(\cdot; \theta, \pi_0)$  are given by

$$\begin{aligned}\nabla_{(\pi_0, \mu)} \ell_Y(\pi_0, \mu) &= \nabla_{(\pi_0, \mu)} \sum_{i=1}^n \log(\pi_0 + (1 - \pi_0)e^{-\mu}) \mathbb{1}_{\{Y_i=0\}} \\ &\quad + \nabla_{(\pi_0, \mu)} \sum_{i=1}^n \log\left((1 - \pi_0)e^{-\mu} \frac{\mu^{y_i}}{y_i!}\right) \mathbb{1}_{\{Y_i>0\}} = 0.\end{aligned}$$

The R package `pscl` [401] has a function called `zeroinfl` which uses the general purpose optimizer `optim` to find the MLEs in the ZIP model. Alternatively, we could explore the EM algorithm for mixture distributions presented in Sect. 6.3, below.

In insurance applications, the ZIP application can be problematic if we have different exposures  $v_i > 0$  for different insurance policies  $i$ . In the Poisson GLM case with canonical link choice we typically integrate the different exposures into the offset, see (5.27). However, it is not clear whether and how we should integrate the different exposures into the zero-inflation probability  $\pi_0$ . It seems natural to believe that shorter exposures should increase  $\pi_0$ , but the explicit functional form of this increase can be debated, some options are discussed in Section 5 of Lee [239].

**Listing 5.10** Implementation of model ZIP GLM3

```
1 d.ZIP <- zeroinfl(ClaimNb ~ VehPowerGLM + VehAgeGLM
2   + log(DrivAge) + I(DrivAge^3) + I(DrivAge^4)
3   + BonusMalusGLM*DrivAge + BonusMalusGLM*I(DrivAge^2)
4   + VehBrand + VehGas + DensityGLM + Region
5   + AreaGLM | 1,
6   data=learn, offset=log(Exposure), dist='poisson', link='logit',
7   start=list(count=glm3$coefficients, zero=c(-0.4153)) )
```

**Table 5.10** Run times, number of parameters, AICs, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the null models (Poisson, negative-binomial and ZIP) and the Poisson, negative-binomial and ZIP GLMs. The optimal model is highlighted in boldface

	Run time	# Param.	AIC	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	199’506	25.213	25.445	7.36%
Poisson GLM3	15 s	50	192’716	24.084	24.102	7.36%
NB null $\hat{\alpha}_{\text{null}}^{\text{MLE}} = 1.059$	–	2	198’466	20.357	20.489	7.36%
NB null $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$	–	1	198’564	21.796	21.948	7.36%
NB GLM3 $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$	85 s	51	<b>192’113</b>	20.722	20.674	7.38%
ZIP null	20 s	2	198’638	–	–	7.43%
ZIP GLM3 (null $\pi_0$ )	270 s	51	192’393	–	–	7.37%

In the following application, we simply choose  $\pi_0$  independent of the exposures, but certainly this is not the best modeling choice.

*Example 5.25 (ZIP Model for Claim Counts)* We revisit the MTPL claim frequency example of Sect. 5.3.4, but this time we fit a ZIP model. For the Poisson part we use exactly the same GLM regression function as in model Poisson GLM3 and, in particular, we use for the different exposures  $v_i$  of the insurance policies the offset term  $o_i = \log v_i$ , see line 6 of Listing 5.10. This offset only acts on the Poisson part of the ZIP GLM. The zero-inflating probability  $\pi_0$  is modeled with a logistic Bernoulli model, see Sect. 2.1.2. For computational reasons, we choose the null model for the Bernoulli part modeling the zero-inflation  $\pi_0$ . This is indicated by the “1” on line 5 of Listing 5.10. This 1 should be expanded if we also want to consider a regression model for the zero-inflating probability  $\pi_0$  and, in particular, if we want to integrate an offset term for the exposure. We can set this term to `offset(f)`, where `f` is a suitable transformation of the exposure. Furthermore, successful calibration requires meaningful starting values, otherwise `zeroinfl` will not find the MLEs. We start the algorithm in the parameters of model Poisson GLM3, see line 7 of Listing 5.10. The results are presented in Table 5.10.

Firstly, we see that the run times are not fully competitive in this implementation, even if we choose the null model for the zero-inflating probability  $\pi_0$ , i.e., only



**Table 5.11** Out-of-sample deviance losses: forecast dominance. The optimal model is highlighted in boldface

Model	Poisson deviance	NB deviance $\hat{\alpha}_{\text{null}}^{\text{MLE}} = 1.059$	NB deviance $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$
Null model	25.445	20.489	21.948
Poisson GLM3	24.102	19.266	20.678
NB GLM3 $\hat{\alpha}_{\text{NB}}^{\text{MLE}} = 1.810$	<b>24.100</b>	<b>19.262</b>	<b>20.674</b>
ZIP null model	25.446	20.490	21.949
ZIP GLM3	24.103	19.267	20.679

**Table 5.12** Contingency table of observed numbers of policies against predicted numbers of policies with given claim counts ClaimNb

	Numbers of claims ClaimNb					
	0	1	2	3	4	5
Observed number of policies	587'772	21'198	1'174	57	4	1
Poisson predicted number of policies	587'325	22'064	779	34	3	0.3
NB predicted number of policies	587'902	20'982	1'200	100	15	4
ZIP predicted number of policies	587'829	21'094	1'191	79	9	4

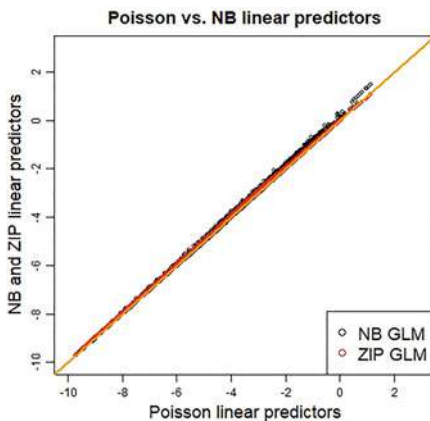
one intercept parameter is involved for determining  $\pi_0$ . Secondly, in this model we cannot calculate deviance losses because the saturated model has two parameters for each observation. Thirdly, the model does not satisfy the balance property though we work with the canonical links for the Poisson part and the Bernoulli part, however, this property gets lost under the combination of these two model parts.

Most interesting are the AIC values. We observe that the ZIP GLM improves the Poisson GLM, but it has a bigger AIC value than the negative-binomial GLM. From this we conclude that we give preference to the negative-binomial model in our case.

Considering forecast dominance according to Definition 4.20, but restricted to the three deviance losses studied in Example 5.23, we receive Table 5.11. Also this table gives preference to the negative-binomial GLM. However, if we consider the table of the observed numbers of policies against the predicted numbers of claims, see Table 5.12, we give preference to the ZIP GLM because it has the lowest  $\chi^2$ -value, i.e., it reflects best (in-sample) our observations.

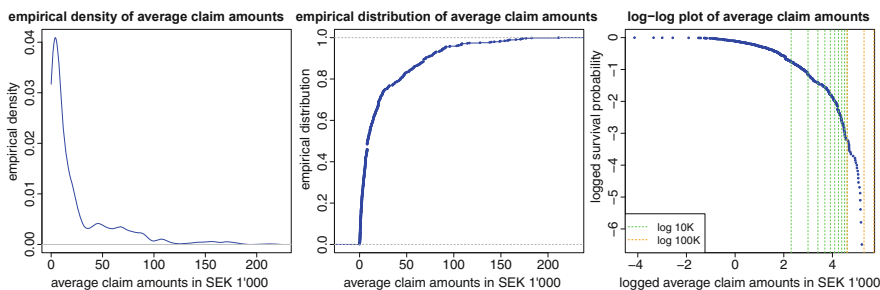
Figure 5.8 compares the resulting predictors on the log-scale. From this plot we conclude that in our example the predictors of the ZIP GLM are closer to the Poisson ones than the NB GLM predictors. In a next step, one could refine the zero-inflating probability  $\pi_0$  modeling by integrating the exposure and further feature information. This would lead to a further model improvement. We refrain here from doing so and close this example; in Example 6.19, below, we study the hurdle Poisson model. ■

**Fig. 5.8** Comparison linear predictors of the NB and ZIP GLMs against the ones of the Poisson GLM



### 5.3.7 Lab: Gamma GLM for Claim Sizes

As a second example we consider claim size modeling within GLMs. For this example we do not use the French MTPL claims data because the empirical density plot in Fig. 13.15 indicates that a GLM will not fit to that data. The French MTPL data seems to have three distinct modes, which suggests to use a mixture distribution. Moreover, the log-log plot indicates a regularly varying tail, which cannot be captured by the EDF on the original observation scale; we are going to study this data in Example 6.14, below. Here, we use the Swedish motorcycle data, previously used in the textbook of Ohlsson–Johansson [290] and described in Chap. 13.2. From Fig. 5.9 we see that the empirical density has one mode, and the log-log plot supports light tails, i.e., the gamma model might be a suitable choice for this data. Therefore, we choose a gamma GLM with log-link  $g$ . As described above, the log-link is not the canonical link for the gamma EDF distribution but it ensures the right sign w.r.t. the linear predictor  $\eta_i = \langle \beta, x_i \rangle$ . Working with the log-link in the gamma model will imply that the balance property is not fulfilled.



**Fig. 5.9** (lhs) Empirical density, (middle) empirical distribution and (rhs) log-log plot of claim amounts of the Swedish motorcycle data presented in Chap. 13.2

## Feature Engineering

We have 4 continuous feature components OwnerAge, RiskClass, VehAge and BonusClass, one binary feature component Gender and a categorical component Area, see Listing 13.4. We have decided for a minimal feature engineering; we refer to Figs. 13.19 (rhs) and 13.20 (rhs) for descriptive plots. We use the continuous variables directly in a log-linear fashion, we add quadratic terms for OwnerAge and VehAge, we merge RiskClass 6 and 7, and we censor VehAge at 20. Area is categorical, but we may interpret the Zone levels as ordinal categorical, and mapping them to integers allows us to use them in a continuous fashion; Fig. 13.19 (middle row, rhs) shows that this is a reasonable choice. Moreover, we merge Zone 5, 6 and 7 due to small volumes and their similar behavior.

## Gamma Generalized Linear Model

The Swedish motorcycle claim amount data poses the special difficulty that we do not have individual claim observations  $Z_{i,j}$ , but we only know the total claim amounts  $S_i = \sum_{j=1}^{N_i} Z_{i,j}$  and the number of claims  $N_i$  on each insurance policy; Fig. 5.9 shows average claims  $S_i/N_i$  of insurance policies  $i$  with  $N_i > 0$ . In general, this poses a problem in statistical modeling, but in the gamma model this problem can be handled because the gamma distribution is closed under aggregation of i.i.d. gamma claims  $Z_{i,j}$ . In all what follows in this section, we only study insurance policies with  $N_i > 0$ , and we label these insurance policies  $i$  accordingly.

Assume that  $Z_{i,j}$  are i.i.d. gamma distributed with shape parameter  $\alpha_i$  and scale parameter  $c_i$ , we refer to (2.6). The mean, the variance and the moment generating function of  $Z_{i,j}$  are given by

$$\mathbb{E}[Z_{i,j}] = \frac{\alpha_i}{c_i}, \quad \text{Var}(Z_{i,j}) = \frac{\alpha_i}{c_i^2} \quad \text{and} \quad M_{Z_{i,j}}(r) = \left( \frac{c_i}{c_i - r} \right)^{\alpha_i}, \quad (5.43)$$

where the moment generating function requires  $r < c_i$  to be finite. Assuming that the number of claims  $N_i$  is a known positive integer  $n_i \in \mathbb{N}$ , we see from the moment generating function that  $S_i = \sum_{j=1}^{n_i} Z_{i,j}$  is again gamma distributed with shape parameter  $n_i \alpha_i$  and scale parameter  $c_i$ . We change the notation from  $N_i$  to  $n_i$  to emphasize that the number of claims is treated as a known constant (and also to avoid using the notation of conditional probabilities, here). Finally, we scale  $Y_i = S_i/(n_i \alpha_i) \sim \Gamma(n_i \alpha_i, n_i \alpha_i c_i)$ . This random variable  $Y_i$  has a single-parameter EDF gamma distribution with weight  $v_i = n_i$ , dispersion  $\varphi_i = 1/\alpha_i$  and cumulant function  $\kappa(\theta_i) = -\log(-\theta_i)$ , for  $\theta_i \in \Theta = (-\infty, 0)$ ,

$$\begin{aligned} Y_i &\sim f(y; \theta_i, v_i/\varphi_i) = \exp \left\{ \frac{y\theta_i - \kappa(\theta_i)}{\varphi_i/v_i} + a(y; v_i/\varphi_i) \right\} \\ &= \frac{(-\theta_i \alpha_i v_i)^{v_i \alpha_i}}{\Gamma(v_i \alpha_i)} y^{v_i \alpha_i - 1} \exp \{ -(-\theta_i \alpha_i v_i) y \}, \end{aligned} \quad (5.44)$$

and the canonical parameter is  $\theta_i = -c_i$ . For our GLM analysis we treat the shape parameter  $\alpha_i \equiv \alpha > 0$  as a nuisance parameter that does not depend on the specific policy  $i$ , i.e., we set constant dispersion  $\varphi = 1/\alpha$ , and only the scale parameter  $c_i$  is chosen policy dependent through  $\theta_i = -c_i$ .

Random variable  $Y_i = S_i/(n_i\alpha) \sim \Gamma(n_i\alpha, n_i\alpha c_i)$  gives the reproductive form of the gamma EDF, see Remarks 2.13. In applications, this form is not directly useful because under unknown shape parameter  $\alpha$ , we cannot calculate observations  $Y_i = S_i/(n_i\alpha)$ . For this reason, we parametrize the model differently, here. We consider instead

$$Y_i = S_i/n_i \sim \Gamma(n_i\alpha, n_i c_i). \quad (5.45)$$

This (new) random variable has the same gamma EDF (5.44), we only need to reinterpret the canonical parameter as  $\theta_i = -c_i/\alpha$ . Then, we choose the log-link for  $g$  which implies

$$\mu_i = \mathbb{E}_{\theta_i}[Y_i] = \kappa'(\theta_i) = -\frac{1}{\theta_i} = \exp\{\eta_i\} = \exp\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle,$$

if  $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^{q+1}$  describes the pre-processed features of policy  $i$ . The gamma GLM is now fully specified and can be fitted to the data; from Example 5.5 we know that we have a concave maximization problem. We call this model Gamma GLM1 (with the feature pre-processing as described above). Note that the (constant) dispersion parameter  $\varphi$  cancels in the score equations, thus, we do not need to explicitly specify the nuisance parameter  $\alpha$  to estimate regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ .

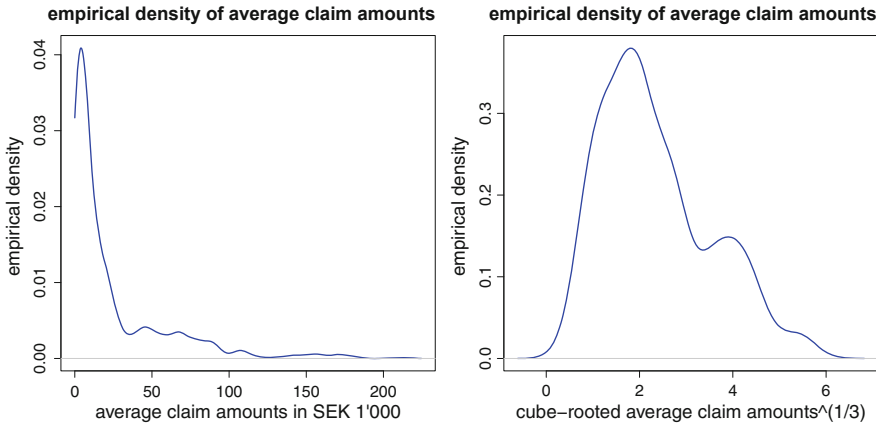
### Maximum Likelihood Estimation and Model Selection

Because we have only few claims data in this Swedish motorcycle example (only  $m = 656$  insurance policies suffer claims), we do not perform a generalization analysis with learning and test samples. In this situation we need all data for model fitting, and model performance is analyzed with AIC and with tenfold cross-validation.

The in-sample deviance loss in the gamma GLM is given by

$$\mathfrak{D}(\mathcal{L}, \hat{\mu}(\cdot)) = \frac{2}{m} \sum_{i=1}^m \frac{n_i}{\varphi} \left( \frac{Y_i - \hat{\mu}(\mathbf{x}_i)}{\hat{\mu}(\mathbf{x}_i)} - \log \left( \frac{Y_i}{\hat{\mu}(\mathbf{x}_i)} \right) \right), \quad (5.46)$$

where  $i$  runs over the policies  $i = 1, \dots, m$  with positive claims  $Y_i = S_i/n_i > 0$ , and  $\hat{\mu}(\mathbf{x}_i) = \exp\langle \hat{\boldsymbol{\beta}}^{\text{MLE}}, \mathbf{x}_i \rangle$  is the MLE estimated regression function. Similar to the Poisson case (5.29), McCullagh–Nelder [265] derive the following behavior



**Fig. 5.10** (lhs) Empirical density of  $Y_i$  and (rhs) empirical density of  $Y_i^{1/3}$

for the gamma unit deviance around its mode, see Section 7.2 and Figure 7.2 in McCullagh–Nelder [265],

$$\mathfrak{d}(Y_i, \mu_i) \approx 9Y_i^{2/3} \left( Y_i^{-1/3} - \mu_i^{-1/3} \right)^2, \quad (5.47)$$

this uses that the log-likelihood is symmetric around its mode for scale  $\mu_i^{-1/3}$ , see Fig. 5.5 (middle). This shows that the gamma deviance scales differently around  $Y_i$  compared to the square loss function. From this we receive an approximation to the deviance residuals (for  $v/\varphi = 1$ )

$$r_i^D = \text{sign}(Y_i - \mu_i) \sqrt{\mathfrak{d}(Y_i, \mu_i)} \approx 3 \left( \left( \frac{Y_i}{\mu_i} \right)^{1/3} - 1 \right) = 3 \frac{Y_i^{1/3} - \mu_i^{1/3}}{\mu_i^{1/3}}. \quad (5.48)$$

This is the cube-root transformation derived by Wilson–Hilferty [383]. This suggests that if the empirical distribution of  $Y_i^{1/3}$  looks roughly Gaussian we can use a gamma distribution. Figure 5.10 gives the empirical densities of  $Y_i$  on the left-hand side and of  $Y_i^{1/3}$  on the right-hand side. The latter looks roughly Gaussian (except of the second mode close to 4), this supports the use of a gamma model.

Listing 5.11 provides the summary statistics of the fitted model Gamma GLM1; note that we integrate the number of claims  $n_i$  through scaling into the `weights`. We have  $q + 1 = 9$  regression parameters, and from this summary statistics we observe that not all variables should be kept in the model. If we perform backward elimination using `drop1` in each step, see Sect. 5.3.3, we first drop `BonusClass` and then `Gender`, resulting in a reduced model with 7 parameters. We call this reduced model Gamma GLM2.

**Listing 5.11** Results in model Gamma GLM1 using the R command glm

```
1 Call:
2 glm(formula = ClaimAmount/ClaimNb ~ OwnerAge + I(OwnerAge^2) +
3   AreaGLM + RiskClass + VehAge + I(VehAge^2) + Gender + BonusClass,
4   family = Gamma(link = "log"), data = mcdata0, weights = ClaimNb)
5
6 Deviance Residuals:
7     Min       1Q   Median       3Q      Max
8 -3.3683  -1.4585  -0.5979   0.4354   3.4763
9
10 Coefficients:
11             Estimate Std. Error t value Pr(>|t|)
12 (Intercept)   8.9737854   0.5532821  16.219 < 2e-16 ***
13 OwnerAge      0.1072781   0.0280862   3.820 0.000147 ***
14 I(OwnerAge^2) -0.0014508   0.0003489  -4.158 3.65e-05 ***
15 AreaGLM      -0.0768512   0.0368284  -2.087 0.037303 *
16 RiskClass     0.0615575   0.0327553   1.879 0.060651 .
17 VehAge       -0.2051148   0.0296184  -6.925 1.05e-11 ***
18 I(VehAge^2)   0.0062649   0.0015946   3.929 9.45e-05 ***
19 GenderMale    0.1085538   0.1673443   0.649 0.516772
20 BonusClass    0.0089004   0.0225371   0.395 0.693029
21 ---
22 Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
23
24 (Dispersion parameter for Gamma family taken to be 1.536577)
25
26 Null deviance: 1368.0 on 655 degrees of freedom
27 Residual deviance: 1126.5 on 647 degrees of freedom
28 AIC: 14922
29
30 Number of Fisher Scoring iterations: 11
```

**Table 5.13** Run times, number of parameters, AICs, Pearson’s dispersion estimate, in-sample losses, tenfold cross-validation losses and the in-sample average claim amounts of the null model (gamma intercept model) and the gamma GLMs

	Run time	# Param.	AIC	Dispersion est. $\hat{\varphi}^P$	In-sample loss on $\mathcal{L}$	Tenfold CV loss $\hat{\mathfrak{D}}^{CV}$	Average amount
Gamma null	–	1 + 1	14’416	2.057	2.085	2.091	24’641
Gamma GLM1	1s	9 + 1	14’277	1.537	1.717	1.752	25’105
Gamma GLM2	1s	7 + 1	14’274	1.544	1.719	1.747	25’130

The results of models Gamma GLM1 and Gamma GLM2 are presented in Table 5.13. We show AICs, Pearson’s dispersion estimate, the in-sample deviance losses on all available data, the corresponding tenfold cross-validation losses, and the average claim amounts.

Firstly, we observe that the GLMs do not meet the balance property. This is implied by the fact that we do not use the canonical link to avoid any sort of difficulty of dealing with the one-sided bounded effective domain  $\Theta = (-\infty, 0)$ . For pricing, the intercept parameter  $\hat{\beta}_0^{MLE}$  should be shifted to eliminate this bias, i.e, we need to shift this parameter under the log-link by  $-\log(25'130/24'641)$  for model Gamma GLM2.

Secondly, the in-sample and tenfold cross-validation losses are not directly comparable to AIC. Observe that we need to know the dispersion parameter  $\varphi$  in order to calculate both of these statistics. For the in-sample and cross-validation

losses we have set  $\varphi = 1$ , thus, all these figures are directly comparable. For AIC we have estimated the dispersion parameter  $\varphi$  with MLE. This is the reason for increasing the number of parameters in Table 5.13 by +1. Moreover, the resulting AICs differ from the ones received from the `R` command `glm`, see, for instance, Listing 5.11. The AIC value in Listing 5.11 does not consider all terms appropriately due to the inclusion of `weights`, this is similar to Remark 5.22, it uses the deviance dispersion estimate  $\hat{\varphi}^D$ , i.e., not the MLE and (still) increases the number of parameters by 1 because the dispersion is estimated. For these reasons, we have implemented our own code for calculating AIC. Both AIC and the tenfold cross-validation losses say that we should give preference to model Gamma GLM2.

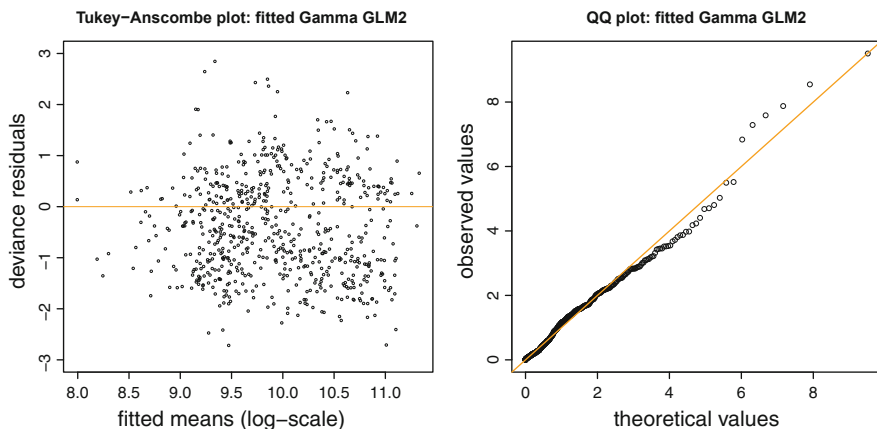
The dispersion estimate in Listing 5.11 corresponds to Pearson's estimate

$$\hat{\varphi}^P = \frac{1}{m - (q + 1)} \sum_{i=1}^m n_i \frac{(Y_i - \hat{\mu}_i)^2}{\hat{\mu}_i^2}. \quad (5.49)$$

We observe that the dispersion estimate is roughly 1.5 which gives an estimate of the shape parameter  $\alpha = 1/\varphi$  of 2/3. A shape parameter less than 1 implies that the density of the gamma distribution is strictly decreasing, see Fig. 2.1. Often this is a sign that the model does not fully fit the data, and if we use this model for simulation we may receive too many observations close to zero compared to the true data. A shape parameter less than 1 may be implied by more heterogeneity in the data compared to what the chosen gamma GLM allows for or by large claims that cannot be explained by the present gamma density structure. Thus, there is some sign here that the data is more heavy-tailed than our model choice suggests. Alternatively, there might be some need to also model the shape parameter with a regression model; this could be done using the vector-valued parameter EF representation of the gamma model, see Sect. 2.1.3. In view of Fig. 5.10 (rhs) it may also be that the feature information is not sufficient to describe the second mode in 4, thus, we probably need more explanatory information to reduce dispersion.

In Fig. 5.11 we give the Tukey–Anscombe plot and a QQ plot. Note that the observations for  $n_i = 1$  follow a gamma distribution with shape parameter  $\alpha$  and scale parameter  $c_i = \alpha/\mu_i = -\alpha\theta_i$ . Thus, if we scale  $Y_i/\mu_i$ , we receive i.i.d. gamma random variables with shape and scale parameters equal to  $\alpha$ . This then allows us for  $n_i = 1$  to plot the empirical distribution of  $Y_i/\hat{\mu}_i$  against  $\Gamma(\alpha, \alpha)$  in a QQ plot where we estimate  $1/\alpha$  by Pearson's dispersion estimate. The Tukey–Anscombe plot looks reasonable, but the QQ plot shows that the gamma model does not entirely fit the data. From this plot we cannot conclude whether the gamma distribution is causing the problem or whether it is a missing term in the regression structure. We only see that the data is over-dispersed, resulting in more heavy-tailed observations than the theoretical gamma model can explain, and a compensation by too many small observations (which is induced by over-dispersion, i.e., a shape parameter smaller than one). In the network chapter we will refine the regression function, keeping the gamma assumption, to understand which modeling part is causing the difficulty.

*Remark 5.26* For the calculation of AIC in Table 5.13 we have used the MLE of the dispersion parameter  $\varphi$ . This is obtained by solving the score equation (5.11) for the



**Fig. 5.11** (lhs) Tukey–Anscombe plot of the fitted model Gamma GLM2, and (rhs) QQ plot of the fitted model Gamma GLM2

gamma case. It is given by, we set  $\alpha = 1/\varphi$  and we calculate the MLE of  $\alpha$  instead,

$$\frac{\partial}{\partial \alpha} \ell_Y(\boldsymbol{\beta}, \alpha) = \sum_{i=1}^n v_i \left[ Y_i h(\mu(x_i)) - \kappa(h(\mu(x_i))) + \log Y_i + \log(\alpha v_i) + 1 - \Psi(\alpha v_i) \right] = 0,$$

where  $\Psi(\alpha) = \Gamma'(\alpha)/\Gamma(\alpha)$  is the digamma function. We calculate the second derivative w.r.t.  $\alpha$ , see also (2.30),

$$\frac{\partial^2}{\partial \alpha^2} \ell_Y(\boldsymbol{\beta}, \alpha) = \sum_{i=1}^n v_i \left[ \frac{1}{\alpha} - v_i \Psi'(\alpha v_i) \right] = \sum_{i=1}^n v_i^2 \left[ \frac{1}{\alpha v_i} - \Psi'(\alpha v_i) \right] < 0 \quad \text{for } \alpha > 0,$$

the negativity follows from Theorem 1 in Alzner [9]. In fact, the function  $\log \alpha - \Psi(\alpha)$  is strictly completely monotonic for  $\alpha > 0$ . This says that the log-likelihood  $\ell_Y(\boldsymbol{\beta}, \alpha)$  is a concave function in  $\alpha > 0$  and the solution to the score equation is unique, giving the MLE of  $\alpha$  and  $\varphi$ , respectively.

### 5.3.8 Lab: Inverse Gaussian GLM for Claim Sizes

We present the inverse Gaussian GLM in this section as a competing model to the gamma GLM studied in the previous section.

#### Infinite Divisibility

In the gamma model above we have used that the total claim amount  $S = \sum_{j=1}^n Z_j$  has a gamma distribution for given claim counts  $N = n > 0$  and i.i.d. gamma claim sizes  $Z_j$ . This property is closely related to divisibility. A random variable  $S$  is called divisible by  $n \in \mathbb{N}$  if there exist i.i.d. random variables  $Z_1, \dots, Z_n$  such



that

$$S \stackrel{(d)}{=} \sum_{j=1}^n Z_j,$$

and  $S$  is called *infinitely divisible* if  $S$  is divisible by  $n$  for all  $n \in \mathbb{N}$ . The EDF is based on parameters  $(\theta, \omega) \in \Theta \times \mathcal{W}$ . Jørgensen [203] gives the following interesting result.

**Theorem 5.27 (Theorem 3.7 in Jørgensen [203], Without Proof)** *Choose a member of the EDF with parameter set  $\Theta \times \mathcal{W}$ . Then*

- *the index set  $\mathcal{W}$  is an additive semi-group and  $\mathbb{N} \subseteq \mathcal{W} \subseteq \mathbb{R}_+$ , and*
- *the members of the chosen EDF are infinitely divisible if and only if  $\mathcal{W} = \mathbb{R}_+$ .*

This theorem tells us how to aggregate and disaggregate within EDFs, e.g., the Poisson, gamma and inverse Gaussian models are infinitely divisible, and the binomial distribution is divisible by  $n$  with the disaggregated random variables belonging to the same EDF and the same canonical parameter, see Sect. 2.2.2. In particular, we also refer to Corollary 2.15 on the convolution property.

### Inverse Gaussian Generalized Linear Model

Alternatively to the gamma GLM one often explores an inverse Gaussian GLM which has a cubic variance function  $V(\mu) = \mu^3$ . We bring this inverse Gaussian model into the same form as the gamma model of Sect. 5.3.7, so that we can aggregate claims within insurance policies. The mean, the variance and the moment generating function of an inverse Gaussian random variable  $Z_{i,j}$  with parameters  $\alpha_i, c_i > 0$  are given by

$$\mathbb{E}[Z_{i,j}] = \frac{\alpha_i}{c_i}, \quad \text{Var}(Z_{i,j}) = \frac{\alpha_i}{c_i^3} \quad \text{and} \quad M_{Z_{i,j}}(r) = \exp \left\{ \alpha_i \left[ c_i - \sqrt{c_i^2 - 2r} \right] \right\},$$

where the moment generating function requires  $r < c_i^2/2$  to be finite. From the moment generating function we see that  $S_i = \sum_{j=1}^{n_i} Z_{i,j}$  is inverse Gaussian distributed with parameters  $n_i \alpha_i$  and  $c_i$ . Finally, we scale  $Y_i = S_i / (n_i \alpha_i)$  which provides us with an inverse Gaussian distribution with parameters  $n_i^{1/2} \alpha_i^{1/2}$  and  $n_i^{1/2} \alpha_i^{1/2} c_i$ . This random variable  $Y_i$  has a single-parameter EDF inverse Gaussian distribution in its reproductive form, namely,

$$\begin{aligned} Y_i \sim f(y; \theta_i, v_i / \varphi_i) &= \exp \left\{ \frac{y \theta_i - \kappa(\theta_i)}{\varphi_i / v_i} + a(y; v_i / \varphi_i) \right\} \\ &= \frac{\alpha_i^{1/2}}{\sqrt{\frac{2\pi}{v_i}} y^3} \exp \left\{ -\frac{\alpha_i}{2y/v_i} \left( 1 - \sqrt{-2\theta_i y} \right)^2 \right\}, \end{aligned} \quad (5.50)$$

with cumulant function  $\kappa(\theta) = -\sqrt{-2\theta}$  for  $\theta \in \Theta = (-\infty, 0]$ , weight  $v_i = n_i$ , dispersion parameter  $\varphi_i = 1/\alpha_i$  and canonical parameter  $\theta_i = -c_i^2/2$ .

Similarly to the gamma case, this representation is not directly useful if the parameter  $\alpha_i$  is not known. Therefore, we parametrize this model differently. Namely, we consider

$$Y_i = S_i/n_i \sim \text{InvGauss}\left(n_i^{1/2}\alpha_i, n_i^{1/2}c_i\right). \quad (5.51)$$

This re-scaled random variable has that same inverse Gaussian EDF (5.50), but we need to re-interpret the parameters. We have dispersion parameter  $\varphi_i = 1/\alpha_i^2$  and canonical parameter  $\theta_i = -c_i^2/(2\alpha_i^2)$ . For our GLM analysis we will treat the parameter  $\alpha_i \equiv \alpha > 0$  as a nuisance parameter that does not depend on the specific policy  $i$ . Thus, we have constant dispersion  $\varphi = 1/\alpha^2$  and only the scale parameter  $c_i$  is assumed to be policy dependent through the canonical parameter  $\theta_i = -c_i^2/(2\alpha^2)$ .

We are now in the same situation as in the gamma case in Sect. 5.3.7. We choose the log-link for  $g$  which implies

$$\mu_i = \mathbb{E}_{\theta_i}[Y_i] = \kappa'(\theta_i) = \frac{1}{\sqrt{-2\theta_i}} = \exp\{\eta_i\} = \exp\langle \boldsymbol{\beta}, \mathbf{x}_i \rangle,$$

for  $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^{q+1}$  describing the pre-processed features of policy  $i$ . We use the same feature pre-processing as in model Gamma GLM2, and we call this resulting model IG GLM2. Again the constant dispersion parameter  $\varphi = 1/\alpha^2$  cancels in the score equations, thus, we do not need to explicitly specify the nuisance parameter  $\alpha$  to estimate the regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ . However, there is an important difference to the gamma GLM, namely, as stated in Example 5.6, we do not have a concave maximization problem and Fisher's scoring method needs a suitable initial value. We start the fitting algorithm in the parameters of model Gamma GLM2.

The in-sample deviance loss in the inverse Gaussian GLM is given by

$$\mathfrak{D}(\mathcal{L}, \widehat{\mu}(\cdot)) = \frac{1}{m} \sum_{i=1}^m \frac{n_i}{\varphi} \frac{(Y_i - \widehat{\mu}(\mathbf{x}_i))^2}{\widehat{\mu}(\mathbf{x}_i)^2 Y_i}, \quad (5.52)$$

where  $i$  runs over the policies  $i = 1, \dots, m$  with positive claims  $Y_i = S_i/n_i > 0$ , and  $\widehat{\mu}(\mathbf{x}_i) = \exp\langle \widehat{\boldsymbol{\beta}}^{\text{MLE}}, \mathbf{x}_i \rangle$  is the MLE estimated regression function. The unit deviances behave as

$$\mathfrak{d}(Y_i, \mu_i) = Y_i \left( Y_i^{-1} - \mu_i^{-1} \right)^2, \quad (5.53)$$

**Table 5.14** Run times, number of parameters, AICs, in-sample losses, tenfold cross-validation losses and the in-sample average claim amounts of the null gamma model, model Gamma GLM2, the null inverse Gaussian model, and model inverse Gaussian GLM2; the deviance losses use unit dispersion  $\varphi = 1$

	Run time	# Param.	AIC	In-sample loss on $\mathcal{L}$	Tenfold CV loss $\widehat{\mathfrak{D}}^{\text{CV}}$	Average amount
Gamma null	–	1 + 1	14'416	2.085	2.091	24'641
Gamma GLM2	1 s	7 + 1	14'274	1.719	1.747	25'130
IG null	–	1 + 1	14'715	$5.012 \cdot 10^{-4}$	$5.016 \cdot 10^{-4}$	24'641
IG GLM2	1 s	7 + 1	14'686	$4.793 \cdot 10^{-4}$	$4.820 \cdot 10^{-4}$	32'268

note that the log-likelihood is symmetric around its mode for scale  $\mu_i^{-1}$ , see Fig. 5.5 (rhs). From this we receive deviance residuals (for  $v/\varphi = 1$ )

$$r_i^{\text{D}} = \text{sign}(Y_i - \mu_i) \sqrt{\mathfrak{d}(Y_i, \mu_i)} = Y_i^{1/2} \left( \mu_i^{-1} - Y_i^{-1} \right).$$

Thus, these residuals behave as  $Y_i^{1/2}$  for  $Y_i \rightarrow \infty$  (and fixed  $\mu_i^{-1}$ ), which is more heavy-tailed than the cube-root behavior  $Y_i^{1/3}$  in the gamma case, see (5.48). Another difference to the gamma case is that the deviance loss (5.52) is not scale-invariant, see also (11.4), below.

We revisit the example of Table 5.13, but we replace the gamma distribution by the inverse Gaussian distribution. The results in Table 5.14 show that the inverse Gaussian model is not fully competitive on this data set. In view of (5.43) we observe that the coefficient of variation (standard deviation divided by mean) is in the gamma model given by  $1/\sqrt{\alpha}$ , thus, in the gamma model this coefficient of variation is independent of the expected claim size  $\mu_i$  and only depends on the shape parameter  $\alpha$ . In the inverse Gaussian model the coefficient of variation is given by

$$\text{Vco}(Z_{i,j}) = \frac{\sqrt{\text{Var}(Z_{i,j})}}{\mathbb{E}[Z_{i,j}]} = \frac{\sqrt{\mu_i}}{\alpha},$$

thus, it monotonically increases in the expected claim size  $\mu_i$ . It seems that this structure is not fully suitable for this data set, i.e., there is no indication that the coefficient of variation increases in the expected claim size. We come back to a comparison of the gamma and the inverse Gaussian model in Sect. 11.1, below.

### 5.3.9 Log-Normal Model for Claim Sizes: A Short Discussion

Another way to improve the gamma model of Sect. 5.3.7 could be to use a log-normal distribution instead. In the above situation this does not work because the observations are not in the right format. If the claim observations  $Z_{i,j}$  are log-

normally distributed, then  $\log(Z_{i,j})$  are normally distributed. Unfortunately, in our Swedish motorcycle data set we do not have individual claim observations  $Z_{i,j}$ , but the provided information is aggregated over all claims per insurance policy, i.e.,  $S_i = \sum_{j=1}^{N_i} Z_{i,j}$ . Therefore, there is no possibility here to challenge the gamma framework of Sect. 5.3.7 with a corresponding log-normal framework, because the log-normal framework is not closed under summation of i.i.d. log-normally distributed random variables.

We would like to give some remarks that concern calculations on the log-scale (or any other strictly increasing and concave transformation of the original data). For the log-normal distribution, as well as in similar cases like the log-gamma distribution, one works with logged observations  $Y_i = \log(Z_i)$ . This is a strictly monotone transformation and the MLEs in the log-normal model based on observations  $Z_i$  and in the normal model based on observations  $Y_i = \log(Z_i)$  coincide. This can be seen from the following calculation. We start from the log-normal density on  $\mathbb{R}_+$ , and we do a transformation of variable  $z > 0 \mapsto y = \log(z) \in \mathbb{R}$  with  $dy = dz/z$

$$\begin{aligned} f_{\text{LN}}(z; \mu, \sigma^2) dz &= \frac{1}{\sqrt{2\pi\sigma^2}} \frac{1}{z} \exp \left\{ -\frac{1}{2\sigma^2} (\log(z) - \mu)^2 \right\} dz \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left\{ -\frac{1}{2\sigma^2} (y - \mu)^2 \right\} dy = f_{\Phi}(y; \mu, \sigma^2) dy. \end{aligned}$$

From this we see that the MLEs will coincide.

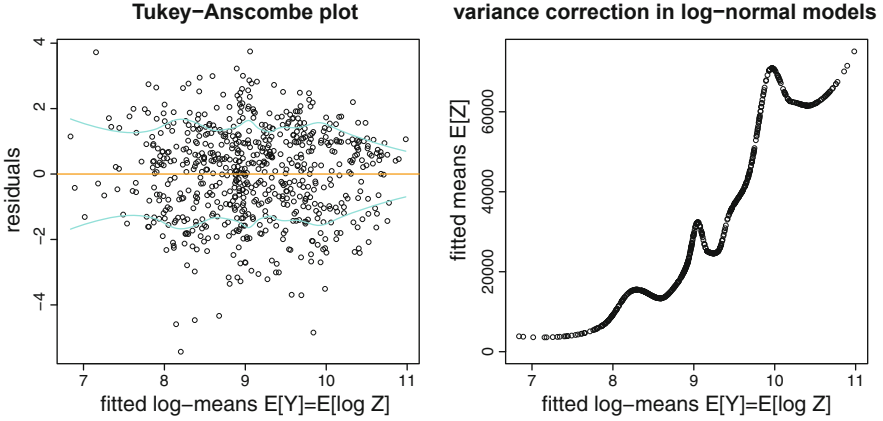
In many situations, one assumes that  $\sigma^2 > 0$  is a given nuisance parameter, and one models  $\mathbf{x} \mapsto \mu(\mathbf{x})$  with a GLM within the single-parameter EDF. In the log-normal/Gaussian case one typically chooses the canonical link on the log-scale which is the identity function. This then allows one to perform a classical linear regression for  $\mu(\mathbf{x}) = \langle \boldsymbol{\beta}, \mathbf{x} \rangle$  using the logged observations  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top = (\log(Z_1), \dots, \log(Z_n))^\top$ , and the corresponding MLE is given by

$$\hat{\boldsymbol{\beta}}^{\text{MLE}} = (\mathfrak{X}^\top \mathfrak{X})^{-1} \mathfrak{X}^\top \mathbf{Y}, \quad (5.54)$$

for full rank  $q + 1 \leq n$  design matrix  $\mathfrak{X}$ . Note that in this case we have a closed-form solution for the MLE of  $\boldsymbol{\beta}$ . This is called the homoskedastic case because all observations  $Y_i$  are assumed to have the same variance  $\sigma^2$ , otherwise, in the heteroskedastic case, we would still have to include the covariance matrix.

Since we work with the canonical link on the log-scale we have the balance property on the log-scale, see Corollary 5.7. Thus, we receive unbiasedness

$$\sum_{i=1}^n \mathbb{E}_{\boldsymbol{\beta}} \left[ \mathbb{E}_{\hat{\boldsymbol{\beta}}^{\text{MLE}}} [Y_i] \right] = \sum_{i=1}^n \mathbb{E}_{\boldsymbol{\beta}} \left[ \langle \hat{\boldsymbol{\beta}}^{\text{MLE}}, \mathbf{x}_i \rangle \right] = \sum_{i=1}^n \mathbb{E}_{\boldsymbol{\beta}} [Y_i] = \sum_{i=1}^n \mu(\mathbf{x}_i). \quad (5.55)$$



**Fig. 5.12** (lhs) Tukey–Anscombe plot of the fitted Gaussian model  $\hat{\mu}(\mathbf{x}_i)$  on the logged claim sizes  $Y_i = \log(Z_i)$ , and (rhs) estimated means  $\hat{\mu}_{Z_i}$  as a function of  $\hat{\mu}(\mathbf{x}_i)$  considering heteroskedasticity  $\hat{\sigma}(\mathbf{x}_i)$

If we move back to the original scale of the observations  $Z_i$  we receive from the log-normal assumption

$$\mathbb{E}_{(\hat{\beta}^{\text{MLE}}, \sigma^2)}[Z_i] = \exp \left\{ \langle \hat{\beta}^{\text{MLE}}, \mathbf{x}_i \rangle + \sigma^2/2 \right\}.$$

Therefore, we need to adjust with the nuisance parameter  $\sigma^2$  for the back-transformation to the original observation scale. At this point, typically, the difficulties start. Often, a good back-transformation involves a feature dependent variance parameter  $\sigma^2(\mathbf{x}_i)$ , thus, in many practical applications the homoskedasticity assumption is not fulfilled, and a constant variance parameter choice leads to a poor model on the original observation scale.

A suitable estimation of  $\sigma^2(\mathbf{x}_i)$  may turn out to be rather difficult. This is illustrated in Fig. 5.12. The left-hand side of this figure shows the Tukey–Anscombe plot of the homoskedastic case providing unscaled ( $\sigma^2 \equiv 1$ ) (Pearson’s) residuals on the log-scale

$$r_i^{\text{P}} = \log(Z_i) - \hat{\mu}(\mathbf{x}_i) = Y_i - \hat{\mu}(\mathbf{x}_i).$$

The light-blue color shows an insurance policy dependent standard deviation estimate  $\hat{\sigma}(\mathbf{x}_i)$ . In our case this estimate is non-monotone in  $\hat{\mu}(\mathbf{x}_i)$  (which is quite common on real data). Using this estimate we can estimate the means of the log-normal random variables by

$$\hat{\mu}_{Z_i} = \hat{\mathbb{E}}[Z_i] = \exp \left\{ \hat{\mu}(\mathbf{x}_i) + \hat{\sigma}(\mathbf{x}_i)^2/2 \right\}.$$

The right-hand side of Fig. 5.12 plots these estimated means  $\hat{\mu}_{Z_i}$  against the estimated means  $\hat{\mu}(x_i)$  on the log-scale. We observe a graph that is non-monotone, implied by the non-monotonicity of the standard deviation estimate  $\hat{\sigma}(x_i)$  as a function of  $\hat{\mu}(x_i)$ . This non-monotonicity is not bad per se, as we still have a proper statistical model, however, it might be rather counter-intuitive and difficult to explain. For this reason it is advisable to directly model the expected value by one single function, and not to decompose it into different regression functions.

Another important point to be considered is that for model selection using AIC we have to work on the same scale for all models. Thus, if we use a gamma model to model  $Z_i$ , then for an AIC selection we need to evaluate also the log-normal model on that scale. This can be seen from the justification in Sect. 4.2.3.

Finally, we focus on unbiasedness. Note that on the log-scale we have unbiasedness (5.55) through the balance property. Unfortunately, this does not carry over to the original scale. We give a small example, where we assume that there is neither any uncertainty about the distributional model nor about the nuisance parameter. That is, we assume that  $Z_i$  are i.i.d. log-normally distributed with parameters  $\mu$  and  $\sigma^2$ , where only  $\mu$  is unknown. The MLE of  $\mu$  is given by

$$\hat{\mu}^{\text{MLE}} = \frac{1}{n} \sum_{i=1}^n \log(Z_i) \sim \mathcal{N}(\mu, \sigma^2/n).$$

In this case we have

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{(\mu, \sigma^2)} [\mathbb{E}_{(\hat{\mu}^{\text{MLE}}, \sigma^2)} [Z_i]] &= \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{(\mu, \sigma^2)} [\exp\{\hat{\mu}^{\text{MLE}}\}] \exp\{\sigma^2/2\} \\ &= \exp\left\{\mu + (1 + n^{-1})\sigma^2/2\right\} \\ &> \exp\left\{\mu + \sigma^2/2\right\} = \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{(\mu, \sigma^2)} [Z_i]. \end{aligned}$$

Volatility in parameter estimation  $\hat{\mu}^{\text{MLE}}$  leads to a positive bias in this case. Note that we have assumed full knowledge of the distributional model (i.i.d. log-normal) and the nuisance parameter  $\sigma^2$  in this calculation. If, for instance, we do not know the true nuisance parameter and we work with (deterministic)  $\tilde{\sigma}^2 \ll \sigma^2$  and  $n > 1$ , we can get a negative bias

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{(\mu, \sigma^2)} [\mathbb{E}_{(\hat{\mu}^{\text{MLE}}, \tilde{\sigma}^2)} [Z_i]] &= \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{(\mu, \sigma^2)} [\exp\{\hat{\mu}^{\text{MLE}}\}] \exp\{\tilde{\sigma}^2/2\} \\ &= \exp\left\{\mu + \sigma^2/(2n) + \tilde{\sigma}^2/2\right\} \\ &< \exp\left\{\mu + \sigma^2/2\right\} = \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{(\mu, \sigma^2)} [Z_i]. \end{aligned}$$

This shows that working on the log-scale is rather difficult because the back-transformation is far from being trivial, and for unknown nuisance parameter not even the sign of the bias is clear. Similar considerations apply to the frequently used Box–Cox transformation [48] for  $\chi \neq 1$

$$Z_i \mapsto Y_i = \frac{Z_i^\chi - 1}{\chi}.$$

For this reason, if unbiasedness is a central requirement (like in insurance pricing) non-linear transformations should only be used with great care (and only if necessary).

## 5.4 Quasi-Likelihoods

Above we have been mentioning the notion of over-dispersed Poisson models. This naturally leads to so-called quasi-Poisson models and quasi-likelihoods. The framework of quasi-likelihoods has been introduced by Wedderburn [376]. In this section we give the main idea behind quasi-likelihoods, and for a more detailed treatment and mathematical results we refer to Chapter 8 of McCullagh–Nelder [265].

In Sect. 5.1.4 we have discussed the estimation of GLMs. This has been based on the explicit knowledge of the full log-likelihood function  $\ell_Y(\boldsymbol{\beta})$  for given data  $\mathbf{Y}$ . This has allowed us to calculate the score equations  $s(\boldsymbol{\beta}, \mathbf{Y}) = \nabla_{\boldsymbol{\beta}} \ell_Y(\boldsymbol{\beta}) = 0$  whose solutions (Z-estimators) contain the MLE for  $\boldsymbol{\beta}$ . The solutions of the score equations themselves, using Fisher’s scoring method, no longer need the explicit functional form of the log-likelihood, but they are only based on the first and second moments, see (5.9) and Remarks 5.4. Thus, all models where these first two moments coincide will provide the same MLE for the regression parameter  $\boldsymbol{\beta}$ ; this is also the explanation behind the IRLS algorithm. Moreover, the first two moments are sufficient for prediction and uncertainty quantification based on mean squared errors, and they are also sufficient to quantify asymptotic normality. This is exactly what motivates the quasi-likelihood considerations, and these considerations are also related to the quasi-generalized pseudo maximum likelihood estimator (QPMLE) that we are going to discuss in Theorem 11.8, below.

Assume that  $\mathbf{Y}$  is a random vector having first moment  $\boldsymbol{\mu} \in \mathbb{R}^n$ , positive definite variance function  $V(\boldsymbol{\mu}) \in \mathbb{R}^{n \times n}$  and dispersion parameter  $\varphi$ . The quasi-(log-)likelihood function  $\ell_Y(\boldsymbol{\mu})$  assumes that its gradient is given by

$$\nabla_{\boldsymbol{\mu}} \ell_Y(\boldsymbol{\mu}) = \frac{1}{\varphi} V(\boldsymbol{\mu})^{-1} (\mathbf{Y} - \boldsymbol{\mu}).$$

In case of a diagonal variance function  $V(\boldsymbol{\mu})$  this relates to the score (5.9). The remaining step is to model the mean parameter  $\boldsymbol{\mu} = \boldsymbol{\mu}(\boldsymbol{\beta}) \in \mathbb{R}^n$  as a function of a lower dimensional regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ , we also refer to Fig. 5.2. For

this last step we assume that the Jacobian  $B \in \mathbb{R}^{n \times (q+1)}$  of  $d\mu/d\beta$  has full rank  $q + 1$ . The score equations for  $\beta$  and given observations  $Y$  then read as

$$\frac{1}{\varphi} B^\top V(\mu(\beta))^{-1} (Y - \mu(\beta)) = 0.$$

This is of exactly the same structure as the score equations in Proposition 5.1, and the roots are found by using the IRLS algorithm for  $t \geq 0$ , see (5.12),

$$\hat{\beta}^{(t)} \mapsto \hat{\beta}^{(t+1)} = \left( B^\top V(\hat{\mu}^{(t)})^{-1} B \right)^{-1} B^\top V(\hat{\mu}^{(t)})^{-1} \left( B \hat{\beta}^{(t)} + Y - \hat{\mu}^{(t)} \right),$$

where  $\hat{\mu}^{(t)} = \mu(\hat{\beta}^{(t)})$ .

We conclude with the following points about quasi-likelihoods:

- For regression parameter estimation within the quasi-likelihood framework it is sufficient to know the structure of the first two moments  $\mu(\beta) \in \mathbb{R}^n$  and  $V(\mu) \in \mathbb{R}^{n \times n}$  as well as the score equations. Thus, we do not need to explicitly specify a distributional family for the observations  $Y$ . This structure of the first two moments is then sufficient for their estimation using the IRLS algorithm, i.e., we receive the predictors within this framework.
- Since we do not specify the full distribution of  $Y$  we can neither simulate from this model nor can we calculate quantities where the full log-likelihood of the model needs to be known. For example, we cannot calculate AIC in a quasi-likelihood model.
- The quasi-likelihood model is characterized by the functional forms of  $\mu(\beta)$  and  $V(\mu)$ . The former plays the role of the link function and the linear predictor in the GLM, and the latter plays the role of the variance function within the EDF which is characterized through the cumulant function  $\kappa$ . For instance, if we assume to have a diagonal matrix

$$V(\mu) = \text{diag}(V(\mu_1), \dots, V(\mu_n)),$$

then, the choice of the variance function  $\mu \mapsto V(\mu)$  describes the explicit selection of the quasi-likelihood model. If we choose the power variance function  $V(\mu) = \mu^p$ ,  $p \notin (0, 1)$ , we have a quasi-Tweedie's model.

- For prediction uncertainty evaluation we also need an estimate of the dispersion parameter  $\varphi > 0$ . Since we do not know the full likelihood in this approach, Pearson's estimate  $\hat{\varphi}^P$  is the only option we have to estimate  $\varphi$ .
- For asymptotic normality results and hypothesis testing within the quasi-likelihood framework we refer to Section 8.4 of McCullagh–Nelder [265].



## 5.5 Double Generalized Linear Model

In the derivations above we have treated the dispersion parameter  $\varphi$  in the GLM as a nuisance parameter. In the case of a homogeneous dispersion parameter it can be canceled in the score equations for MLE, see (5.9). Therefore, it does not influence MLE, and in a subsequent step this nuisance parameter can still be estimated using, e.g., Pearson's or deviance residuals, see Sect. 5.3.1 and Remark 5.26. In some examples we may have systematic effects in the dispersion parameter, too. In this case the above approach will not work because a heterogeneous dispersion parameter no longer cancels in the score equations. This has been considered in Smyth [341] and Smyth–Verbyla [343]. The heterogeneous dispersion situation is of general interest for GLMs, and it is of particular interest for Tweedie's CP GLM if we interpret Tweedie's distribution [358] as a CP model with i.i.d. gamma claim sizes, see Proposition 2.17; we also refer to Jørgensen–de Souza [204], Smyth–Jørgensen [342] and Delong et al. [94].

### 5.5.1 The Dispersion Submodel

We extend model assumption (5.1) by assuming that also the dispersion parameter  $\varphi_i$  is policy  $i$  dependent. Assume that all random variables  $Y_i$  are independent and have densities w.r.t. a  $\sigma$ -finite measure  $\nu$  on  $\mathbb{R}$  given by

$$Y_i \sim f(y_i; \theta_i, v_i/\varphi_i) = \exp \left\{ \frac{y_i \theta_i - \kappa(\theta_i)}{\varphi_i/v_i} + a(y_i; v_i/\varphi_i) \right\},$$

for  $1 \leq i \leq n$ , with canonical parameters  $\theta_i \in \overset{\circ}{\Theta}$ , exposures  $v_i > 0$  and dispersion parameters  $\varphi_i > 0$ . As in (5.5) we assume that every policy  $i$  is equipped with feature information  $\mathbf{x}_i \in \mathcal{X}$  such that for a given link function  $g : \mathcal{M} \rightarrow \mathbb{R}$  we can model its mean as

$$\mathbf{x}_i \mapsto g(\mu_i) = g(\mu(\mathbf{x}_i)) = g(\mathbb{E}_{\theta(\mathbf{x}_i)}[Y_i]) = \eta_i = \eta(\mathbf{x}_i) = \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle. \quad (5.56)$$

This provides us with log-likelihood function for observation  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$

$$\boldsymbol{\beta} \mapsto \ell_{\mathbf{Y}}(\boldsymbol{\beta}) = \sum_{i=1}^n \frac{v_i}{\varphi_i} \left[ Y_i h(\mu(\mathbf{x}_i)) - \kappa(h(\mu(\mathbf{x}_i))) \right] + a(Y_i; v_i/\varphi_i),$$

with canonical link  $h = (\kappa')^{-1}$ . The difference to (5.7) is that the dispersion parameter  $\varphi_i$  now depends on the insurance policy which requires additional modeling. We choose a second strictly monotone and smooth link function  $g_\varphi :$

$\mathbb{R}_+ \rightarrow \mathbb{R}$ , and we express the dispersion of policy  $1 \leq i \leq n$  by

$$g_\varphi(\varphi_i) = g_\varphi(\varphi(\mathbf{z}_i)) = \langle \boldsymbol{\gamma}, \mathbf{z}_i \rangle, \quad (5.57)$$

where  $\mathbf{z}_i$  is the feature of policy  $i$ , which may potentially differ from  $\mathbf{x}_i$ . The rationale behind this different feature is that different information might be relevant for modeling the dispersion parameter, or feature information might be differently pre-processed compared to the response  $Y_i$ . We now need to estimate two regression parameters  $\boldsymbol{\beta}$  and  $\boldsymbol{\gamma}$  in this approach on possibly differently pre-processed feature information  $\mathbf{x}_i$  and  $\mathbf{z}_i$  of policy  $i$ . In general, this is not easily doable because the term  $a(Y_i; v_i/\varphi_i)$  of the log-likelihood of  $Y_i$  may have a complicated structure (or may not be available in closed form like in Tweedie's CP model).

### 5.5.2 Saddlepoint Approximation

We reformulate the EDF density using the unit deviance  $\mathfrak{d}(Y, \mu)$  defined in (2.25); we drop the lower index  $i$  for the moment. Set  $\theta = h(\mu) \in \mathring{\Theta}$  for the canonical link  $h$ , then

$$\begin{aligned} f(y; \theta, v/\varphi) &= \exp \left\{ \frac{v}{\varphi} [yh(\mu) - \kappa(h(\mu))] + a(y; v/\varphi) \right\} \\ &= \exp \left\{ \frac{v}{\varphi} [yh(y) - \kappa(h(y))] + a(y; v/\varphi) \right\} \exp \left\{ -\frac{1}{2\varphi/v} \mathfrak{d}(y, \mu) \right\} \\ &\stackrel{\text{def.}}{=} a^*(y; \omega) \exp \left\{ -\frac{\omega}{2} \mathfrak{d}(y, \mu) \right\}, \end{aligned} \quad (5.58)$$

with  $\omega = v/\varphi \in \mathcal{W}$ . This corresponds to (2.27), and it brings the EDF density into a Gaussian-looking form. A general difficulty is that the term  $a^*(y; \omega)$  may have a complicated structure or may not be given in closed form. Therefore, we consider its saddlepoint approximation; this is based on Section 3.5 of Jørgensen [203].

Suppose that we are in the absolutely continuous EDF case and that  $\kappa$  is steep. In that case  $Y \in \mathcal{M}$ , a.s., and the variance function  $y \mapsto V(y)$  is well-defined for all observations  $Y = y$ , a.s. Based on Daniels [87], Barndorff-Nielsen–Cox [24] proved the following statement, see Theorem 3.10 in Jørgensen [203]: assume there exists  $\omega_0 \in \mathcal{W}$  such that for all  $\omega > \omega_0$  the density (5.58) is bounded. Then, the following saddlepoint approximation is uniform on compact subsets of the support  $\mathfrak{T}$  of  $Y$

$$f(y; \theta, v/\varphi) = \left( \frac{2\pi\varphi}{v} V(y) \right)^{-1/2} \exp \left\{ -\frac{1}{2\varphi/v} \mathfrak{d}(y, \mu) \right\} (1 + O(\varphi/v)), \quad (5.59)$$

as  $\varphi/v \rightarrow 0$ . What makes this saddlepoint approximation attractive is that we can get rid of a complicated function  $a^*(y; \omega)$  by a neat approximation  $(\frac{2\pi\varphi}{v} V(y))^{-1/2}$  for sufficiently large volumes  $v$ , and at the same time, this does not affect the unit deviance  $\mathfrak{d}(y, \mu)$ , preserving the estimation properties of  $\mu$ . The discrete counterpart is given in Theorem 3.11 of Jørgensen [203].

Using saddlepoint approximation (5.59) we receive an approximate log-likelihood function

$$\ell_Y(\mu, \varphi) \approx \frac{1}{2} \left[ -\varphi^{-1} v \mathfrak{d}(Y, \mu) - \log(\varphi) \right] - \frac{1}{2} \log \left( \frac{2\pi}{v} V(Y) \right).$$

This approximation has an attractive form for dispersion estimation because it gives an approximate EDF for observation  $\mathfrak{d} \stackrel{\text{def}}{=} v \mathfrak{d}(Y, \mu)$ , for given  $\mu$ . Namely, for canonical parameter  $\phi = -\varphi^{-1} < 0$  we have approximation

$$\ell_Y(\mu, \phi) \approx \frac{\mathfrak{d}\phi - (-\log(-\phi))}{2} - \frac{1}{2} \log \left( \frac{2\pi}{v} V(Y) \right). \quad (5.60)$$

The right-hand side has the structure of a gamma EDF for observation  $\mathfrak{d}$  with canonical parameter  $\phi < 0$ , cumulant function  $\kappa_\varphi(\phi) = -\log(-\phi)$  and dispersion parameter 2. Thus, we have the structure of an approximate gamma model on the right-hand side of (5.60) with, for given  $\mu$ ,

$$\mathbb{E}_\phi[\mathfrak{d}|\mu] \approx \kappa'_\varphi(\phi) = -\frac{1}{\phi} = \varphi, \quad (5.61)$$

$$\text{Var}_\phi(\mathfrak{d}|\mu) \approx 2\kappa''_\varphi(\phi) = 2\frac{1}{\phi^2} = 2\varphi^2. \quad (5.62)$$

These statements say that for given  $\mu$  and assuming that the saddlepoint approximation is sufficiently accurate,  $\mathfrak{d}$  is approximately gamma distributed with shape parameter  $1/2$  and canonical parameter  $\phi$  (which relates to the dispersion  $\varphi$  in the mean parametrization). Thus, we can estimate  $\phi$  and  $\varphi$ , respectively, with a (second) GLM from (5.60), for given mean parameter  $\mu$ .

#### Remarks 5.28

- The accuracy of the saddlepoint approximation is discussed in Section 3.2 of Smyth–Verbyla [343]. The saddlepoint approximation is exact in the Gaussian and the inverse Gaussian case. In the Gaussian case, we have log-likelihood

$$\ell_Y(\mu, \phi) = \frac{\mathfrak{d}\phi - (-\log(-\phi))}{2} - \frac{1}{2} \log \left( \frac{2\pi}{v} \right),$$

with variance function  $V(Y) = 1$ . In the inverse Gaussian case, we have log-likelihood

$$\ell_Y(\mu, \phi) = \frac{\mathfrak{d}\phi - (-\log(-\phi))}{2} - \frac{1}{2} \log\left(\frac{2\pi}{v} Y^3\right),$$

with variance function  $V(Y) = Y^3$ . Thus, in the Gaussian case and in the inverse Gaussian case we have a gamma model for  $\mathfrak{d}$  with mean  $\varphi$  and shape parameter  $1/2$ , for given  $\mu$ ; for a related result we also refer to Theorem 3 of Blæsild–Jensen [38]. For Tweedie's models with  $p \geq 1$ , one can show that the relative error of the saddlepoint approximation is a non-increasing function of the squared coefficient of variation  $\tau = \frac{\varphi}{v} V(y)/y^2 = \frac{\varphi}{v} y^{p-2}$ , leading to small approximation errors if  $\varphi/v$  is sufficiently small; typically one requires  $\tau < 1/3$ , see Section 3.2 of Smyth–Verbyla [343].

- The saddlepoint approximation itself does not provide a density because in general the term  $O(\varphi/v)$  in (5.59) is non-zero. Nelder–Pregibon [282] renormalized the saddlepoint approximation to a proper density and studied its properties.
- In the gamma EDF case, the saddlepoint approximation would not be necessary because this case can still be solved in closed form. In fact, in the gamma EDF case we have log-likelihood, set  $\phi = -v/\varphi < 0$ ,

$$\ell_Y(\mu, \phi) = \frac{\phi \mathfrak{d}(Y, \mu) - \chi(\phi)}{2} - \log Y, \quad (5.63)$$

with  $\chi(\phi) = 2(\log \Gamma(-\phi) + \phi \log(-\phi) - \phi)$ . For given  $\mu$ , this is an EDF for  $\mathfrak{d}(Y, \mu)$  with cumulant function  $\chi$  on the effective domain  $(-\infty, 0)$ . This provides us with expected value and variance

$$\begin{aligned} \mathbb{E}_\phi[\mathfrak{d}(Y, \mu)|\mu] &= \chi'(\phi) = 2(-\Psi(-\phi) + \log(-\phi)) \approx -\frac{1}{\phi}, \\ \text{Var}_\phi(\mathfrak{d}(Y, \mu)|\mu) &= 2\chi''(\phi) = 4\left(\Psi'(-\phi) - \frac{1}{-\phi}\right), \end{aligned}$$

with digamma function  $\Psi$  and the approximation exactly refers to the saddlepoint approximation; for the variance statement we also refer to Fisher's information (2.30). For receiving more accurate mean approximations one can consider higher order terms, e.g., the second order approximation is  $\chi'(\phi) \approx -1/\phi + 1/(6\phi^2)$ . In fact, from the saddlepoint approximation (5.60) and from the exact formula (5.63) we receive in the gamma case Stirling's formula

$$\Gamma(\gamma) \approx \sqrt{2\pi} \gamma^{\gamma-1/2} e^{-\gamma}.$$

In the subsequent examples we will just use the saddlepoint approximation also in the gamma EDF case.

### 5.5.3 Residual Maximum Likelihood Estimation

The saddlepoint approximation (5.60) proposes to alternate MLE of  $\boldsymbol{\beta}$  for the mean model (5.56) and of  $\boldsymbol{\gamma}$  for the dispersion model (5.57). Fisher's information matrix of the saddlepoint approximation (5.60) w.r.t. the canonical parameters  $\theta$  and  $\phi$  is given by

$$\mathcal{I}(\theta, \phi) = -\mathbb{E}_{\theta, \phi} \begin{pmatrix} \phi v \kappa''(\theta) & -v(Y - \kappa'(\theta)) \\ -v(Y - \kappa'(\theta)) & -\frac{1}{2} \frac{1}{\phi^2} \end{pmatrix} = \begin{pmatrix} \frac{v}{\phi(\phi)} V(\mu(\theta)) & 0 \\ 0 & \frac{1}{2} V_{\phi}(\phi) \end{pmatrix},$$

with variance function  $V_{\phi}(\phi) = \phi^2$ , and emphasizing that we work in the canonical parametrization  $(\theta, \phi)$ . This is a positive definite diagonal matrix which suggests that the algorithm alternating the  $\boldsymbol{\beta}$  and  $\boldsymbol{\gamma}$  estimations will have a fast convergence. For fixed estimate  $\widehat{\boldsymbol{\gamma}}$  we calculate estimated dispersion parameters  $\widehat{\varphi}_i = g_{\varphi}^{-1}(\widehat{\boldsymbol{\gamma}}, \mathbf{z}_i)$  of policies  $1 \leq i \leq n$ , see (5.57). These then allow us to calculate diagonal working weight matrix

$$W(\boldsymbol{\beta}) = \text{diag} \left( \left( \frac{\partial g(\mu_i)}{\partial \mu_i} \right)^{-2} \frac{v_i}{\widehat{\varphi}_i} \frac{1}{V(\mu_i)} \right)_{1 \leq i \leq n} \in \mathbb{R}^{n \times n},$$

which is used in Fisher's scoring method/IRLS algorithm (5.12) to receive MLE  $\widehat{\boldsymbol{\beta}}$ , given the estimates  $(\widehat{\varphi}_i)_i$ . These MLEs allow us to estimate the mean parameters  $\widehat{\mu}_i = g^{-1}(\widehat{\boldsymbol{\beta}}, \mathbf{x}_i)$ , and to calculate the deviances

$$\mathfrak{d}_i = v_i \mathfrak{d}(Y_i, \widehat{\mu}_i) = 2v_i \left( Y_i h(Y_i) - \kappa(h(Y_i)) - Y_i h(\widehat{\mu}_i) + \kappa(h(\widehat{\mu}_i)) \right) \geq 0.$$

Using (5.60) we know that these deviances can be approximated by gamma distributions  $\Gamma(1/2, 1/(2\varphi_i))$ . This is a single-parameter EDF with dispersion parameter 2 (as nuisance parameter) and mean parameter  $\varphi_i$ . This motivates the definition of the working weight matrix (based on the gamma EDF model)

$$W_{\varphi}(\boldsymbol{\gamma}) = \text{diag} \left( \left( \frac{\partial g_{\varphi}(\varphi_i)}{\partial \varphi_i} \right)^{-2} \frac{1}{2} \frac{1}{V_{\varphi}(\varphi_i)} \right)_{1 \leq i \leq n} \in \mathbb{R}^{n \times n},$$

and the working residuals

$$\mathbf{R}_{\varphi}(\mathfrak{d}, \boldsymbol{\gamma}) = \left( \frac{\partial g_{\varphi}(\varphi_i)}{\partial \varphi_i} (\mathfrak{d}_i - \varphi_i) \right)_{1 \leq i \leq n}^{\top} \in \mathbb{R}^n.$$

Fisher's scoring method (5.12) iterates for  $s \geq 0$  the following recursion to receive  $\hat{\boldsymbol{\gamma}}$

$$\hat{\boldsymbol{\gamma}}^{(s)} \mapsto \hat{\boldsymbol{\gamma}}^{(s+1)} = \left( \mathfrak{Z}^\top W_\varphi(\hat{\boldsymbol{\gamma}}^{(s)}) \mathfrak{Z} \right)^{-1} \mathfrak{Z}^\top W_\varphi(\hat{\boldsymbol{\gamma}}^{(s)}) \left( \mathfrak{Z} \hat{\boldsymbol{\gamma}}^{(s)} + \mathbf{R}_\varphi(\mathfrak{d}, \hat{\boldsymbol{\gamma}}^{(s)}) \right), \quad (5.64)$$

where  $\mathfrak{Z} = (\mathbf{z}_1, \dots, \mathbf{z}_n)^\top$  is the design matrix used to estimate the dispersion parameters.

### 5.5.4 Lab: Double GLM Algorithm for Gamma Claim Sizes

We revisit the Swedish motorcycle claim size data studied in Sect. 5.3.7. We expand the gamma claim size GLM to a double GLM also modeling the systematic effects in the dispersion parameter. In a first step we need to change the parametrization of the gamma model of Sect. 5.3.7. In the former section we have modeled the average claim size  $S_i/n_i \sim \Gamma(n_i\alpha_i, n_i c_i)$ , but for applying the saddlepoint approximation we should use the reproductive form (5.44) of the gamma model. We therefore set

$$Y_i = S_i/(n_i\alpha_i) \sim \Gamma(n_i\alpha_i, n_i\alpha_i c_i). \quad (5.65)$$

The reason for the different parametrization in Sect. 5.3.7 has been that (5.65) is not directly useful if  $\alpha_i$  is unknown because in that case the observations  $Y_i$  cannot be calculated. In this section we estimate  $\varphi_i = 1/\alpha_i$  which allows us to model (5.65); a different treatment within Tweedie's family is presented in Sect. 11.1.3. The only difficulty is to initialize the double GLM algorithm. We proceed as follows.

- (0) In an initial step we assume constant dispersion  $\varphi_i = 1/\alpha_i \equiv 1/\alpha = 1$ . This gives us exactly the mean estimates of Sect. 5.3.7 for  $S_i/n_i \sim \Gamma(n_i\alpha, n_i c_i)$ ; note that for constant shape parameter  $\alpha$  the mean of  $S_i/n_i$  can be estimated without explicit knowledge of  $\alpha$  (because it cancels in the score equations). Using these mean estimates we calculate the MLE  $\hat{\alpha}^{(0)}$  of the (constant) shape parameter  $\alpha$ , see Remark 5.26. This then allows us to determine the (scaled) observations  $Y_i^{(1)} = S_i/(n_i\hat{\alpha}^{(0)})$  and we initialize  $\hat{\varphi}_i^{(0)} = 1/\hat{\alpha}^{(0)}$ .
- (1) Iterate for  $t \geq 1$ :
  - estimate the mean  $\mu_i$  of  $Y_i$  using the mean GLM (5.56) based on the observations  $Y_i^{(t)}$  and the dispersion estimates  $\hat{\varphi}_i^{(t-1)}$ . This provides us with  $\hat{\mu}_i^{(t)}$ ;
  - based on the deviances  $\mathfrak{d}_i^{(t)} = v_i \mathfrak{d}(Y_i^{(t)}, \hat{\mu}_i^{(t)})$ , calculate the updated dispersion estimates  $\hat{\varphi}_i^{(t)}$  using the dispersion GLM (5.57) and the residual MLE iteration (5.64) with the saddlepoint approximation. Set for the updated observations  $Y_i^{(t+1)} = S_i \hat{\varphi}_i^{(t)} / n_i$ .

**Table 5.15** Number of parameters, AICs, Pearson's dispersion estimate, in-sample losses, tenfold cross-validation losses and the in-sample average claim amounts of the null model (gamma intercept model) and the (double) gamma GLM

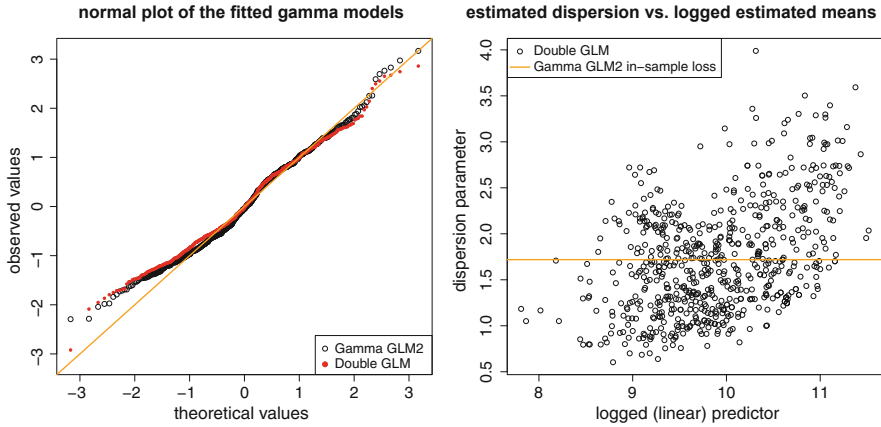
	# Param.	AIC	Dispersion est. $\hat{\varphi}^p$	In-sample loss on $\mathcal{L}$	Tenfold CV loss $\hat{\mathcal{D}}^{CV}$	Average amount
Gamma null	1 + 1	14'416	2.057	2.085	2.091	24'641
Gamma GLM2	7 + 1	14'274	1.544	1.719	1.747	25'130
Double gamma GLM	7 + 6	14'258	–	(1.721)	–	26'413

In an initial double GLM analysis we use the feature information  $\mathbf{z}_i = \mathbf{x}_i$  for the dispersion  $\varphi_i$  modeling (5.57). We choose for both GLMs the log-link which leads to concave maximization problems, see Example 5.5. Running the above double GLM algorithm converges in 4 iterations, and analyzing the resulting model we observe that we should drop the variable `RiskClass` from the feature  $\mathbf{z}_i$ . We then run the same double GLM algorithm with the feature information  $\mathbf{x}_i$  and the new  $\mathbf{z}_i$  again, and the results are presented in Table 5.15.

The considered double GLM has parameter dimensions  $\boldsymbol{\beta} \in \mathbb{R}^7$  and  $\boldsymbol{\gamma} \in \mathbb{R}^6$ . To have comparability with AIC of Sect. 5.3.7, we evaluate AIC of the double GLM in the observations  $S_i/n_i$  (and not in  $Y_i$ ; i.e., similar to the gamma GLM). We observe that it has an improved AIC value compared to model Gamma GLM2. Thus, indeed, dispersion modeling seems necessary in this example (under the GLM2 regression structure). We do not calculate in-sample and cross-validation losses in the double GLM because in the other two models of Table 5.15 we have set  $\varphi = 1$  in these statistics. However, the in-sample loss of model Gamma GLM2 with  $\varphi = 1$  corresponds to the (homogeneous) deviance dispersion estimate (up to scaling  $n/(n - (q + 1))$ ), and this in-sample loss of 1.719 can directly be compared to the average estimated dispersion  $m^{-1} \sum_{i=1}^m \hat{\varphi}_i = 1.721$  (in round brackets in Table 5.15). On the downside, the double GLM has a bigger bias which needs an adjustment.

In Fig. 5.13 (lhs) we give the normal plots of model Gamma GLM2 and the double gamma GLM model. This plot is received by transforming the observations to normal quantiles using the corresponding estimated gamma models. We see quite some similarity between the two estimated gamma models. Both models seem to have similar deficiencies, i.e., dispersion modeling improves explanation of observations, however, either the regression function or the gamma distributional assumption does not fully fit the data, especially for small claims. Finally, in Fig. 5.13 (rhs) we plot the estimated dispersion parameters  $\hat{\varphi}_i$  against the logged estimated means  $\log(\hat{\mu}_i)$  (linear predictors). We observe that the estimated dispersion has a (weak) U-shape as a function of the expected claim sizes which indicates that the tails cannot fully be captured by our model. This closes this example.

*Remark 5.29* For the dispersion estimation  $\hat{\varphi}_i$  we use as observations the deviances  $\mathfrak{d}_i = v_i \mathfrak{d}(Y_i, \hat{\mu}_i)$ ,  $1 \leq i \leq n$ . On a finite sample, these deviances are typically biased due to the use of the estimated means  $\hat{\mu}_i$ . Smyth–Verbyla [343] propose the



**Fig. 5.13** (lhs) Normal plot of the fitted models Gamma GLM2 and double GLM, (rhs) estimated dispersion parameters  $\hat{\varphi}_i$  against the logged estimated means  $\log(\hat{\mu}_i)$  (the orange line gives the in-sample loss in model Gamma GLM2)

following bias correction. Consider the estimated hat matrix defined by

$$H = W(\hat{\beta}, \hat{\gamma})^{1/2} \mathbf{x} \left( \mathbf{x}^\top W(\hat{\beta}, \hat{\gamma}) \mathbf{x} \right)^{-1} \mathbf{x}^\top W(\hat{\beta}, \hat{\gamma})^{1/2},$$

with the diagonal work weight matrix  $W(\hat{\beta}, \hat{\gamma})$  depending on the estimated regression parameters  $\hat{\beta}$  and  $\hat{\gamma}$  through  $\mu$  and  $\varphi$ . Denote the diagonal entries of the hat matrix by  $(h_{i,i})_{1 \leq i \leq n}$ . A bias corrected version of the deviances is received by considering observations  $(1 - h_{i,i})^{-1} \mathfrak{d}_i = (1 - h_{i,i})^{-1} v_i \mathfrak{d}(Y_i, \hat{\mu}_i)$ ,  $1 \leq i \leq n$ . We will come back to the hat matrix  $H$  in Sect. 5.6.1, below.

### 5.5.5 Tweedie's Compound Poisson GLM

A popular situation for applying the double GLM framework is Tweedie's CP model introduced in Sect. 2.2.3, in particular, we refer to Proposition 2.17 for the corresponding parametrization. Having claim frequency and claim sizes involved, such a model can hardly be calibrated with one single regression function and a constant dispersion parameter. An obvious choice is a double GLM, this is the proposal presented in Smyth–Jørgensen [342]. In most of the cases one chooses for both link functions  $g$  and  $g_\varphi$  the log-links because positivity needs to be guaranteed.



This implies for the two working weight matrices of the double GLM

$$W(\boldsymbol{\beta}) = \text{diag} \left( \mu_i^2 \frac{v_i}{\varphi_i} \frac{1}{V(\mu_i)} \right)_{1 \leq i \leq n} = \text{diag} \left( \mu_i^{2-p} \frac{v_i}{\varphi_i} \right)_{1 \leq i \leq n},$$

$$W_\varphi(\boldsymbol{\gamma}) = \text{diag} \left( \varphi_i^2 \frac{1}{2} \frac{1}{V_\varphi(\varphi_i)} \right)_{1 \leq i \leq n} = \text{diag}(1/2, \dots, 1/2).$$

The deviances in Tweedie's CP model are given by, see (4.18),

$$\mathfrak{d}_i = v_i \mathfrak{d}(Y_i, \hat{\mu}_i) = 2v_i \left( Y_i \frac{Y_i^{1-p} - \hat{\mu}_i^{1-p}}{1-p} - \frac{Y_i^{2-p} - \hat{\mu}_i^{2-p}}{2-p} \right) \geq 0,$$

and these deviances could still be de-biased, see Remark 5.29. The working responses for the two GLMs are

$$\mathbf{R} = (Y_i/\mu_i - 1)_{1 \leq i \leq n}^\top \quad \text{and} \quad \mathbf{R}_\varphi = (\mathfrak{d}_i/\varphi_i - 1)_{1 \leq i \leq n}^\top.$$

The drawback of this approach is that it only considers the (scaled) total claim amounts  $Y_i = S_i \varphi_i / v_i$  as observations, see Proposition 2.17. These total claim amounts consist of the number of claims  $N_i$  and i.i.d. individual claim sizes  $Z_{i,j} \sim \Gamma(\alpha, c_i)$ , supposed  $N_i \geq 1$ . Having observations of both claim amounts  $S_i$  and claim counts  $N_i$  allows one to build a Poisson GLM for claim counts and a gamma GLM for claim sizes which can be estimated separately. This has also been the reason of Smyth–Jørgensen [342] to enhance Tweedie's model estimation for known claim counts in their Section 4. Moreover, in Theorem 4 of Delong et al. [94] it is proved that the two GLM approaches can be identified under log-link choices.

## 5.6 Diagnostic Tools

In our examples we have studied several figures like AIC, cross-validation losses, etc., for model and parameter selection. Moreover, we have plotted the results, for instance, using the Tukey–Anscombe plot or the QQ plot. Of course, there are numerous other plots and tools that can help us to analyze the results and to improve the resulting models. We present some of these in this section.

### 5.6.1 The Hat Matrix

The MLE  $\hat{\boldsymbol{\beta}}^{\text{MLE}}$  satisfies at convergence of the IRLS algorithm, see (5.12),

$$\hat{\boldsymbol{\beta}}^{\text{MLE}} = \left( \mathfrak{X}^\top W(\hat{\boldsymbol{\beta}}^{\text{MLE}}) \mathfrak{X} \right)^{-1} \mathfrak{X}^\top W(\hat{\boldsymbol{\beta}}^{\text{MLE}}) \left( \mathfrak{X} \hat{\boldsymbol{\beta}}^{\text{MLE}} + \mathbf{R}(Y, \hat{\boldsymbol{\beta}}^{\text{MLE}}) \right),$$

with working residuals for  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$

$$\mathbf{R}(Y, \boldsymbol{\beta}) = \left( \frac{\partial g(\mu_i)}{\partial \mu_i} \bigg|_{\mu_i = \mu_i(\boldsymbol{\beta})} (Y_i - \mu_i(\boldsymbol{\beta})) \right)_{1 \leq i \leq n}^\top \in \mathbb{R}^n.$$

Following Section 4.2.2 of Fahrmeir–Tutz [123], this allows us to define the so-called *hat matrix*, see also Remark 5.29,

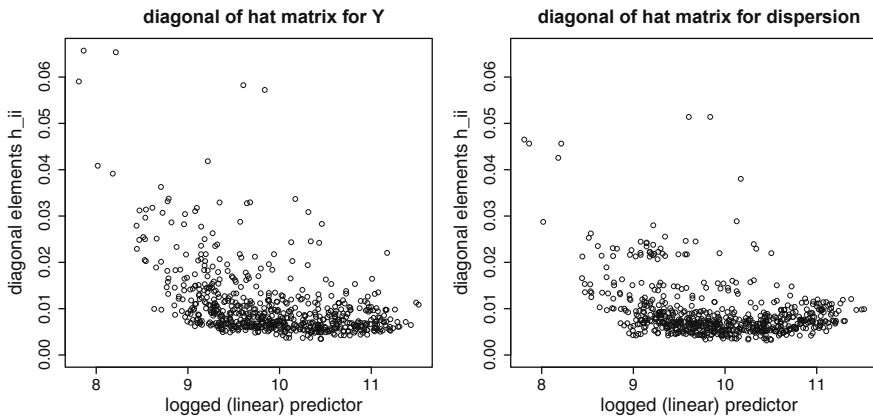
$$H = H(\hat{\boldsymbol{\beta}}^{\text{MLE}}) = W(\hat{\boldsymbol{\beta}}^{\text{MLE}})^{1/2} \mathbf{X} (\mathbf{X}^\top W(\hat{\boldsymbol{\beta}}^{\text{MLE}}) \mathbf{X})^{-1} \mathbf{X}^\top W(\hat{\boldsymbol{\beta}}^{\text{MLE}})^{1/2} \in \mathbb{R}^{n \times n}, \quad (5.66)$$

recall that the working weight matrix  $W(\boldsymbol{\beta})$  is diagonal. The hat matrix  $H$  is symmetric and idempotent, i.e.  $H^2 = H$ , with  $\text{trace}(H) = \text{rank}(H) = q + 1$ . Therefore,  $H$  acts as a projection, mapping the observations  $\tilde{Y}$  to the fitted values

$$\begin{aligned} \tilde{Y} &\stackrel{\text{def.}}{=} W(\hat{\boldsymbol{\beta}}^{\text{MLE}})^{1/2} (\mathbf{X} \hat{\boldsymbol{\beta}}^{\text{MLE}} + \mathbf{R}(Y, \hat{\boldsymbol{\beta}}^{\text{MLE}})) \mapsto H\tilde{Y} = W(\hat{\boldsymbol{\beta}}^{\text{MLE}})^{1/2} \mathbf{X} \hat{\boldsymbol{\beta}}^{\text{MLE}} \\ &= W(\hat{\boldsymbol{\beta}}^{\text{MLE}})^{1/2} \hat{\boldsymbol{\eta}}, \end{aligned}$$

the latter being the fitted linear predictors. The diagonal elements  $h_{i,i}$  of this hat matrix  $H$  satisfy  $0 \leq h_{i,i} \leq 1$ , and values close to 1 correspond to extreme data points  $i$ , in particular, for  $h_{i,i} = 1$  only observation  $\tilde{Y}_i$  influences  $\hat{\eta}_i$ , whereas for  $h_{i,i} = 0$  observation  $\tilde{Y}_i$  has no influence on  $\hat{\eta}_i$ .

Figure 5.14 gives the resulting hat matrices of the double gamma GLM of Sect. 5.5.4. On the left-hand side we show the diagonal entries  $h_{i,i}$  of the claim



**Fig. 5.14** Diagonal entries  $h_{i,i}$  of the two hat matrices of the example in Sect. 5.5.4: (lhs) for means  $\hat{\mu}_i$  and responses  $Y_i$ , and (rhs) for dispersions  $\hat{\varphi}_i$  and responses  $\mathfrak{d}_i$

amount responses  $Y_i$  (for the estimation of  $\mu_i$ ), and on the right-hand side the corresponding plots for the deviance responses  $\mathfrak{d}_i$  (for the estimation of  $\varphi_i$ ). These diagonal elements  $h_{i,i}$  are ordered on the  $x$ -axis w.r.t. the linear predictors  $\hat{\eta}_i$ . From this figure we conclude that the diagonal entries of the hat matrices are bigger for very small responses in our example, and the dispersion plot has a couple of more special observations that may require further analysis.

### 5.6.2 Case Deletion and Generalized Cross-Validation

As a continuation of the previous subsection we can analyze the influence of an individual observation  $Y_i$  on the estimation of regression parameter  $\beta$ . This influence is naturally measured by fitting the regression parameter based on the full data  $\mathcal{D}$  and based only on the observations  $\mathcal{L}_{(-i)} = \mathcal{D} \setminus \{Y_i\}$ , we also refer to leave-one-out cross-validation in Sect. 4.2.2. The influence of observation  $Y_i$  is then obtained by comparing  $\hat{\beta}^{\text{MLE}}$  and  $\hat{\beta}_{(-i)}^{\text{MLE}}$ . Since fitting  $n$  different models by individually leaving out each observation  $Y_i$  is too costly, one only explores a one-step Fisher's scoring update starting from  $\hat{\beta}^{\text{MLE}}$  that provides an approximation to  $\hat{\beta}_{(-i)}^{\text{MLE}}$ , that is,

$$\begin{aligned} \hat{\beta}_{(-i)}^{(1)} &= \left( \mathfrak{X}_{(-i)}^\top W_{(-i)} (\hat{\beta}^{\text{MLE}}) \mathfrak{X}_{(-i)} \right)^{-1} \mathfrak{X}_{(-i)}^\top W_{(-i)} (\hat{\beta}^{\text{MLE}}) \left( \mathfrak{X} \hat{\beta}^{\text{MLE}} + R(Y, \hat{\beta}^{\text{MLE}}) \right)_{(-i)} \\ &= \left( \mathfrak{X}_{(-i)}^\top W_{(-i)} (\hat{\beta}^{\text{MLE}}) \mathfrak{X}_{(-i)} \right)^{-1} \mathfrak{X}_{(-i)}^\top W_{(-i)} (\hat{\beta}^{\text{MLE}})^{1/2} \tilde{Y}_{(-i)}, \end{aligned}$$

where all lower indices  $_{(-i)}$  indicate that we drop the corresponding row or/and column from the matrices and vectors, and where  $\tilde{Y}$  has been defined in the previous subsection. This allows us to compare  $\hat{\beta}^{\text{MLE}}$  and  $\hat{\beta}_{(-i)}^{(1)}$  to analyze the influence of observation  $Y_i$ .

To reformulate this approximation, we come back to the hat matrix  $H = H(\hat{\beta}^{\text{MLE}}) = (h_{i,j})_{1 \leq i, j \leq n}$  defined in (5.66). It fulfills

$$W(\hat{\beta}^{\text{MLE}})^{1/2} \mathfrak{X} \hat{\beta}^{\text{MLE}} = H \tilde{Y} = \left( \sum_{j=1}^n h_{1,j} \tilde{Y}_j, \dots, \sum_{j=1}^n h_{n,j} \tilde{Y}_j \right)^\top \in \mathbb{R}^n.$$

Thus, for predicting  $Y_i$  we can consider the linear predictor (for the chosen link  $g$ )

$$\hat{\eta}_i = g(\hat{\mu}_i) = \langle \hat{\beta}^{\text{MLE}}, \mathbf{x}_i \rangle = (\mathfrak{X} \hat{\beta}^{\text{MLE}})_i = W_{i,i} (\hat{\beta}^{\text{MLE}})^{-1/2} \sum_{j=1}^n h_{i,j} \tilde{Y}_j.$$

A computation of the linear predictor of  $Y_i$  using the leave-one-out approximation  $\widehat{\boldsymbol{\beta}}_{(-i)}^{(1)}$  gives

$$\widehat{\eta}_i^{(-i,1)} = \langle \widehat{\boldsymbol{\beta}}_{(-i)}^{(1)}, \mathbf{x}_i \rangle = \frac{1}{1 - h_{i,i}} \widehat{\eta}_i - W_{i,i} (\widehat{\boldsymbol{\beta}}^{\text{MLE}})^{-1/2} \frac{h_{i,i}}{1 - h_{i,i}} \widetilde{Y}_i.$$

This allows one to efficiently calculate a leave-one-out prediction using the hat matrix  $H$ . This also motivates to study the *generalized cross-validation* (GCV) loss which is an approximation to leave-one-out cross-validation, see Sect. 4.2.2,

$$\begin{aligned} \widehat{\mathfrak{D}}^{\text{GCV}} &= \frac{1}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \mathfrak{d} \left( Y_i, g^{-1}(\widehat{\eta}_i^{(-i,1)}) \right) \\ &= \frac{2}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \left[ Y_i h(Y_i) - \kappa(h(Y_i)) - Y_i h \left( g^{-1}(\widehat{\eta}_i^{(-i,1)}) \right) + \kappa \left( h \left( g^{-1}(\widehat{\eta}_i^{(-i,1)}) \right) \right) \right]. \end{aligned} \quad (5.67)$$

*Example 5.30 (Generalized Cross-Validation Loss in the Gaussian Case)* We study the generalized cross-validation loss  $\widehat{\mathfrak{D}}^{\text{GCV}}$  in the homoskedastic Gaussian case  $v_i/\varphi \equiv 1/\sigma^2$  with cumulant function  $\kappa(\theta) = \theta^2/2$  and canonical link  $g(\mu) = h(\mu) = \mu$ . The generalized cross-validation loss in the Gaussian case is given by

$$\widehat{\mathfrak{D}}^{\text{GCV}} = \frac{1}{n} \sum_{i=1}^n \frac{1}{\sigma^2} \left( Y_i - \widehat{\eta}_i^{(-i,1)} \right)^2,$$

with (linear) leave-one-out predictor

$$\widehat{\eta}_i^{(-i,1)} = \langle \widehat{\boldsymbol{\beta}}_{(-i)}^{(1)}, \mathbf{x}_i \rangle = \sum_{j=1, j \neq i}^n \frac{h_{i,j}}{1 - h_{i,i}} Y_j = \frac{1}{1 - h_{i,i}} \widehat{\eta}_i - \frac{h_{i,i}}{1 - h_{i,i}} Y_i.$$

This gives us generalized cross-validation loss in the Gaussian case

$$\widehat{\mathfrak{D}}^{\text{GCV}} = \frac{1}{n} \sum_{i=1}^n \frac{1}{\sigma^2} \left( \frac{Y_i - \widehat{\eta}_i}{1 - h_{i,i}} \right)^2,$$

with  $\boldsymbol{\beta}$  independent hat matrix

$$H = \mathfrak{X} \left( \mathfrak{X}^\top \mathfrak{X} \right)^{-1} \mathfrak{X}^\top.$$

The generalized cross-validation loss is used, for instance, for generalized additive model (GAM) fitting where an efficient and fast cross-validation method is required to select regularization parameters. Generalized cross-validation has been introduced by Craven–Wahba [84] but these authors replaced  $h_{i,i}$  by  $\sum_{j=1}^n h_{j,j}/n$ . It holds that  $\sum_{j=1}^n h_{j,j} = \text{trace}(H) = q + 1$ , thus, using this approximation we receive

$$\begin{aligned}\widehat{\mathfrak{D}}^{\text{GCV}} &\approx \frac{1}{n} \sum_{i=1}^n \frac{1}{\sigma^2} \left( \frac{Y_i - \widehat{\eta}_i}{1 - \sum_{j=1}^n h_{j,j}/n} \right)^2 = \frac{n}{(n - (q + 1))^2} \sum_{i=1}^n \frac{(Y_i - \widehat{\eta}_i)^2}{\sigma^2} \\ &= \frac{n}{n - (q + 1)} \frac{\widehat{\varphi}^{\text{P}}}{\sigma^2},\end{aligned}$$

with  $\widehat{\varphi}^{\text{P}}$  being Pearson’s dispersion estimate in the Gaussian model, see (5.30). ■

We give a numerical example based on the gamma GLM for the claim sizes studied in Sect. 5.3.7.

*Example 5.31 (Leave-One-Out Cross-Validation)* The aim of this example is to compare the generalized cross-validation loss  $\widehat{\mathfrak{D}}^{\text{GCV}}$  to the leave-one-out cross-validation loss  $\widehat{\mathfrak{D}}^{\text{loo}}$ , see (4.34), the former being an approximation to the latter. We do this for the gamma claim size model studied in Sect. 5.3.7. In this example it is feasible to exactly calculate the leave-one-out cross-validation loss because we have only 656 claims.

The results are presented in Table 5.16. Firstly, the different cross-validation losses confirm that the model slightly (in-sample) over-fits to the data, which is not a surprise when estimating 7 regression parameters based on 656 observations. Secondly, the cross-validation losses provide similar numbers with leave-one-out being slightly bigger than tenfold cross-validation, here. Thirdly, the generalized cross-validation loss  $\widehat{\mathfrak{D}}^{\text{GCV}}$  manages to approximate the leave-one-out cross-validation loss  $\widehat{\mathfrak{D}}^{\text{loo}}$  very well in this example.

Table 5.17 gives the corresponding results for model Poisson GLM1 of Sect. 5.2.4. Firstly, in this example with 610’206 observations it is not feasible to calculate the leave-one-out cross-validation loss (for computational reasons). Therefore, we rely on the generalized cross-validation loss as an approximation. From the results of Table 5.17 it seems that this approximation (rather) underestimates the loss (compared to tenfold cross-validation). Indeed, this is an observation that we have made also in other examples. ■

**Table 5.16** Comparison of different cross-validation losses for model Gamma GLM2

	Gamma GLM2
In-sample loss $\mathfrak{D}(\mathcal{L}, \widehat{\mu}_{\mathcal{L}}^{\text{MLE}})$	1.719
Tenfold CV loss $\widehat{\mathfrak{D}}^{\text{CV}}$	1.747
Leave-one-out CV loss $\widehat{\mathfrak{D}}^{\text{loo}}$	1.756
Generalized CV loss $\widehat{\mathfrak{D}}^{\text{GCV}}$	1.758

**Table 5.17** Comparison of different cross-validation losses for model Poisson GLM1

	Poisson GLM1
In-sample loss $\mathfrak{D}(\mathcal{L}, \hat{\mu}_{\mathcal{L}}^{\text{MLE}})$	24.101
Tenfold CV loss $\hat{\mathfrak{D}}^{\text{CV}}$	24.121
Leave-one-out CV loss $\hat{\mathfrak{D}}^{\text{loo}}$	N/A
Generalized CV loss $\hat{\mathfrak{D}}^{\text{GCV}}$	24.105

## 5.7 Generalized Linear Models with Categorical Responses

The reader will have noticed that the discussion of GLMs in this chapter has been focusing on the single-parameter linear EDF case (5.1). In many actuarial applications we also want to study examples of the vector-valued parameter EF (2.2). We briefly discuss the categorical case since this case is frequently used.

### 5.7.1 Logistic Categorical Generalized Linear Model

We recall the EF representation of the categorical distribution studied in Sect. 2.1.4. We choose as  $\nu$  the counting measure on the finite set  $\mathcal{Y} = \{1, \dots, k+1\}$ . A random variable  $Y$  taking values in  $\mathcal{Y}$  is called categorical, and the levels  $y \in \mathcal{Y}$  can either be ordinal or nominal. This motivates dummy coding of the categorical random variable  $Y$  providing

$$T(Y) = (\mathbb{1}_{\{Y=1\}}, \dots, \mathbb{1}_{\{Y=k\}})^{\top} \in \{0, 1\}^k, \quad (5.68)$$

thus,  $k+1$  has been chosen as reference level. For the canonical parameter  $\theta = (\theta_1, \dots, \theta_k)^{\top} \in \Theta = \mathbb{R}^k$  we have cumulant function and mean functional, respectively,

$$\kappa(\theta) = \log \left( 1 + \sum_{j=1}^k e^{\theta_j} \right), \quad p = \mathbb{E}_{\theta}[T(Y)] = \nabla_{\theta} \kappa(\theta) = \frac{e^{\theta}}{1 + \sum_{j=1}^k e^{\theta_j}}.$$

With these choices we receive the EF representation of the categorical distribution (set  $\theta_{k+1} = 0$ )

$$dF(y; \theta) = \exp \left\{ \theta^{\top} T(y) - \log \left( 1 + \sum_{j=1}^k e^{\theta_j} \right) \right\} d\nu(y) = \prod_{l=1}^{k+1} \left( \frac{e^{\theta_l}}{\sum_{j=1}^{k+1} e^{\theta_j}} \right)^{\mathbb{1}_{\{y=l\}}} d\nu(y).$$

The covariance matrix of  $T(Y)$  is given by

$$\Sigma(\theta) = \text{Var}_{\theta}(T(Y)) = \nabla_{\theta}^2 \kappa(\theta) = \text{diag}(p) - pp^{\top} \in \mathbb{R}^{k \times k}.$$

Assume that we have feature information  $\mathbf{x} \in \mathcal{X} \subset \{1\} \times \mathbb{R}^q$  for response variable  $Y$ . This allows us to lift this categorical model to a GLM. The *logistic GLM* assumes for  $\mathbf{p} = (p_1, \dots, p_k)^\top \in (0, 1)^k$  a regression function,  $1 \leq l \leq k$ ,

$$\mathbf{x} \mapsto p_l = p_l(\mathbf{x}) = \mathbb{P}_{\boldsymbol{\beta}}[Y = l] = \frac{\exp\langle \boldsymbol{\beta}_l, \mathbf{x} \rangle}{1 + \sum_{j=1}^k \exp\langle \boldsymbol{\beta}_j, \mathbf{x} \rangle}, \quad (5.69)$$

for regression parameter  $\boldsymbol{\beta} = (\boldsymbol{\beta}_1^\top, \dots, \boldsymbol{\beta}_k^\top)^\top \in \mathbb{R}^{k(q+1)}$ . Equivalently, we can rewrite these regression probabilities relative to the reference level, that is, we consider linear predictors for  $1 \leq l \leq k$

$$\eta_l(\mathbf{x}) = \log \left( \frac{\mathbb{P}_{\boldsymbol{\beta}}[Y = l]}{\mathbb{P}_{\boldsymbol{\beta}}[Y = k+1]} \right) = \langle \boldsymbol{\beta}_l, \mathbf{x} \rangle. \quad (5.70)$$

Note that this naturally gives us the canonical link  $h$  which we have already derived in Sect. 2.1.4. Define the matrix for feature  $\mathbf{x} \in \mathcal{X} \subset \{1\} \times \mathbb{R}^q$

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}^\top & 0 & 0 & \dots & 0 \\ 0 & \mathbf{x}^\top & 0 & \dots & 0 \\ 0 & 0 & \mathbf{x}^\top & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \mathbf{x}^\top \end{pmatrix} \in \mathbb{R}^{k \times k(q+1)}. \quad (5.71)$$

This gives linear predictor and canonical parameter, respectively, under the canonical link  $h$

$$\boldsymbol{\theta} = h(\mathbf{p}(\mathbf{x})) = \boldsymbol{\eta}(\mathbf{x}) = \mathbf{X}\boldsymbol{\beta} = (\langle \boldsymbol{\beta}_1, \mathbf{x} \rangle, \dots, \langle \boldsymbol{\beta}_k, \mathbf{x} \rangle)^\top \in \boldsymbol{\Theta} = \mathbb{R}^k. \quad (5.72)$$

### 5.7.2 Maximum Likelihood Estimation in Categorical Models

Assume we have  $n$  independent observations  $Y_i$  following the logistic categorical GLM (5.69) with features  $\mathbf{x}_i \in \mathbb{R}^{q+1}$  and  $\mathbf{X}_i \in \mathbb{R}^{k \times k(q+1)}$ , respectively, for  $1 \leq i \leq n$ . The joint log-likelihood function is given by, we use (5.72),

$$\boldsymbol{\beta} \mapsto \ell_Y(\boldsymbol{\beta}) = \sum_{i=1}^n (\mathbf{X}_i \boldsymbol{\beta})^\top T(Y_i) - \kappa(\mathbf{X}_i \boldsymbol{\beta}).$$

This provides us with score equations

$$s(\boldsymbol{\beta}, Y) = \nabla_{\boldsymbol{\beta}} \ell_Y(\boldsymbol{\beta}) = \sum_{i=1}^n \mathbf{X}_i^\top [T(Y_i) - \nabla_{\boldsymbol{\theta}} \kappa(\mathbf{X}_i \boldsymbol{\beta})] = \sum_{i=1}^n \mathbf{X}_i^\top [T(Y_i) - \mathbf{p}(\mathbf{x}_i)] = 0,$$

with logistic regression function (5.69) for  $\mathbf{p}(\mathbf{x})$ . For the score equations with canonical link we also refer to the second case in Proposition 5.1. Next, we calculate Fisher's information matrix, we also refer to (3.16),

$$\mathcal{I}_n(\boldsymbol{\beta}) = -\mathbb{E}_{\boldsymbol{\beta}} \left[ \nabla_{\boldsymbol{\beta}}^2 \ell_Y(\boldsymbol{\beta}) \right] = \sum_{i=1}^n \mathbf{X}_i^\top \Sigma_i(\boldsymbol{\beta}) \mathbf{X}_i,$$

with covariance matrix of  $T(Y_i)$

$$\Sigma_i(\boldsymbol{\beta}) = \nabla_{\boldsymbol{\theta}}^2 \kappa(\mathbf{X}_i \boldsymbol{\beta}) = \text{diag}(\mathbf{p}(\mathbf{x}_i)) - \mathbf{p}(\mathbf{x}_i) \mathbf{p}(\mathbf{x}_i)^\top.$$

We rewrite the score in a similar way as in Sect. 5.1.4. This requires for general link  $g(\mathbf{p}) = \boldsymbol{\eta}$  and inverse link  $\mathbf{p} = g^{-1}(\boldsymbol{\eta})$ , respectively, the following block diagonal matrix

$$\begin{aligned} W(\boldsymbol{\beta}) &= \text{diag} \left( \left( \nabla_{\boldsymbol{\eta}} g^{-1}(\boldsymbol{\eta}) \Big|_{\boldsymbol{\eta}=\mathbf{X}_i \boldsymbol{\beta}} \right) \Sigma_i(\boldsymbol{\beta})^{-1} \left( \nabla_{\boldsymbol{\eta}} g^{-1}(\boldsymbol{\eta}) \Big|_{\boldsymbol{\eta}=\mathbf{X}_i \boldsymbol{\beta}} \right)^\top \right)_{1 \leq i \leq n} \\ &= \text{diag} \left( \left( \nabla_{\mathbf{p}} g(\mathbf{p}) \Big|_{\mathbf{p}=g^{-1}(\mathbf{X}_i \boldsymbol{\beta})} \right)^\top \Sigma_i(\boldsymbol{\beta}) \left( \nabla_{\mathbf{p}} g(\mathbf{p}) \Big|_{\mathbf{p}=g^{-1}(\mathbf{X}_i \boldsymbol{\beta})} \right) \right)_{1 \leq i \leq n}^{-1}, \end{aligned} \quad (5.73)$$

and the working residuals

$$\mathbf{R}(Y, \boldsymbol{\beta}) = \left( \left( \nabla_{\mathbf{p}} g(\mathbf{p}) \Big|_{\mathbf{p}=g^{-1}(\mathbf{X}_i \boldsymbol{\beta})} \right)^\top (T(Y_i) - \mathbf{p}(\mathbf{x}_i)) \right)_{1 \leq i \leq n}. \quad (5.74)$$

Because we work with the canonical link  $g = h$  and  $g^{-1} = \nabla_{\boldsymbol{\theta}} \kappa$ , we can use the simplified block diagonal matrix

$$W(\boldsymbol{\beta}) = \text{diag}(\Sigma_1(\boldsymbol{\beta}), \dots, \Sigma_n(\boldsymbol{\beta})) \in \mathbb{R}^{kn \times kn},$$

and the working residuals

$$\mathbf{R}(Y, \boldsymbol{\beta}) = \left( \Sigma_i(\boldsymbol{\beta})^{-1} (T(Y_i) - \mathbf{p}(\mathbf{x}_i)) \right)_{1 \leq i \leq n} \in \mathbb{R}^{kn}.$$

Finally, we define the design matrix

$$\mathfrak{X} = \begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \\ \vdots \\ \mathbf{X}_n \end{pmatrix} \in \mathbb{R}^{kn \times k(q+1)}.$$



Putting everything together we receive the score equations

$$s(\boldsymbol{\beta}, Y) = \nabla_{\boldsymbol{\beta}} \ell_Y(\boldsymbol{\beta}) = \mathfrak{X}^\top W(\boldsymbol{\beta}) \mathbf{R}(Y, \boldsymbol{\beta}) = 0. \quad (5.75)$$

This is now exactly in the same form as in Proposition 5.1. Fisher's scoring method/IRLS algorithm then allows us to recursively calculate the MLE of  $\boldsymbol{\beta} \in \mathbb{R}^{k(q+1)}$  by

$$\widehat{\boldsymbol{\beta}}^{(t)} \mapsto \widehat{\boldsymbol{\beta}}^{(t+1)} = \left( \mathfrak{X}^\top W(\widehat{\boldsymbol{\beta}}^{(t)}) \mathfrak{X} \right)^{-1} \mathfrak{X}^\top W(\widehat{\boldsymbol{\beta}}^{(t)}) \left( \mathfrak{X} \widehat{\boldsymbol{\beta}}^{(t)} + \mathbf{R}(Y, \widehat{\boldsymbol{\beta}}^{(t)}) \right).$$

We have asymptotic normality of the MLE (under suitable regularity conditions)

$$\widehat{\boldsymbol{\beta}}_n^{\text{MLE}} \stackrel{(d)}{\approx} \mathcal{N}(\boldsymbol{\beta}, \mathcal{I}_n(\boldsymbol{\beta})^{-1}),$$

for large sample sizes  $n$ . This allows us to apply the Wald test (5.32) for backward parameter elimination. Moreover, in-sample and out-of-sample losses can be analyzed with unit deviances coming from the categorical cross-entropy loss function (4.19).

*Remarks 5.32* The above derivations have been done for the categorical distribution under the canonical link choice. However, these considerations hold true for more general links  $g$  within the vector-valued parameter EF. That is, the block diagonal matrix  $W(\boldsymbol{\beta})$  in (5.73) and the working residuals  $\mathbf{R}(Y, \boldsymbol{\beta})$  in (5.74) provide score equations (5.75) for general vector-valued parameter EF examples, and where we replace the categorical probability  $\mathbf{p}$  by the mean  $\boldsymbol{\mu} = \mathbb{E}_{\boldsymbol{\beta}}[T(Y)]$ .

## 5.8 Further Topics of Regression Modeling

There are several special topics and tools in regression modeling that we have not discussed, yet. Some of them will be considered in selected chapters below, and some points are mentioned here, without going into detail.

### 5.8.1 Longitudinal Data and Random Effects

The GLMs studied above have been considering cross-sectional data, meaning that we have fixed one time period  $t$  and studied this time period in an isolated fashion. Time-dependent extensions are called longitudinal or panel data. Consider a time series of data  $(Y_{i,t}, \mathbf{x}_{i,t})$  for policies  $1 \leq i \leq n$  and time points  $t \geq 1$ . For the prediction of response variable  $Y_{i,t}$  we may then regress on the individual past

history of policy  $i$ , given by the data

$$\mathcal{D}_{i,t} = \{Y_{i,1}, \dots, Y_{i,t-1}, \mathbf{x}_{i,1}, \dots, \mathbf{x}_{i,t}\}.$$

In particular, we may explore the distribution of  $Y_{i,t}$ , conditionally given  $\mathcal{D}_{i,t}$ ,

$$Y_{i,t}|\mathcal{D}_{i,t} \sim F(\cdot|\mathcal{D}_{i,t}; \theta),$$

for canonical parameter  $\theta \in \Theta$  and  $F(\cdot|\mathcal{D}_{i,t}; \theta)$  being a member of the EDF. For a GLM we choose a link function  $g$  and make the assumption

$$g(\mathbb{E}_{\beta}[Y_{i,t}|\mathcal{D}_{i,t}]) = \langle \beta, \mathbf{z}_{i,t} \rangle, \quad (5.76)$$

where  $\mathbf{z}_{i,t} \in \mathbb{R}^{q+1}$  is a  $(q+1)$ -dimensional and  $\sigma(\mathcal{D}_{i,t})$ -measurable feature vector, and regression parameter  $\beta \in \mathbb{R}^{q+1}$  describes the common systematic effects across all policies  $1 \leq i \leq n$ . This gives a generalized auto-regressive model, and if we have the Markov property

$$F(\cdot|\mathcal{D}_{i,t}; \theta) \stackrel{(d)}{=} F(\cdot|Y_{i,t-1}, \mathbf{x}_{i,t}; \theta) \quad \text{for all } t \geq 2 \text{ and } \theta \in \Theta,$$

we obtain a generalized auto-regressive model of order 1. These longitudinal models allow one to model experience rating, for instance, in car insurance where the past claims history directly influences the future insurance prices, we refer to Remark 5.15 on bonus-malus systems (BMS).

The next level of complexity is obtained by extending regression structure (5.76) by policy  $i$  specific random effects  $\mathbf{B}_i$  such that we may postulate

$$g(\mathbb{E}_{\beta}[Y_{i,t}|\mathcal{D}_{i,t}, \mathbf{B}_i]) = \langle \beta, \mathbf{z}_{i,t} \rangle + \langle \mathbf{B}_i, \mathbf{w}_{i,t} \rangle, \quad (5.77)$$

with  $\sigma(\mathcal{D}_{i,t})$ -measurable feature vector  $\mathbf{w}_{i,t}$ . Regression parameter  $\beta$  then describes the fixed systematic effects that are common over the entire portfolio  $1 \leq i \leq n$  and  $\mathbf{B}_i$  describes the policy dependent random effects (assumed to be normalized  $\mathbb{E}[\mathbf{B}_i] = 0$ ). Typically one assumes that  $\mathbf{B}_1, \dots, \mathbf{B}_n$  are centered and i.i.d. Such effects are called static random effects because they are not time-dependent, and they may also be interpreted in a Bayesian sense.

Finally, extending these static random effects to dynamic random effects  $\mathbf{B}_{i,t}$ ,  $t \geq 1$ , leads to so-called state-space models, the linear state-space model being the most popular example and being fitted using the Kalman filter [207].

### 5.8.2 Regression Models Beyond the GLM Framework

There are several ways in which the GLM framework can be modified.

## Siblings of Generalized Linear Regression Functions

The most common modification of GLMs concerns the regression structure, namely, that the scalar product in the linear predictor

$$\mathbf{x} \mapsto g(\mu) = \eta = \langle \boldsymbol{\beta}, \mathbf{x} \rangle,$$

is replaced by another regression function. A popular alternative is the framework of generalized additive models (GAMs). GAMs go back to Hastie–Tibshirani [181, 182] and the standard reference is Wood [384]. GAMs consider the regression functions

$$\mathbf{x} \mapsto g(\mu) = \eta = \beta_0 + \sum_j \beta_j s_j(x_j), \quad (5.78)$$

where  $s_j : \mathbb{R} \rightarrow \mathbb{R}$  are natural cubic splines. Natural cubic splines  $s_j$  are obtained by concatenating cubic functions in so-called nodes. A GAM can have as many nodes in each cubic spline  $s_j$  as there are different levels  $x_{i,j}$  in the data  $1 \leq i \leq n$ . In general, this leads to very flexible regression models, and to control in-sample over-fitting regularization is applied, for regularization we also refer to Sect. 6.2. Regularization requires setting a tuning parameter, and an efficient determination of this tuning parameter uses generalized cross-validation, see Sect. 5.6. Nevertheless, fitting GAMs can be very computational, already for portfolios with 1 million policies and involving 20 feature components the calibration can be very slow. Moreover, regression function (5.78) does not (directly) allow for a data driven method of finding interactions between feature components. For these reasons, we do not further study GAMs in this monograph.

A modification in the regression function that is able to consider interactions between feature components is the framework of classification and regression trees (CARTs). CARTs have been introduced by Breiman et al. [54] in 1984, and they are still used in its original form today. Regression trees aim to partition the feature space  $\mathcal{X}$  into a finite number of disjoint subsets  $\mathcal{X}_t$ ,  $1 \leq t \leq T$ , such that all policies  $(Y_i, \mathbf{x}_i)$  in the same subset  $\mathbf{x}_i \in \mathcal{X}_t$  satisfy a certain homogeneity property w.r.t. the regression task (and the chosen loss function). The CART regression function is then defined by

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = \sum_{t=1}^T \hat{\mu}_t \mathbb{1}_{\{\mathbf{x} \in \mathcal{X}_t\}},$$

where  $\hat{\mu}_t$  is the homogeneous mean estimator on  $\mathcal{X}_t$ . These CARTs are popular building blocks for ensemble methods where different regression functions are combined, we mention random forests and boosting algorithms that mainly rely on CARTs. Random forests have been introduced by Breiman [52], and boosting has been popularized by Valiant [362], Kearns–Valiant [209, 210], Schapire [328],

Freund [139] and Freund–Schapire [140]. Today boosting belongs to the most powerful predictive regression methods, we mention the XGBoost algorithm of Chen–Guestrin [71] that has won many competitions. We will not further study CARTs and boosting in these notes because these methods also have some drawbacks. For instance, resulting regression functions are not continuous nor do they easily allow to extrapolate data beyond the (observed) feature space, e.g., if we have a time component. Moreover, they are more difficult in the use of unstructured data such as text data. For more on CARTs and boosting in actuarial science we refer to Denuit et al. [100] and Ferrario–Hämmerli [125].

### Other Distributional Models

The theory above has been relying on the EDF, but, of course, we could also study any other family of distribution functions. A clear drawback of the EDF is that it only considers light-tailed distribution functions, i.e., distribution functions for which the moment generating function exists around the origin. If the data is more heavy-tailed, one may need to transform this data and then use the EDF on the transformed data (with the drawback that one loses the balance property) or one chooses another family of distribution functions. Transformations have already been discussed in Remarks 2.11 and Sect. 5.3.9. Another two families of distributions that have been studied in the actuarial literature are the generalized beta of the second kind (GB2) distribution, see Venter [369], Frees et al. [137] and Chan et al. [66], and inhomogeneous phase type (IHP) distributions, see Albrecher et al. [8] and Bladt [37]. The GB2 family is a 4-parameter family, and it nests several examples such as the gamma, the Weibull, the Pareto and the Lomax distributions, see Table B1 in Chan et al. [66]. The density of the GB2 distribution is for  $y > 0$  given by

$$\begin{aligned} f(y; a, b, \alpha_1, \alpha_2) &= \frac{\frac{|a|}{b} \left(\frac{y}{b}\right)^{a\alpha_1-1}}{B(\alpha_1, \alpha_2) \left(1 + \left(\frac{y}{b}\right)^a\right)^{\alpha_1+\alpha_2}} \\ &= \frac{\frac{|a|}{y}}{B(\alpha_1, \alpha_2)} \left(\frac{\left(\frac{y}{b}\right)^a}{1 + \left(\frac{y}{b}\right)^a}\right)^{\alpha_1} \left(\frac{1}{1 + \left(\frac{y}{b}\right)^a}\right)^{\alpha_2}, \end{aligned} \quad (5.79)$$

with scale parameter  $b > 0$ , shape parameters  $a \in \mathbb{R}$  and  $\alpha_1, \alpha_2 > 0$ , and beta function

$$B(\alpha_1, \alpha_2) = \frac{\Gamma(\alpha_1)\Gamma(\alpha_2)}{\Gamma(\alpha_1 + \alpha_2)}.$$

Consider a modified logistic transformation of variable  $y \mapsto z = (y/b)^a / (1 + (y/b)^a) \in (0, 1)$ . This gives us the beta density

$$f(z; \alpha_1, \alpha_2) = \frac{z^{\alpha_1-1} (1-z)^{\alpha_2-1}}{B(\alpha_1, \alpha_2)}.$$

Thus, the GB2 distribution can be obtained by a transformation of the beta distribution. The latter provides that a GB2 distributed random variable  $Y$  can be simulated from  $Y \stackrel{(d)}{=} b(Z/(1-Z))^{1/a}$  with  $Z \sim \text{Beta}(\alpha_1, \alpha_2)$ .

A GB2 distributed random variable  $Y$  has first moment

$$\mathbb{E}_{a,b,\alpha_1,\alpha_2}[Y] = \frac{B(\alpha_1 + 1/a, \alpha_2 - 1/a)}{B(\alpha_1, \alpha_2)} b,$$

for  $-\alpha_1 a < 1 < \alpha_2 a$ . Observe that for  $a > 0$  we have that the survival function of  $Y$  is regularly varying with tail index  $\alpha_2 a > 0$ . Thus, we can model Pareto-like tails with the GB2 family; for regular variation we refer to (1.3).

As proposed in Frees et al. [137], one can introduce a regression structure for  $b > 0$  by choosing a log-link and setting

$$\log(\mathbb{E}_{a,b,\alpha_1,\alpha_2}[Y]) = \log\left(\frac{B(\alpha_1 + 1/a, \alpha_2 - 1/a)}{B(\alpha_1, \alpha_2)}\right) + \langle \boldsymbol{\beta}, \mathbf{x} \rangle.$$

MLE of  $\boldsymbol{\beta}$  may pose some challenge because it depends on nuisance parameters  $a, \alpha_1, \alpha_2$ . In a recent paper Li et al. [251], there is a proposal to extend this GB2 regression to a composite regression model; composite models are discussed in Sect. 6.4.4, below. This closes this short section, and for more examples we refer to the literature.

### 5.8.3 Quantile Regression

#### Pinball Loss Function

The GLMs introduced above aim at estimating the means  $\mu(\mathbf{x}) = \mathbb{E}_{\theta(\mathbf{x})}[Y]$  of random variables  $Y$  being explained by features  $\mathbf{x}$ . Since mean estimation can be rather sensitive in situations where we have large claims, the more robust quantile regression has attracted some attention, recently. Quantile regression has been introduced by Koenker–Bassett [220]. The idea is that instead of estimating the mean  $\mu$  of a random variable  $Y$ , we rather try to estimate its  $\tau$ -quantile for given  $\tau \in (0, 1)$ . The  $\tau$ -quantile is given by the generalized inverse  $F^{-1}(\tau)$  of the distribution function  $F$  of  $Y$ , that is,

$$F^{-1}(\tau) = \inf\{y \in \mathbb{R}; F(y) \geq \tau\}. \quad (5.80)$$

Consider the *pinball loss function* for  $y \in \mathfrak{C}$  (convex closure of the support of  $Y$ ) and actions  $a \in \mathbb{A} = \mathbb{R}$

$$(y, a) \mapsto L_\tau(y, a) = (y - a) \left( \tau - \mathbb{1}_{\{y-a < 0\}} \right) \geq 0. \quad (5.81)$$

This provides us with the expected loss for  $Y \sim F$  and action  $a \in \mathbb{A}$

$$\begin{aligned}\mathbb{E}_F [L_\tau(Y, a)] &= \mathbb{E}_F [(Y - a) (\tau - \mathbb{1}_{\{Y < a\}})] \\ &= (\tau - 1) \mathbb{E}_F [(Y - a) \mathbb{1}_{\{Y < a\}}] + \tau \mathbb{E}_F [(Y - a) \mathbb{1}_{\{Y \geq a\}}] \\ &= (\tau - 1) \int_{-\infty}^a (y - a) dF(y) + \tau \int_a^{\infty} (y - a) dF(y).\end{aligned}$$

The aim is to find an optimal action  $\hat{a}(F)$  that minimizes this expected loss, see (4.24),

$$\hat{a}(F) \in \mathfrak{A}(F) = \arg \min_{a \in \mathbb{A}} \mathbb{E}_F [L_\tau(Y, a)].$$

Note that for the time being we do not know whether the solution to this minimization problem is a singleton. For this reason, we state the solution (subject to existence) as a set-valued functional  $\mathfrak{A}$ , see (4.25).

We calculate the score equation of the expected loss using the Leibniz rule

$$\begin{aligned}\frac{\partial}{\partial a} \mathbb{E}_F [L_\tau(Y, a)] &= -(\tau - 1) \int_{-\infty}^a dF(y) - \tau \int_a^{\infty} dF(y) \\ &= -(\tau - 1)F(a) - \tau(1 - F(a)) = F(a) - \tau \stackrel{!}{=} 0.\end{aligned}$$

Assume the distribution  $F$  is continuous. This implies  $F(F^{-1}(\tau)) = \tau$ , and we have

$$F^{-1}(\tau) \in \mathfrak{A}(F) = \arg \min_{a \in \mathbb{A}} \mathbb{E}_F [L_\tau(Y, a)].$$

In fact, using the pinball loss, we have just seen that the  $\tau$ -quantile is elicitable within the class of continuous distributions, see Definition 4.18.

For a more general result we need a more general definition of a (set-valued)  $\tau$ -quantile

$$Q_\tau(F) = \left\{ y \in \mathbb{R}; \lim_{z \uparrow y} F(z) \leq \tau \leq F(y) \right\}. \quad (5.82)$$

This defines a closed interval and its lower endpoint corresponds to the generalized inverse  $F^{-1}(\tau)$  given in (5.80). In complete analogy to Theorem 4.19 on the elicibility of the mean functional, we have the following statement for the  $\tau$ -quantile; this result goes back to Thomson [351] and Sauerens [326].

**Theorem 5.33 (Gneiting [162, Theorem 9], Without Proof)** *Let  $\mathcal{F}$  be the class of distribution functions on an interval  $\mathfrak{C} \subseteq \mathbb{R}$  and choose quantile level  $\tau \in (0, 1)$ .*

- *The  $\tau$ -quantile (5.82) is elicitable relative to  $\mathcal{F}$ .*

- Assume the loss function  $L : \mathfrak{C} \times \mathbb{A} \rightarrow \mathbb{R}_+$  satisfies (L0)-(L2) on page 92 for interval  $\mathfrak{C} = \mathbb{A} \subseteq \mathbb{R}$ .  $L$  is consistent for the  $\tau$ -quantile (5.82) relative to the class  $\mathcal{F}$  of compactly supported distributions on  $\mathfrak{C}$  if and only if  $L$  is of the form

$$L(y, a) = (G(y) - G(a)) (\tau - \mathbb{1}_{\{y-a < 0\}}),$$

for a non-decreasing function  $G$  on  $\mathfrak{C}$ .

- If  $G$  is strictly increasing on  $\mathfrak{C}$  and if  $\mathbb{E}_F[G(Y)]$  exists and is finite for all  $F \in \mathcal{F}$ , then the above loss function  $L$  is strictly consistent for the  $\tau$ -quantile (5.82) relative to the class  $\mathcal{F}$ .

Theorem 5.33 characterizes the strictly consistent loss functions for quantile estimation, the pinball loss being the special case  $G(y) = y$ .

## Quantile Regression

The idea behind quantile regression is that we build a regression model for the  $\tau$ -quantile. Assume we have a datum  $(Y, \mathbf{x})$  whose conditional  $\tau$ -quantile, given  $\mathbf{x} \in \{1\} \times \mathbb{R}^q$ , can be described by the regression function

$$\mathbf{x} \mapsto g \left( F_{Y|\mathbf{x}}^{-1}(\tau) \right) = \langle \boldsymbol{\beta}_\tau, \mathbf{x} \rangle,$$

for a strictly monotone and smooth link function  $g : \mathfrak{C} \rightarrow \mathbb{R}$ , and for a regression parameter  $\boldsymbol{\beta}_\tau \in \mathbb{R}^{q+1}$ . The aim now is to estimate this regression parameter from independent data  $(Y_i, \mathbf{x}_i)$ ,  $1 \leq i \leq n$ . The pinball loss  $L_\tau$ , given in (5.81), provides us with the following optimization problem

$$\hat{\boldsymbol{\beta}}_\tau = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \sum_{i=1}^n L_\tau \left( Y_i, g^{-1} \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle \right).$$

This then allows us to estimate the corresponding  $\tau$ -quantile as a function of the feature information  $\mathbf{x}$ . For  $\tau = 1/2$  we estimate the median by

$$\hat{F}_{Y|\mathbf{x}}^{-1}(1/2) = g^{-1} \langle \hat{\boldsymbol{\beta}}_{1/2}, \mathbf{x} \rangle.$$

We conclude from this short section that we can regress any quantity  $a(F)$  that is elicitable, i.e., for which a loss function exists that is strictly consistent for  $a(F)$  on  $F \in \mathcal{F}$ . For more on quantile regression we refer to the monograph of Uribe–Guillén [361], and an interesting paper is Dimitriadis et al. [106]. We will study quantile regression within deep networks in Chap. 11.2, below.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.





# Chapter 6

## Bayesian Methods, Regularization and Expectation-Maximization



The previous chapter has been focusing on MLE of regression parameters within GLMs. Alternatively, we could address the parameter estimation problem within a Bayesian setting. The purpose of this chapter is to discuss the Bayesian estimation approach. This leads us to the notion of regularization within GLMs. Bayesian methods are also used in the Expectation-Maximization (EM) algorithm for MLE in the case of incomplete data. For literature on Bayesian theory we recommend Gelman et al. [157], Congdon [79], Robert [319], Bühlmann–Gisler [58] and Gilks et al. [158]. A nice historical (non-mathematical) review of Bayesian methods is presented in McGrayne [266]. Regularization is discussed in the book of Hastie et al. [184], and a good reference for the EM algorithm is McLachlan–Krishnan [267].

### 6.1 Bayesian Parameter Estimation

The Bayesian estimator has been introduced in Definition 3.6. Assume that the observation  $\mathbf{Y}$  has independent components  $Y_i$  that can be described by a GLM with link function  $g$  and regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ , i.e., the random variables  $Y_i$  have densities

$$Y_i \stackrel{\text{ind.}}{\sim} f(y; \boldsymbol{\beta}, \mathbf{x}_i, v_i/\varphi) = \exp \left\{ \frac{y(h \circ g^{-1})(\boldsymbol{\beta}, \mathbf{x}_i) - (\kappa \circ h \circ g^{-1})(\boldsymbol{\beta}, \mathbf{x}_i)}{\varphi/v_i} + a(y; v_i/\varphi) \right\},$$

with canonical link  $h = (\kappa')^{-1}$ . In a Bayesian approach one models the regression parameter  $\boldsymbol{\beta}$  with a prior distribution<sup>1</sup>  $\pi(\boldsymbol{\beta})$  on the parameter space  $\mathbb{R}^{q+1}$ , and the independence assumption between the components of  $\mathbf{Y}$  needs to be understood

<sup>1</sup> Often, in Bayesian arguing, distribution and density is used in an interchangeable (and not fully precise) way, and it is left to the reader to give the right meaning to  $\pi$ .

conditionally, given the regression parameter  $\beta$ . In other words, all observations  $Y_i$  share the same regression parameter  $\beta$ , which itself is modeled by a prior distribution  $\pi$ .

The joint density of  $Y$  and  $\beta$  is given by

$$p(y, \beta) = \left( \prod_{i=1}^n f(y_i; \beta, x_i, v_i/\varphi) \right) \pi(\beta) = \exp \{ \ell_{Y=y}(\beta) + \log \pi(\beta) \}. \quad (6.1)$$

For the given observation  $Y$ , this allows us to calculate the posterior density of  $\beta$  using Bayes' rule

$$\pi(\beta|Y) = \frac{p(Y, \beta)}{\int p(Y, \tilde{\beta}) d\tilde{\beta}} \propto \left( \prod_{i=1}^n f(Y_i; \beta, x_i, v_i/\varphi) \right) \pi(\beta), \quad (6.2)$$

where the proportionality sign  $\propto$  indicates that we have dropped the terms that do not depend on  $\beta$ . Thus, the functional form in  $\beta$  of the posterior density  $\pi(\beta|Y)$  is fully determined by the joint density  $p(Y, \beta)$ , and the remaining term is a normalization to obtain a proper probability distribution. In many situations, the knowledge of the functional form of the posterior density in  $\beta$  is sufficient to perform Bayesian parameter estimation, at least, numerically. We will give some references, below.

The Bayesian estimator for  $\beta$  is given by the posterior mean (supposed it exists)

$$\hat{\beta}^{\text{Bayes}} = \mathbb{E}_{\pi} [\beta|Y] = \int \beta \pi(\beta|Y) d\nu(\beta).$$

If we want to calculate the expectation of a new random variable  $Y_{n+1}$  that is conditionally, given  $\beta$ , independent of  $Y$  and follows the same GLM as  $Y$ , we can directly calculate, using the tower property and conditional independence,<sup>2</sup>

$$\begin{aligned} \mathbb{E}_{\pi} [Y_{n+1}|Y] &= \mathbb{E}_{\pi} [\mathbb{E}[Y_{n+1}|\beta, Y]|Y] = \mathbb{E}_{\pi} [\mathbb{E}[Y_{n+1}|\beta]|Y] \\ &= \mathbb{E}_{\pi} \left[ g^{-1} \langle \beta, x_{n+1} \rangle \middle| Y \right] = \int g^{-1} \langle \beta, x_{n+1} \rangle \pi(\beta|Y) d\nu(\beta), \end{aligned}$$

supposed that this first moment exists and that  $x_{n+1}$  is the feature of  $Y_{n+1}$ . We see that it all boils down to have sufficiently explicit knowledge about the posterior density  $\pi(\beta|Y)$  given in (6.2).

*Remark 6.1 (Conditional MSEPE)* Based on the assumption that the posterior distribution  $\pi(\beta|Y)$  can be determined, we can analyze the GL. In a Bayesian setup one

---

<sup>2</sup> Note that we identify probabilities  $\mathbb{P}_{\beta}[\cdot] = \mathbb{P}[\cdot|\beta]$  for given  $\beta$ .

usually does not calculate the MSE<sub>P</sub> as described in Theorem 4.1, but one rather studies the conditional MSE<sub>P</sub>, conditioned exactly on the collected information  $\mathbf{Y}$ . That is,

$$\begin{aligned}
 \mathbb{E}_\pi \left[ (Y_{n+1} - \mathbb{E}_\pi [Y_{n+1} | \mathbf{Y}])^2 \middle| \mathbf{Y} \right] &= \text{Var}_\pi (Y_{n+1} | \mathbf{Y}) \\
 &= \text{Var}_\pi (\mathbb{E} [Y_{n+1} | \boldsymbol{\beta}, \mathbf{Y}] | \mathbf{Y}) + \mathbb{E}_\pi [\text{Var} (Y_{n+1} | \boldsymbol{\beta}, \mathbf{Y}) | \mathbf{Y}] \\
 &= \text{Var}_\pi \left( g^{-1}(\boldsymbol{\beta}, \mathbf{x}_{n+1}) \middle| \mathbf{Y} \right) + \frac{\varphi}{v_{n+1}} \mathbb{E}_\pi \left[ (\kappa'' \circ h \circ g^{-1})(\boldsymbol{\beta}, \mathbf{x}_{n+1}) \middle| \mathbf{Y} \right] \\
 &= \text{Var}_\pi \left( g^{-1}(\boldsymbol{\beta}, \mathbf{x}_{n+1}) \middle| \mathbf{Y} \right) + \frac{\varphi}{v_{n+1}} \mathbb{E}_\pi \left[ V(g^{-1}(\boldsymbol{\beta}, \mathbf{x}_{n+1})) \middle| \mathbf{Y} \right],
 \end{aligned}$$

where we need to assume existence of second moments. Similar to Theorem 4.1, the first term is the estimation variance (in a Bayesian setting) and the second term is the average process variance (using the EDF variance function  $\mu \mapsto V(\mu)$ ).

The remaining difficulty is the calculation of the posterior expectation of functions of  $\boldsymbol{\beta}$ , based on posterior density (6.2). In very well-designed experiments the posterior density  $\pi(\boldsymbol{\beta} | \mathbf{Y})$  can be determined explicitly, for instance, in the homogeneous EDF case with so-called conjugate priors, see Chapter 2 in Bühlmann–Gisler [58]. But in most cases, there is no closed form solution for the posterior distribution. Major progress in Bayesian modeling has been made with the emergence of computational methods like the Markov chain Monte Carlo (MCMC) method, Gibbs sampling, the Metropolis–Hastings (MH) algorithm [185, 274], sequential Monte Carlo (SMC) sampling, non-linear particle filters, and the Hamilton Monte Carlo (HMC) algorithm. These methods help us to empirically approximate the posterior density  $\pi(\boldsymbol{\beta} | \mathbf{Y})$  in different modeling setups. These methods have in common that the explicit knowledge of the normalizing constant in (6.2) is not necessary, but it suffices to know the functional form in  $\boldsymbol{\beta}$  of the posterior density  $\pi(\boldsymbol{\beta} | \mathbf{Y})$ .

For a detailed description of MCMC methods in general, which includes Gibbs sampling and MH algorithms, we refer to Gilks et al. [158], Green [169, 170], Johansen et al. [199]; SMC sampling and non-linear particle filters are explained in Del Moral et al. [92, 93], Johansen–Evers [199], Doucet–Johansen [111], Creal [85] and Wüthrich [389]; the HMC algorithm is described in Neal [281]. We do not present these algorithms here, but for the description of the most popular algorithms we refer to Section 4.4 in Wüthrich–Buser [392]. The reason for not presenting these algorithms here is that they still face the curse of dimensionality, which makes it difficult to use Bayesian methods for high-dimensional data sets in large models; we provide another short discussion in Sect. 11.6.3, below.

## 6.2 Regularization

### 6.2.1 Maximal a Posterior Estimator

In the previous section we have proposed to approximate the posterior density  $\pi(\boldsymbol{\beta}|Y)$  of the regression parameter  $\boldsymbol{\beta}$ , given  $Y$ , using MCMC methods. The posterior log-likelihood in the Bayesian GLM is given by, see (6.2),

$$\begin{aligned} \log \pi(\boldsymbol{\beta}|Y) &\propto \ell_Y(\boldsymbol{\beta}) + \log \pi(\boldsymbol{\beta}) \\ &\propto \sum_{i=1}^n \frac{Y_i(h \circ g^{-1})(\boldsymbol{\beta}, \mathbf{x}_i) - (\kappa \circ h \circ g^{-1})(\boldsymbol{\beta}, \mathbf{x}_i)}{\varphi/v_i} + \log \pi(\boldsymbol{\beta}). \end{aligned}$$

Compared to the classical log-likelihood function  $\ell_Y(\boldsymbol{\beta})$  for MLE, there is an additional log-density term  $\log \pi(\boldsymbol{\beta})$  that comes from the prior distribution of  $\boldsymbol{\beta}$ . Thus, the posterior log-likelihood is a balanced version of the log-likelihood  $\ell_Y(\boldsymbol{\beta})$  of the data  $Y$  and the prior log-density  $\log \pi(\boldsymbol{\beta})$  of the regression parameter  $\boldsymbol{\beta}$ . We interpret this as *regularization* because the prior  $\pi$  smooths extremes in the log-likelihood of the observation  $Y$ . This gives rise to estimate the regression parameter  $\boldsymbol{\beta}$  by the so-called maximal a posterior (MAP) estimator

$$\hat{\boldsymbol{\beta}}^{\text{MAP}} = \arg \max_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \log \pi(\boldsymbol{\beta}|Y) = \arg \max_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \ell_Y(\boldsymbol{\beta}) + \log \pi(\boldsymbol{\beta}). \quad (6.3)$$

This  $\pi$ -regularized (MAP) parameter estimation has gained much popularity because it is a useful tool to prevent the model from over-fitting under suitable prior choices. Moreover, under specific choices, it allows for parameter selection. This is especially useful in high-dimensional problems; for a reference we refer to Hastie et al. [184].

Popular choices for  $\pi$  are prior densities coming from  $L^p$ -norms for some  $p \geq 1$ , that is,  $\pi(\boldsymbol{\beta}) \propto \exp\{-\lambda \|\boldsymbol{\beta}\|_p^p\}$  for  $\lambda > 0$ . Optimization problem (6.3) then becomes

$$\hat{\boldsymbol{\beta}}^{\text{MAP}} = \arg \max_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \ell_Y(\boldsymbol{\beta}) - \lambda \|\boldsymbol{\beta}\|_p^p,$$

for a fixed *regularization parameter*  $\lambda > 0$  (also called tuning parameter). In practical applications we should exclude the intercept parameter  $\beta_0 \in \mathbb{R}$  from regularization: if we work with the canonical link within the GLM framework we have the balance property which implies unbiasedness, see Corollary 5.7. This property gets lost if  $\beta_0$  is included in the regularization term. For this reason, we set  $\boldsymbol{\beta}_- = (\beta_1, \dots, \beta_q)^\top \in \mathbb{R}^q$  and we let regularization only act on these components

$$\hat{\boldsymbol{\beta}}^{\text{MAP}} = \hat{\boldsymbol{\beta}}^{\text{MAP}}(\lambda) = \arg \max_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \frac{1}{n} \ell_Y(\boldsymbol{\beta}) - \lambda \|\boldsymbol{\beta}_-\|_p^p, \quad (6.4)$$

we also scale with the sample size  $n$  to make the units of the tuning parameter  $\lambda$  independent of the sample size  $n$ .

*Remarks 6.2*

- The regularization term  $\lambda \|\boldsymbol{\beta}_-\|_p^p$  keeps the components of the regression parameter  $\boldsymbol{\beta}_-$  close to zero, thus, it prevents from over-fitting by letting parameters only take moderate values. The magnitudes of the parameter values are controlled by the regularization parameter  $\lambda > 0$  which acts as a hyper-parameter. Optimal hyper-parameters are determined by cross-validation.
- In (6.4) all components of  $\boldsymbol{\beta}_-$  are treated equally. This may not be appropriate if the feature components of  $\mathbf{x}$  live on different scales. This problem of different scales can be solved by either scaling the components of  $\mathbf{x}$  to a unit scale, or by introducing a diagonal importance matrix  $T = \text{diag}(t_1, \dots, t_q)$  with  $t_j > 0$  that describes the scales of the components of  $\mathbf{x}$ . This allows us to regularize  $\|T^{-1}\boldsymbol{\beta}_-\|_p^p$  instead of  $\|\boldsymbol{\beta}_-\|_p^p$ . Thus, in this latter case we replace (6.4) by the weighted version

$$\hat{\boldsymbol{\beta}}^{\text{MAP}} = \arg \max_{\boldsymbol{\beta}} \frac{1}{n} \ell_Y(\boldsymbol{\beta}) - \lambda \sum_{j=1}^q t_j^{-p} |\beta_j|^p.$$

- Often, the features have a natural group structure  $\mathbf{x} = (x_0, \mathbf{x}_1, \dots, \mathbf{x}_K)$ , for instance,  $\mathbf{x}_k \in \{0, 1\}^{q_k}$  may represent dummy coding of a categorical feature component with  $q_k + 1$  levels. In that case regularization should equally act on all components of  $\boldsymbol{\beta}_k \in \mathbb{R}^{q_k}$  (that correspond to  $\mathbf{x}_k$ ) because these components describe the same systematic effect. Yuan–Lin [398] proposed for this problem grouped penalties of the form

$$\hat{\boldsymbol{\beta}}^{\text{MAP}} = \arg \max_{\boldsymbol{\beta}} \frac{1}{n} \ell_Y(\boldsymbol{\beta}) - \lambda \sum_{k=1}^K \|\boldsymbol{\beta}_k\|_2. \quad (6.5)$$

This proposal leads to sparsity, i.e., for large regularization parameters  $\lambda$  the entire  $\boldsymbol{\beta}_k$  may be shrunk (exactly) to zero; this is discussed in Sect. 6.2.5, below. We also refer to Section 4.3 in Hastie et al. [184], and Devriendt et al. [104] proposed this approach in the actuarial literature.

- There are more versions of regularization, e.g., in the fused LASSO approach we ensure that the first differences  $\beta_j - \beta_{j-1}$  remain small.

Our motivation for considering regularization has been inspired by Bayesian theory, but we can also come from a completely different angle, namely, we can consider a constraint optimization problem with a given budget constraint  $c > 0$ . That is, we can consider

$$\arg \max_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \frac{1}{n} \ell_Y(\boldsymbol{\beta}) \quad \text{subject to } \|\boldsymbol{\beta}_-\|_p^p \leq c. \quad (6.6)$$

This optimization problem can be tackled by the method of Karush, Kuhn and Tucker (KKT) [208, 228]. Optimization problem (6.4) corresponds by Lagrangian duality to the constraint optimization problem (6.6). For every  $c$  for which the budget constraint in (6.6) is binding  $\|\boldsymbol{\beta}_-\|_p^p = c$ , there is a corresponding regularization parameter  $\lambda = \lambda(c)$ , and, conversely, the solution of (6.4) solves (6.6) with  $c = \|\hat{\boldsymbol{\beta}}_-^{\text{MAP}}(\lambda)\|_p^p$ .

### 6.2.2 Ridge vs. LASSO Regularization

We compare the two special cases of  $p = 1, 2$  in this section, and in the subsequent Sects. 6.2.3 and 6.2.4 we discuss how these two cases can be solved numerically.

**Ridge Regularization**  $p = 2$  For  $p = 2$ , the prior distribution  $\pi$  in (6.4) is a centered Gaussian distribution. This  $L^2$ -regularization is called *ridge regularization* or Tikhonov regularization [353], and we have

$$\hat{\boldsymbol{\beta}}^{\text{ridge}} = \hat{\boldsymbol{\beta}}^{\text{ridge}}(\lambda) = \arg \max_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \frac{1}{n} \ell_Y(\boldsymbol{\beta}) - \lambda \sum_{j=1}^q \beta_j^2. \quad (6.7)$$

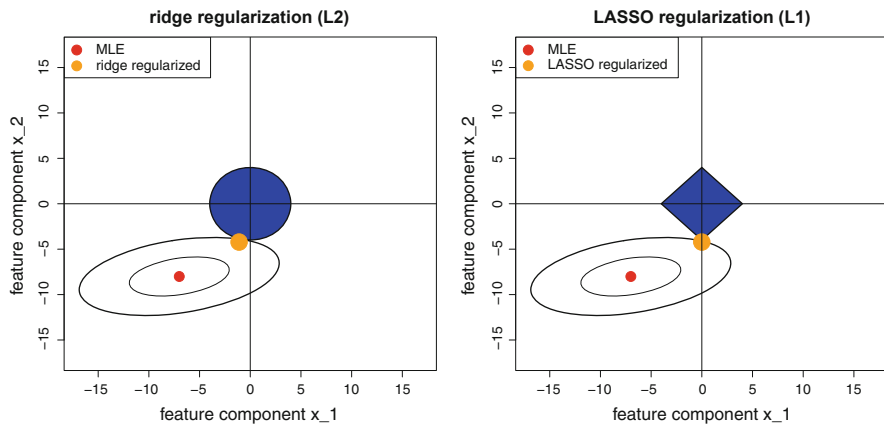
**LASSO Regularization**  $p = 1$  For  $p = 1$ , the prior distribution  $\pi$  in (6.4) is a Laplace distribution. This  $L^1$ -regularization is called *LASSO regularization* (least absolute shrinkage and selection operator), see Tibshirani [352], and we have

$$\hat{\boldsymbol{\beta}}^{\text{LASSO}} = \hat{\boldsymbol{\beta}}^{\text{LASSO}}(\lambda) = \arg \max_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \frac{1}{n} \ell_Y(\boldsymbol{\beta}) - \lambda \sum_{j=1}^q |\beta_j|. \quad (6.8)$$

LASSO regularization has the advantage that it shrinks (unimportant) regression components to exactly zero, i.e., LASSO regularization can be used for parameter elimination and model reduction. This is discussed in the next paragraphs.

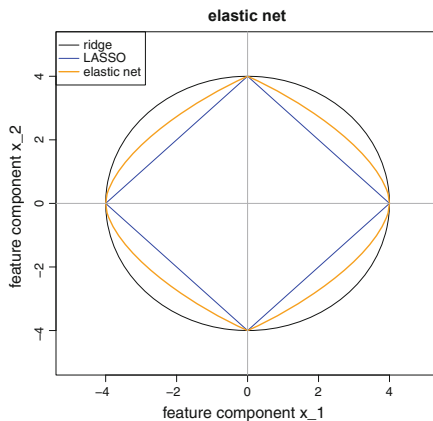
**Ridge vs. LASSO Regularization** Ridge ( $p = 2$ ) and LASSO ( $p = 1$ ) regularization behave rather differently. This can be understood best by using the budget constraint (6.6) interpretation which gives us a nice geometric illustration. The crucial part is that the side constraint gives us either a budget constraint  $\|\beta_-\|_2^2 = \sum_{j=1}^q \beta_j^2 \leq c$  (squared Euclidean norm) or  $\|\beta_-\|_1 = \sum_{j=1}^q |\beta_j| \leq c$  (Manhattan norm). In Fig. 6.1 we illustrate these two cases, the left-hand side shows the Euclidean ball in blue color (in two dimensions) and the right-hand side shows the corresponding Manhattan square in blue color; this figure is similar to Figure 2.2 in Hastie et al. [184].

The (unconstraint) MLE  $\hat{\beta}^{\text{MLE}}$  is illustrated by the red dot in Fig. 6.1. If the red dot would lie within the blue area, the budget constraint would not be binding. In Fig. 6.1 the red dot (MLE) does not lie within the blue budget constraint, and we need to compromise in the optimality of the MLE. Assume that the log-likelihood  $\beta \mapsto \ell_Y(\beta)$  is a concave function in  $\beta$ , then we receive convex level sets  $\{\beta; \ell_Y(\beta) \geq \gamma_0\}$  around the MLE  $\hat{\beta}^{\text{MLE}}$ . The critical constant  $\gamma_0$  for which this level set is tangential to the blue budget constraint exactly gives us the solution to (6.6); this solution corresponds to the yellow dots in Fig. 6.1. The crucial difference between ridge and LASSO regularization is that in the latter case the yellow dot will eventually be in the corner of the Manhattan square if we shrink the budget constraint  $c$  to zero. Or in other words, some of the components of  $\beta$  are set exactly equal to zero for small  $c$  or large  $\lambda$ , respectively; in Fig. 6.1 (rhs) this happens to the first component of  $\hat{\beta}^{\text{LASSO}}$  (under the given budget constraint  $c$ ). In



**Fig. 6.1** Illustration of optimization problem (6.6) under a budget constraint (lhs) for  $p = 2$  (Euclidean norm) and (rhs)  $p = 1$  (Manhattan norm)

**Fig. 6.2** Elastic net regularization



ridge regularization this is not the case, except for special situations concerning the position of the red MLE. Thus, ridge regression makes components of parameter estimates generally smaller, whereas LASSO shrinks some of these components exactly to zero (this also explains the name LASSO).

*Remark 6.3 (Elastic Net)* LASSO regularization faces difficulties with collinearity in feature components. In particular, if we have a group of highly correlated feature components, LASSO fails to do a grouped selection, but it selects one component and ignores the other ones. On the other hand, ridge regularization can deal with this issue. For this reason, Zou–Hastie [409] proposed the *elastic net regularization*, which uses a combined regularization term

$$\hat{\boldsymbol{\beta}}^{\text{elastic net}} = \arg \max_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \frac{1}{n} \ell_Y(\boldsymbol{\beta}) - \lambda \left[ (1 - \alpha) \|\boldsymbol{\beta}\|_2^2 + \alpha \|\boldsymbol{\beta}\|_1 \right],$$

for some  $\alpha \in (0, 1)$ . The  $L^1$ -term gives sparsity and the quadratic term removes the limitation on the number of selected variables, providing a grouped selection. In Fig. 6.2 we compare the elastic net regularization (orange color) to ridge and LASSO regularization (black and blue color). Ridge regularization provides a smooth strictly convex boundary (black), whereas LASSO provides a boundary that is non-differentiable in the corners (blue). The elastic net is still non-differentiable in the corners, this is needed for variable selection, and at the same time it is strictly convex between the corners which is needed for grouping.



### 6.2.3 Ridge Regression

In this section we consider ridge regression ( $p = 2$ ) in more detail and we provide an example. The ridge estimator  $\hat{\beta}^{\text{ridge}}$  in (6.7) is found by solving the score equations

$$\tilde{s}(\beta, Y) = \nabla_{\beta} \left( \ell_Y(\beta) - n\lambda \|\beta_{-}\|_2^2 \right) = \mathfrak{X}^{\top} W(\beta) \mathbf{R}(Y, \beta) - 2n\lambda \beta_{-} = 0, \quad (6.9)$$

note that we exclude the intercept  $\beta_0$  from regularization (we use a slight abuse of notation, here), and we also refer to Proposition 5.1. The negative expected Hessian of this optimization problem is given by

$$\mathcal{J}(\beta) = -\mathbb{E}_{\beta} \left[ \nabla_{\beta}^2 \left( \ell_Y(\beta) - n\lambda \|\beta_{-}\|_2^2 \right) \right] = \mathcal{I}(\beta) + 2n\lambda \text{diag}(0, 1, \dots, 1) \in \mathbb{R}^{(q+1) \times (q+1)},$$

where  $\mathcal{I}(\beta) = \mathfrak{X}^{\top} W(\beta) \mathfrak{X}$  is Fisher's information matrix of the unconstrained MLE problem. This provides us with Fisher's scoring updates for  $t \geq 0$ , see (5.13),

$$\hat{\beta}^{(t)} \mapsto \hat{\beta}^{(t+1)} = \hat{\beta}^{(t)} + \mathcal{J}(\hat{\beta}^{(t)})^{-1} \tilde{s}(\hat{\beta}^{(t)}, Y). \quad (6.10)$$

**Lemma 6.4** *Fisher's scoring update (6.10) can be rewritten as follows*

$$\hat{\beta}^{(t)} \mapsto \hat{\beta}^{(t+1)} = \mathcal{J}(\hat{\beta}^{(t)})^{-1} \mathfrak{X}^{\top} W(\hat{\beta}^{(t)}) \left( \mathfrak{X} \hat{\beta}^{(t)} + \mathbf{R}(Y, \hat{\beta}^{(t)}) \right).$$

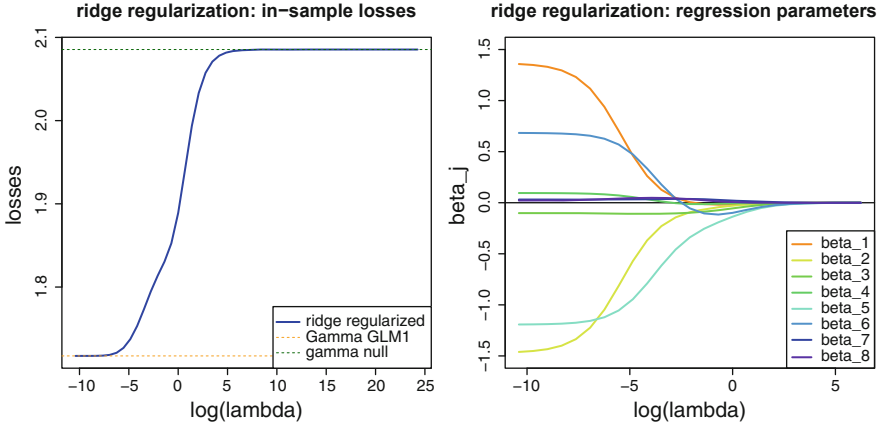
**Proof** A straightforward calculation shows

$$\begin{aligned} \hat{\beta}^{(t+1)} &= \hat{\beta}^{(t)} + \mathcal{J}(\hat{\beta}^{(t)})^{-1} \tilde{s}(\hat{\beta}^{(t)}, Y) \\ &= \mathcal{J}(\hat{\beta}^{(t)})^{-1} \left( \mathcal{J}(\hat{\beta}^{(t)}) \hat{\beta}^{(t)} + \mathfrak{X}^{\top} W(\hat{\beta}^{(t)}) \mathbf{R}(Y, \hat{\beta}^{(t)}) - 2n\lambda \hat{\beta}_{-}^{(t)} \right) \\ &= \mathcal{J}(\hat{\beta}^{(t)})^{-1} \left( \mathcal{I}(\hat{\beta}^{(t)}) \hat{\beta}^{(t)} + \mathfrak{X}^{\top} W(\hat{\beta}^{(t)}) \mathbf{R}(Y, \hat{\beta}^{(t)}) \right) \\ &= \mathcal{J}(\hat{\beta}^{(t)})^{-1} \mathfrak{X}^{\top} W(\hat{\beta}^{(t)}) \left( \mathfrak{X} \hat{\beta}^{(t)} + \mathbf{R}(Y, \hat{\beta}^{(t)}) \right). \end{aligned}$$

This proves the claim.  $\square$

Lemma 6.4 allows us to fit a ridge regularized GLM. To determine an optimal regularization parameter  $\lambda \geq 0$  one uses cross-validation, in particular, generalized cross-validation is used to receive an efficient cross-validation method, see (5.67).

**Example 6.5 (Ridge Regression)** We revisit the gamma claim size example of Sect. 5.3.7, and we choose model Gamma GLM1, see Listing 5.11. This example does not consider any categorical features, but only continuous ones. We directly



**Fig. 6.3** Ridge regularized MLEs in model Gamma GLM1: (lhs) in-sample deviance losses as a function of the regularization parameter  $\lambda > 0$ , (rhs) resulting  $\hat{\beta}_j^{\text{ridge}}(\lambda)$  for  $1 \leq j \leq q = 8$

apply Fisher's scoring updates (6.10).<sup>3</sup> For this analysis we center and normalize (to unit variance) the columns of the design matrix (except for the initial column of  $\mathfrak{X}$  encoding the intercept).

Figure 6.3 (lhs) shows the resulting in-sample deviance losses as a function of  $\lambda > 0$ . Regularization parameter  $\lambda$  allows us to continuously connect the in-sample deviance losses of the null model (2.085) and model Gamma GLM1 (1.717), see Table 5.13. Figure 6.3 (rhs) shows the regression parameter estimates  $\hat{\beta}_j^{\text{ridge}}(\lambda)$ ,  $1 \leq j \leq q = 8$ , as a function of  $\lambda > 0$ . Overall they decrease because the budget constraint gets more tight for increasing  $\lambda$ , however, the individual parameters do not need to be monotone, since one parameter may (better) compensate a decrease of another (through correlations in feature components).

Finally, we need to choose the optimal regularization parameter  $\lambda > 0$ . This is done by cross-validation. We exploit the generalized cross-validation loss, see (5.67), and the hat matrix in this ridge regularized case is given by

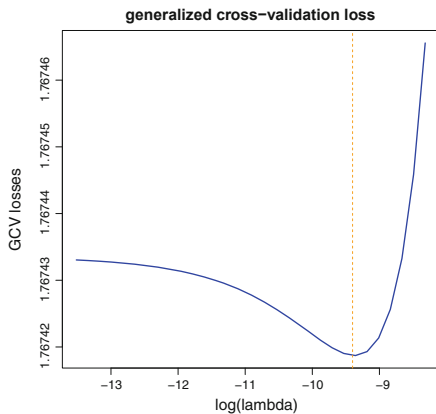
$$H_\lambda = W(\hat{\beta}^{\text{ridge}})^{1/2} \mathfrak{X} \mathcal{J}(\hat{\beta}^{\text{ridge}})^{-1} \mathfrak{X}^\top W(\hat{\beta}^{\text{ridge}})^{1/2}.$$

In contrast to (5.66), this hat matrix  $H_\lambda$  is not a projection but we would need to work in an augmented model to receive the projection property (accounting for the regularization part).

Figure 6.4 plots the generalized cross-validation loss as a function of  $\lambda > 0$ . We observe the minimum in parameter  $\lambda = e^{-9.4}$ . The resulting generalized cross-validation loss is 1.76742. This is bigger than the one received in model Gamma

<sup>3</sup> The R command `glmnet` [142] allows for regularized MLE, however, the current version does not include the gamma distribution. Therefore, we have implemented our own routine.

**Fig. 6.4** Generalized cross-validation loss  $\widehat{\mathcal{D}}^{\text{GCV}}(\lambda)$  as a function of  $\lambda > 0$



GLM2, see Table 5.16, thus, we still prefer model Gamma GLM2 over the optimally ridge regularized model GLM1. Note that for model Gamma GLM2 we did variable selection, whereas ridge regression just generally shrinks regression parameters. For more interpretation we refer to Example 6.8, below, which considers LASSO regularization. ■

### 6.2.4 LASSO Regularization

In this section we consider LASSO regularization ( $p = 1$ ). This is more challenging than ridge regularization because of the non-differentiability of the budget constraint, see Fig. 6.1 (rhs). This section follows Chapters 2 and 5 of Hastie et al. [184] and Parikh–Boyd [292].

#### Gaussian Case

We start with the homoskedastic Gaussian model having unit variance  $\sigma^2 = 1$ . In a first step, the regression model only involves one feature component  $q = 1$ . Thus, we aim at solving LASSO optimization

$$\widehat{\beta}^{\text{LASSO}} = \arg \max_{\beta \in \mathbb{R}^2} -\frac{1}{2n} \sum_{i=1}^n (Y_i - \beta_0 - \beta_1 x_i)^2 - \lambda |\beta_1|.$$

We standardize the observations and features  $(Y_i, x_i)_{1 \leq i \leq n}$  such that we have  $\sum_{i=1}^n Y_i = 0$ ,  $\sum_{i=1}^n x_i = 0$  and  $n^{-1} \sum_{i=1}^n x_i^2 = 1$ . This implies that we can omit the intercept parameter  $\beta_0$ , as the optimal intercept satisfies for this standardized data (and any  $\beta_1 \in \mathbb{R}$ )

$$\widehat{\beta}_0 = \frac{1}{n} \sum_{i=1}^n Y_i - \beta_1 x_i = 0. \quad (6.11)$$

Thus, w.l.o.g., we assume to work with standardized data in this section, this gives us the optimization problem (we drop the lower index in  $\beta_1$  because we only have one component)

$$\hat{\beta}^{\text{LASSO}} = \hat{\beta}^{\text{LASSO}}(\lambda) = \arg \max_{\beta \in \mathbb{R}} -\frac{1}{2n} \sum_{i=1}^n (Y_i - \beta x_i)^2 - \lambda |\beta|. \quad (6.12)$$

The difficulty is that the regularization term is not differentiable in zero. Since this term is convex we can express its derivative in terms of a sub-gradient  $\mathfrak{s}$ . This provides score

$$\frac{\partial}{\partial \beta} \left( -\frac{1}{2n} \sum_{i=1}^n (Y_i - \beta x_i)^2 - \lambda |\beta| \right) = \frac{1}{n} \sum_{i=1}^n (Y_i - \beta x_i) x_i - \lambda \mathfrak{s} = \frac{1}{n} \langle \mathbf{Y}, \mathbf{x} \rangle - \beta - \lambda \mathfrak{s},$$

where we use standardization  $n^{-1} \sum_{i=1}^n x_i^2 = 1$  in the second step,  $\langle \mathbf{Y}, \mathbf{x} \rangle$  is the scalar product of  $\mathbf{Y}, \mathbf{x} = (x_1, \dots, x_n)^\top \in \mathbb{R}^n$ , and where we consider the sub-gradient

$$\mathfrak{s} = \mathfrak{s}(\beta) = \begin{cases} +1 & \text{if } \beta > 0, \\ -1 & \text{if } \beta < 0, \\ \in [-1, 1] & \text{otherwise.} \end{cases}$$

Henceforth, we receive the score equation for  $\beta \neq 0$

$$n^{-1} \langle \mathbf{Y}, \mathbf{x} \rangle - \beta - \lambda \mathfrak{s} = n^{-1} \langle \mathbf{Y}, \mathbf{x} \rangle - \beta - \text{sign}(\beta) \lambda \stackrel{!}{=} 0.$$

This score equation has a proper solution  $\hat{\beta} > 0$  if  $n^{-1} \langle \mathbf{Y}, \mathbf{x} \rangle > \lambda$ , and it has a proper solution  $\hat{\beta} < 0$  if  $n^{-1} \langle \mathbf{Y}, \mathbf{x} \rangle < -\lambda$ . In any other case we have a boundary solution  $\hat{\beta} = 0$  for our maximization problem (6.12).

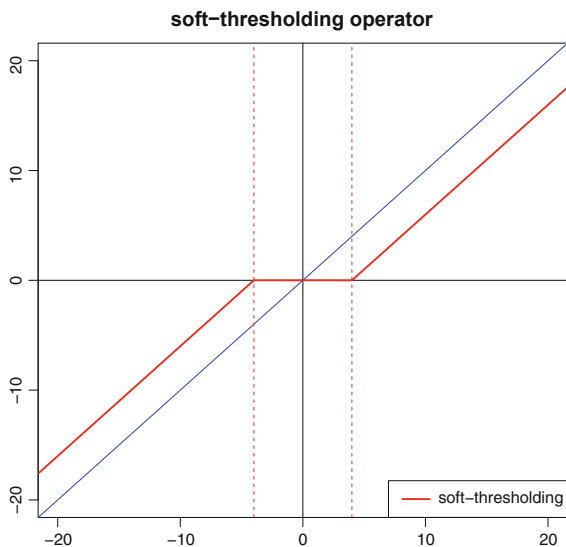
This solution can be written in terms of the following *soft-thresholding operator* for  $\lambda \geq 0$

$$\hat{\beta}^{\text{LASSO}} = \mathcal{S}_\lambda \left( n^{-1} \langle \mathbf{Y}, \mathbf{x} \rangle \right) \quad \text{with} \quad \mathcal{S}_\lambda(x) = \text{sign}(x)(|x| - \lambda)_+. \quad (6.13)$$

This soft-thresholding operator is illustrated in Fig. 6.5 for  $\lambda = 4$ .

This approach can be generalized to multiple feature components  $\mathbf{x} \in \mathbb{R}^q$ . We standardize the observations and features  $\sum_{i=1}^n Y_i = 0$ ,  $\sum_{i=1}^n x_{i,j} = 0$  and

**Fig. 6.5** Soft-thresholding operator  $x \mapsto \mathcal{S}_\lambda(x)$  for  $\lambda = 4$  (red dotted lines)



$n^{-1} \sum_{i=1}^n x_{i,j}^2 = 1$  for all  $1 \leq j \leq q$ . This allows us again to drop the intercept term and to directly consider

$$\hat{\boldsymbol{\beta}}^{\text{LASSO}} = \hat{\boldsymbol{\beta}}^{\text{LASSO}}(\lambda) = \arg \max_{\boldsymbol{\beta} \in \mathbb{R}^q} -\frac{1}{2n} \sum_{i=1}^n \left( Y_i - \sum_{j=1}^q \beta_j x_{i,j} \right)^2 - \lambda \|\boldsymbol{\beta}\|_1.$$

Since this is a concave (quadratic) maximization problem with a separable (convex) penalty term, we can apply a *cycle coordinate descent method* that iterates a cyclic coordinate-wise maximization until convergence. Thus, if we want to maximize in the  $t$ -th iteration the  $j$ -th coordinate of the regression parameter we consider recursively

$$\hat{\beta}_j^{(t)} = \arg \max_{\beta_j \in \mathbb{R}} -\frac{1}{2n} \sum_{i=1}^n \left( Y_i - \sum_{l=1}^{j-1} \beta_l^{(t)} x_{i,l} - \sum_{l=j+1}^q \beta_l^{(t-1)} x_{i,l} - \beta_j x_{i,j} \right)^2 - \lambda |\beta_j|.$$

Using the soft-thresholding operator (6.13) we find the optimal solution

$$\hat{\beta}_j^{(t)} = \mathcal{S}_\lambda \left( n^{-1} \left\langle \mathbf{Y} - \sum_{l=1}^{j-1} \beta_l^{(t)} \mathbf{x}_l - \sum_{l=j+1}^q \beta_l^{(t-1)} \mathbf{x}_l, \mathbf{x}_j \right\rangle \right),$$

with vectors  $\mathbf{x}_l = (x_{1,l}, \dots, x_{n,l})^\top \in \mathbb{R}^n$  for  $1 \leq l \leq q$ . Iteration until convergence provides the LASSO regularized estimator  $\hat{\boldsymbol{\beta}}^{\text{LASSO}}(\lambda)$  for given regularization parameter  $\lambda > 0$ .

Typically, we want to explore  $\hat{\beta}^{\text{LASSO}}(\lambda)$  for multiple  $\lambda$ 's. For this, one runs a *pathwise cyclic coordinate descent method*. We start with a large value for  $\lambda$ , namely, we define

$$\lambda^{\max} = \max_{1 \leq j \leq q} n^{-1} |\langle \mathbf{Y}, \mathbf{x}_j \rangle|.$$

For  $\lambda \geq \lambda^{\max}$ , we have  $\hat{\beta}^{\text{LASSO}}(\lambda) = 0$ , i.e., we have the null model. Pathwise cycle coordinate descent starts with this solution for  $\lambda_0 = \lambda^{\max}$ . In a next step, one slightly decreases  $\lambda_0$  and runs the cyclic coordinate descent algorithm until convergence for this slightly smaller  $\lambda_1 < \lambda_0$ , and with starting value  $\hat{\beta}^{\text{LASSO}}(\lambda_0)$ . This is then iterated for  $\lambda_{t+1} < \lambda_t$ ,  $t \geq 0$ , which provides a sequence of LASSO regularized estimators  $\hat{\beta}^{\text{LASSO}}(\lambda_t)$  along the path  $(\lambda_t)_{t \geq 0}$ .

For further remarks we refer to Section 2.6 in Hastie et al. [184]. This concerns statements about uniqueness for general design matrices, also in the set-up where  $q > n$ , i.e., where we have more parameters than observations. Moreover, references to convergence results are given in Section 2.7 of Hastie et al. [184]. This closes the Gaussian case.

## Gradient Descent Algorithm for LASSO Regularization

In Sect. 7.2.3 we will discuss gradient descent methods for network fitting. In this section we provide preliminary considerations on gradient descent methods because these are also useful to fit LASSO regularized parameters within GLMs (different from Gaussian GLMs). Remark that we do a sign switch in what follows, and we aim at minimizing an objective function  $g$ .

Choose a convex and differentiable function  $g : \mathbb{R}^{q+1} \rightarrow \mathbb{R}$ . Assuming that the global minimum of  $g$  is achieved, a necessary and sufficient condition for the optimality of  $\beta^* \in \mathbb{R}^{q+1}$  in this convex setting is  $\nabla_{\beta} g(\beta)|_{\beta=\beta^*} = 0$ . *Gradient descent algorithms* find this optimal point by iterating for  $t \geq 0$

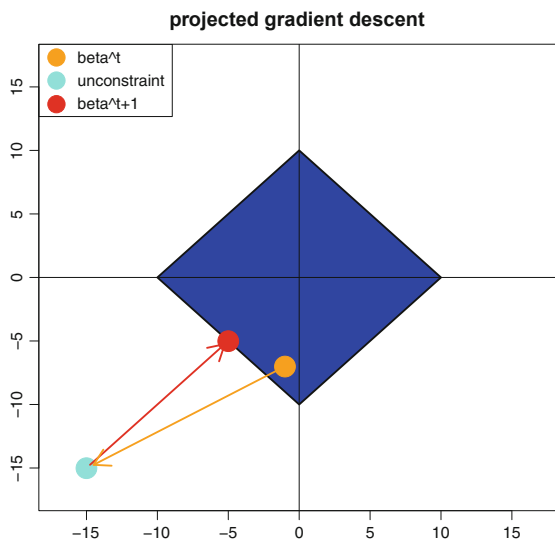
$$\beta^{(t)} \mapsto \beta^{(t+1)} = \beta^{(t)} - \varrho_{t+1} \nabla_{\beta} g(\beta^{(t)}), \quad (6.14)$$

for tempered *learning rates*  $\varrho_{t+1} > 0$ . This algorithm is motivated by a first order Taylor expansion that determines the direction of the maximal local decrease of the objective function  $g$  supposed we are in position  $\beta$ , i.e.,

$$g(\tilde{\beta}) = g(\beta) + \nabla_{\beta} g(\beta)^{\top} (\tilde{\beta} - \beta) + o(\|\tilde{\beta} - \beta\|_2) \quad \text{as } \|\tilde{\beta} - \beta\|_2 \rightarrow 0.$$

The gradient descent algorithm (6.14) leads to the (unconstraint) minimum of the objective function  $g$  at convergence. A budget constraint like (6.6) leads to a convex constraint  $\beta \in \mathcal{C} \subset \mathbb{R}^{q+1}$ . Consideration of such a convex constraint requires that we reformulate the gradient descent algorithm (6.14). The gradient descent step (6.14) can also be found, for given learning rate  $\varrho_{t+1}$ , by solving the following

**Fig. 6.6** Projected gradient descent step, first, mapping  $\beta^{(t)}$  to the unconstrained solution  $\beta^{(t)} - \varrho_{t+1} \nabla_{\beta} g(\beta^{(t)})$  of (6.15) and, second, projecting this unconstrained solution back to the convex set  $\mathcal{C}$  giving  $\beta^{(t+1)}$ ; see also Figure 5.5 in Hastie et al. [184]



linearized problem for  $g$  with the Euclidean square distance penalty term (ridge regularization) for too big gradient descent steps

$$\arg \min_{\beta \in \mathbb{R}^{q+1}} \left\{ g(\beta^{(t)}) + \nabla_{\beta} g(\beta^{(t)})^{\top} (\beta - \beta^{(t)}) + \frac{1}{2\varrho_{t+1}} \|\beta - \beta^{(t)}\|_2^2 \right\}. \quad (6.15)$$

The solution to this optimization problem exactly gives the gradient descent step (6.14). This is now adapted to a constraint gradient descent update for convex constraint  $\mathcal{C}$ :

$$\beta^{(t+1)} = \arg \min_{\beta \in \mathcal{C}} \left\{ g(\beta^{(t)}) + \nabla_{\beta} g(\beta^{(t)})^{\top} (\beta - \beta^{(t)}) + \frac{1}{2\varrho_{t+1}} \|\beta - \beta^{(t)}\|_2^2 \right\}. \quad (6.16)$$

The solution to this constraint convex optimization problem is obtained by, first, taking an unconstrained gradient descent step  $\beta^{(t)} \mapsto \beta^{(t)} - \varrho_{t+1} \nabla_{\beta} g(\beta^{(t)})$ , and, second, if this step is not within the convex set  $\mathcal{C}$ , it is projected back to  $\mathcal{C}$ ; this is illustrated in Fig. 6.6, and it is called *projected gradient descent step* (justification is given in Lemma 6.6 below). Thus, the only difficulty in applying this projected gradient descent step is to find an efficient method of projecting the unconstrained solution (6.14)–(6.15) back to the convex constraint set  $\mathcal{C}$ .

Assume that the convex constraint set  $\mathcal{C}$  is expressed by a convex function  $h$  (not necessarily being differentiable). To solve (6.16) and to motivate the projected gradient descent step, we use the *proximal gradient method* discussed in Section 5.3.3 of Hastie et al. [184]. The proximal gradient method helps us to do the projection in the projected gradient descent step. We introduce the *generalized*

projection operator, for  $\mathbf{z} \in \mathbb{R}^{q+1}$

$$\text{prox}_h(\mathbf{z}) = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \left\{ \frac{1}{2} \|\mathbf{z} - \boldsymbol{\beta}\|_2^2 + h(\boldsymbol{\beta}) \right\}. \quad (6.17)$$

This generalized projection operator should be interpreted as a square minimization problem  $\|\mathbf{z} - \boldsymbol{\beta}\|_2^2/2$  on a convex set  $\mathcal{C}$  being expressed by its dual Lagrangian formulation described by the regularization term  $h(\boldsymbol{\beta})$ . The following lemma shows that the generalized projection operator solves the Lagrangian form of (6.16).

**Lemma 6.6** *Assume the convex constraint  $\mathcal{C}$  is expressed by the convex function  $h$ . The generalized projection operator solves*

$$\begin{aligned} \boldsymbol{\beta}^{(t+1)} &= \text{prox}_{\varrho_{t+1}h} \left( \boldsymbol{\beta}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\beta}} g(\boldsymbol{\beta}^{(t)}) \right) \\ &= \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{q+1}} \left\{ g(\boldsymbol{\beta}^{(t)}) + \nabla_{\boldsymbol{\beta}} g(\boldsymbol{\beta}^{(t)})^\top (\boldsymbol{\beta} - \boldsymbol{\beta}^{(t)}) + \frac{1}{2\varrho_{t+1}} \|\boldsymbol{\beta} - \boldsymbol{\beta}^{(t)}\|_2^2 + h(\boldsymbol{\beta}) \right\}. \end{aligned} \quad (6.18)$$

**Proof of Lemma 6.6** It suffices to consider the following calculation

$$\begin{aligned} &\frac{1}{2} \left\| \boldsymbol{\beta}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\beta}} g(\boldsymbol{\beta}^{(t)}) - \boldsymbol{\beta} \right\|_2^2 + \varrho_{t+1} h(\boldsymbol{\beta}) \\ &= \frac{1}{2} \varrho_{t+1}^2 \left\| \nabla_{\boldsymbol{\beta}} g(\boldsymbol{\beta}^{(t)}) \right\|_2^2 - \varrho_{t+1} \left\langle \nabla_{\boldsymbol{\beta}} g(\boldsymbol{\beta}^{(t)}), \boldsymbol{\beta}^{(t)} - \boldsymbol{\beta} \right\rangle + \frac{1}{2} \left\| \boldsymbol{\beta}^{(t)} - \boldsymbol{\beta} \right\|_2^2 + \varrho_{t+1} h(\boldsymbol{\beta}) \\ &= \frac{1}{2} \varrho_{t+1}^2 \left\| \nabla_{\boldsymbol{\beta}} g(\boldsymbol{\beta}^{(t)}) \right\|_2^2 + \varrho_{t+1} \left( \nabla_{\boldsymbol{\beta}} g(\boldsymbol{\beta}^{(t)})^\top (\boldsymbol{\beta} - \boldsymbol{\beta}^{(t)}) + \frac{1}{2\varrho_{t+1}} \left\| \boldsymbol{\beta}^{(t)} - \boldsymbol{\beta} \right\|_2^2 + h(\boldsymbol{\beta}) \right). \end{aligned}$$

This is exactly the right objective function (in the round brackets) if we ignore all terms that are independent of  $\boldsymbol{\beta}$ . This proves the lemma.  $\square$

Thus, to solve the constraint optimization problem (6.16) we bring it into its dual Lagrangian form (6.18). Then we apply the generalized projection operator to the unconstrained solution to find the constraint solution, see Lemma 6.6. This approach will be successful if we can explicitly compute the generalized projection operator  $\text{prox}_h(\cdot)$ .

**Lemma 6.7** *The generalized projection operator (6.17) satisfies for LASSO constraint  $h(\boldsymbol{\beta}) = \lambda \|\boldsymbol{\beta}_-\|_1$*

$$\text{prox}_h(\mathbf{z}) = \mathcal{S}_\lambda^{\text{LASSO}}(\mathbf{z}) \stackrel{\text{def.}}{=} (z_0, \text{sign}(z_1)(|z_1| - \lambda)_+, \dots, \text{sign}(z_q)(|z_q| - \lambda)_+)^T,$$

for  $\mathbf{z} \in \mathbb{R}^{q+1}$ .



**Proof of Lemma 6.7** We need to solve for function  $\beta \mapsto h(\beta) = \lambda \|\beta_-\|_1$

$$\text{prox}_{\lambda \|\cdot\|_1}(\mathbf{z}) = \arg \min_{\beta \in \mathbb{R}^{q+1}} \left\{ \frac{1}{2} \|\mathbf{z} - \beta\|_2^2 + \lambda \|\beta_-\|_1 \right\} = \arg \min_{\beta \in \mathbb{R}^{q+1}} \left\{ \frac{1}{2} \sum_{j=0}^q (z_j - \beta_j)^2 + \lambda \sum_{j=1}^q |\beta_j| \right\}.$$

This decouples into  $q + 1$  independent optimization problems. The first one is solved by  $\beta_0 = z_0$  and the remaining ones are solved by the soft-thresholding operator (6.13). This finishes the proof.  $\square$

We conclude that the constraint optimization problem (6.16) for the (convex) LASSO constraint  $\mathcal{C} = \{\beta; \|\beta_-\|_1 \leq c\}$  is brought into its dual Lagrangian form (6.18) of Lemma 6.6 with  $h(\beta) = \lambda \|\beta_-\|_1$  for suitable  $\lambda = \lambda(c)$ . The LASSO regularized parameter estimation is then solved by first performing an unconstrained gradient descent step  $\beta^{(t)} \mapsto \beta^{(t)} - \varrho_{t+1} \nabla_{\beta} g(\beta^{(t)})$ , and this updated parameter is projected back to  $\mathcal{C}$  using the generalized projection operator of Lemma 6.7 with  $h(\beta) = \varrho_{t+1} \lambda \|\beta_-\|_1$ .

---

#### Proximal gradient descent algorithm for LASSO

---

1. Make the gradient descent step for a suitable learning rate  $\varrho_{t+1} > 0$

$$\beta^{(t)} \mapsto \tilde{\beta}^{(t+1)} = \beta^{(t)} - \varrho_{t+1} \nabla_{\beta} g(\beta^{(t)}).$$

2. Perform soft-thresholding of the gradient descent solution

$$\tilde{\beta}^{(t+1)} \mapsto \beta^{(t+1)} = \mathcal{S}_{\varrho_{t+1}\lambda}^{\text{LASSO}}(\tilde{\beta}^{(t+1)}),$$

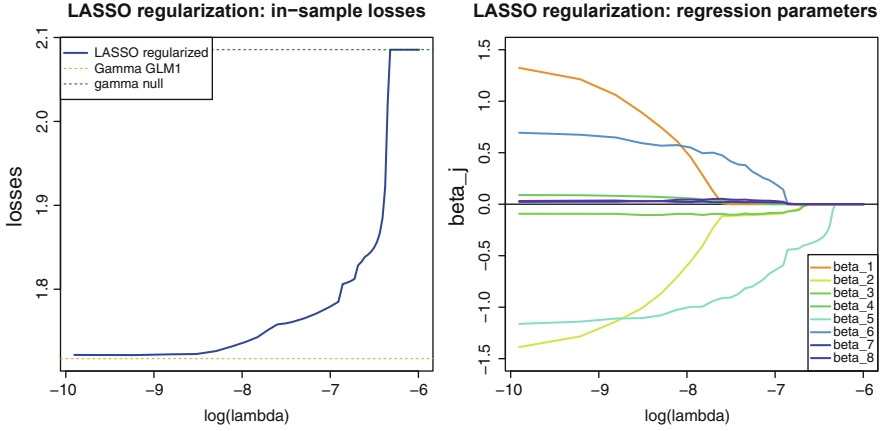
where the latter soft-thresholding function is defined in Lemma 6.7.

3. Iterate these two steps until a stopping criterion is met.
- 

If the gradient  $\nabla_{\beta} g(\cdot)$  is Lipschitz continuous with Lipschitz constant  $L > 0$ , the proximal gradient descent algorithm will converge at rate  $\mathcal{O}(1/t)$  for a fixed step size  $0 < \varrho = \varrho_{t+1} \leq L$ , see Section 4.2 in Parikh–Boyd [292].

**Example 6.8 (LASSO Regression)** We revisit Example 6.5 which considers claim size modeling using model Gamma GLM1. In order to apply the proximal gradient descent algorithm for LASSO regularization we need to calculate the gradient of the negative log-likelihood. In the gamma case with log-link, it is given by, see Example 5.5,

$$\begin{aligned} -\nabla_{\beta} \ell_Y(\beta) &= -\mathfrak{X}^{\top} W(\beta) \mathbf{R}(Y, \beta) \\ &= -\mathfrak{X}^{\top} \text{diag}\left(\frac{n_1}{\varphi}, \dots, \frac{n_m}{\varphi}\right) \left(\frac{Y_1}{\mu_1} - 1, \dots, \frac{Y_m}{\mu_m} - 1\right)^{\top}, \end{aligned}$$



**Fig. 6.7** LASSO regularized MLEs in model Gamma GLM1: (lhs) in-sample losses as a function of the regularization parameter  $\lambda > 0$ , (rhs) resulting  $\hat{\beta}_j^{\text{LASSO}}(\lambda)$  for  $1 \leq j \leq q$

where  $m \in \mathbb{N}$  is the number of policies with claims, and  $\mu_i = \mu_i(\boldsymbol{\beta}) = \exp(\boldsymbol{\beta}, \mathbf{x}_i)$ . We set  $\varphi = 1$  as this constant can be integrated into the learning rates  $\varrho_{t+1}$ .

We have implemented the proximal gradient descent algorithm ourselves using an equidistant grid for the regularization parameter  $\lambda > 0$ , a fixed learning rate  $\varrho_{t+1} = 0.05$  and normalized features. Since this has been done rather brute force, the results presented in Fig. 6.7 look a bit wiggly. These results should be compared to Fig. 6.3. We see that, in contrast to ridge regularization, less important regression parameters are shrunk exactly to zero in LASSO regularization. We give the order in which the parameters are shrunk to zero:  $\beta_1$  (OwnerAge),  $\beta_4$  (RiskClass),  $\beta_6$  (VehAge<sup>2</sup>),  $\beta_8$  (BonusClass),  $\beta_7$  (GenderMale),  $\beta_2$  (OwnerAge<sup>2</sup>),  $\beta_3$  (AreaGLM) and  $\beta_5$  (VehAge). In view of Listing 5.11 this order seems a bit surprising. The reason for this surprising order is that we have grouped features here, and, obviously, these should be considered jointly. In particular, we first drop OwnerAge because this can also be partially explained by OwnerAge<sup>2</sup>, therefore, we should not treat these two variables individually. Having this weakness in mind supports the conclusions drawn from the Wald tests in Listing 5.11, and we come back to this in Example 6.10, below. ■

### Oracle Property

An interesting question is whether the chosen regularization fulfills the so-called oracle property. For simplicity, we assume to work in the normalized Gaussian case that allows us to exclude the intercept  $\beta_0$ , see (6.11). Thus, we work with a regression parameter  $\beta \in \mathbb{R}^q$ . Assume that there is a true data model that can be described by the (true) regression parameter  $\beta^* \in \mathbb{R}^q$ . Denote by  $\mathcal{A}^* = \{j \in \{1, \dots, q\}; \beta_j^* \neq 0\}$  the set of feature components of  $x \in \mathbb{R}^q$  that determine the true regression function, and we assume  $|\mathcal{A}^*| < q$ . Denote by  $\hat{\beta}_n(\lambda)$  the parameter estimate that has been received by the regularized MAP estimation for a given regularization parameter  $\lambda \geq 0$  and based on i.i.d. data of sample size  $n$ . We say that  $(\hat{\beta}_n(\lambda_n))_{n \in \mathbb{N}}$  fulfills the *oracle property* if there exists a sequence  $(\lambda_n)_{n \in \mathbb{N}}$  of regularization parameters  $\lambda_n \geq 0$  such that

$$\lim_{n \rightarrow \infty} \mathbb{P}[\hat{\mathcal{A}}_n = \mathcal{A}^*] = 1, \quad (6.19)$$

$$\sqrt{n} (\hat{\beta}_{n, \mathcal{A}^*}(\lambda_n) - \beta_{\mathcal{A}^*}^*) \Rightarrow \mathcal{N}(0, \mathcal{I}_{\mathcal{A}^*}^{-1}) \quad \text{as } n \rightarrow \infty, \quad (6.20)$$

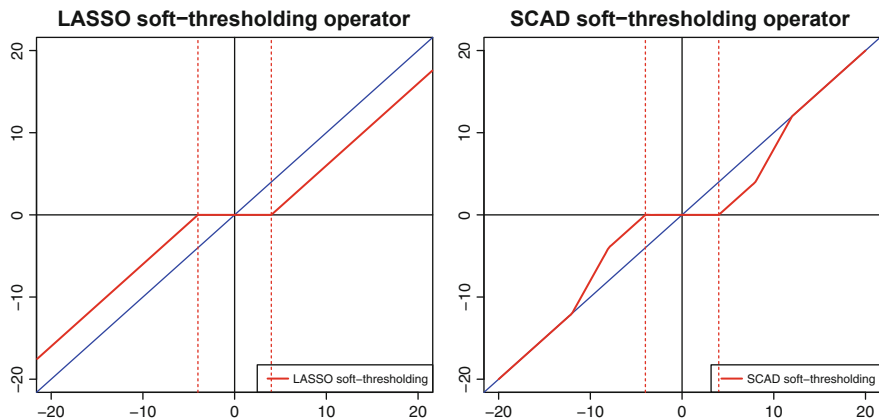
where  $\hat{\mathcal{A}}_n = \{j \in \{1, \dots, q\}; (\hat{\beta}_n(\lambda_n))_j \neq 0\}$ ,  $\beta_{\mathcal{A}}$  only considers the components in  $\mathcal{A} \subset \{1, \dots, q\}$ , and  $\mathcal{I}_{\mathcal{A}^*}$  is Fisher's information matrix on the true feature components. The first oracle property (6.19) tells us that asymptotically we choose the right feature components, and the second oracle property (6.20) tells us that we have asymptotic normality and, in particular, consistency on the right feature components.

Zou [408] states that LASSO regularization, in general, does not satisfy the oracle property. LASSO regularization can perform variable selection, however, as Zou [408] argues, there are situations where consistency is violated and, therefore, the oracle property cannot hold in general. Zou [408] therefore proposes an adaptive LASSO regularization method. Alternatively, Fan–Li [124] introduced smoothly clipped absolute deviation (SCAD) regularization which is a non-convex regularization that possesses the oracle property. SCAD regularization of  $\beta$  is obtained by penalizing

$$J_\lambda(\beta) = \sum_{j=1}^q \lambda |\beta_j| \mathbb{1}_{\{|\beta_j| \leq \lambda\}} - \frac{|\beta_j|^2 - 2a\lambda|\beta_j| + \lambda^2}{2(a-1)} \mathbb{1}_{\{\lambda < |\beta_j| \leq a\lambda\}} + \frac{(a+1)\lambda^2}{2} \mathbb{1}_{\{|\beta_j| > a\lambda\}},$$

for a hyperparameter  $a > 2$ . This function is continuous and differentiable except in  $\beta_j = 0$  with partial derivatives for  $\beta > 0$

$$\lambda \left( \mathbb{1}_{\{\beta \leq \lambda\}} + \frac{(a\lambda - \beta)_+}{\lambda(a-1)} \mathbb{1}_{\{\beta > \lambda\}} \right).$$



**Fig. 6.8** (lhs) LASSO soft-thresholding operator  $x \mapsto \mathcal{S}_\lambda(x)$  for  $\lambda = 4$  (red dotted lines), (rhs) SCAD thresholding operator  $x \mapsto \mathcal{S}_\lambda^{\text{SCAD}}(x)$  for  $\lambda = 4$  and  $a = 3$

Thus, we have a constant LASSO-like slope  $\lambda > 0$  for  $0 < \beta \leq \lambda$ , shrinking some components exactly to zero. For  $\beta > a\lambda$  the slope is 0, removing regularization, and it is concatenated between the two scenarios. The thresholding operator for SCAD regularization is given by, see Fan–Li [124],

$$\mathcal{S}_\lambda^{\text{SCAD}}(x) = \begin{cases} \text{sign}(x)(|x| - \lambda)_+ & \text{for } |x| \leq 2\lambda, \\ \frac{(a-1)x - \text{sign}(x)a\lambda}{a-2} & \text{for } 2\lambda < |x| \leq a\lambda, \\ x & \text{for } |x| > a\lambda. \end{cases}$$

Figure 6.8 compares the two thresholding operators of LASSO and SCAD.

Alternatively, we propose to do variable selection with LASSO regularization in a first step. Since the resulting LASSO regularized estimator may not be consistent, one should explore a second regression step where one uses an un-penalized regression model on the LASSO selected components, we also refer to Lee et al. [237].

### 6.2.5 Group LASSO Regularization

In Example 6.8 we have seen that if there are natural groups within the feature components they should be treated simultaneously. Assume we have a group

structure  $\mathbf{x} = (x_0, \mathbf{x}_1, \dots, \mathbf{x}_K)$  with groups  $\mathbf{x}_k \in \mathbb{R}^{q_k}$  that should be treated simultaneously. This motivates the grouped penalties proposed by Yuan–Lin [398], see (6.5),

$$\widehat{\boldsymbol{\beta}}^{\text{group}} = \widehat{\boldsymbol{\beta}}^{\text{group}}(\lambda) = \arg \max_{\boldsymbol{\beta}=(\beta_0, \boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_K)} \frac{1}{n} \ell_Y(\boldsymbol{\beta}) - \lambda \sum_{k=1}^K \|\boldsymbol{\beta}_k\|_2, \quad (6.21)$$

where we assume a group structure in the linear predictor providing

$$\mathbf{x} \mapsto \eta(\mathbf{x}) = \langle \boldsymbol{\beta}, \mathbf{x} \rangle = \beta_0 + \sum_{k=1}^K \langle \boldsymbol{\beta}_k, \mathbf{x}_k \rangle.$$

LASSO regularization is a special case of this grouped regularization, namely, if all groups  $1 \leq k \leq K$  only contain one single component, i.e.,  $K = q$ , we have  $\widehat{\boldsymbol{\beta}}^{\text{group}} = \widehat{\boldsymbol{\beta}}^{\text{LASSO}}$ .

The side constraint in (6.21) is convex, and the optimization problem (6.21) can again be solved by the proximal gradient descent algorithm. That is, in view of Lemma 6.6, the only difficulty is the calculation of the generalized projection operator for regularization term  $h(\boldsymbol{\beta}) = \lambda \sum_{k=1}^K \|\boldsymbol{\beta}_k\|_2$ . We therefore need to solve for  $\mathbf{z} = (z_0, \mathbf{z}_1, \dots, \mathbf{z}_K)$ ,  $\mathbf{z}_k \in \mathbb{R}^{q_k}$ ,

$$\begin{aligned} \text{prox}_h(\mathbf{z}) &= \arg \min_{\boldsymbol{\beta}=(\beta_0, \boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_K)} \left\{ \frac{1}{2} \|\mathbf{z} - \boldsymbol{\beta}\|_2^2 + \lambda \sum_{k=1}^K \|\boldsymbol{\beta}_k\|_2 \right\} \\ &= \left( z_0, \left( \arg \min_{\boldsymbol{\beta}_k \in \mathbb{R}^{q_k}} \left\{ \frac{1}{2} \|\mathbf{z}_k - \boldsymbol{\beta}_k\|_2^2 + \lambda \|\boldsymbol{\beta}_k\|_2 \right\} \right)_{1 \leq k \leq K} \right). \end{aligned}$$

The latter highlights that the problem decouples into  $K$  independent problems. Thus, we need to solve for all  $1 \leq k \leq K$  the optimization problems

$$\arg \min_{\boldsymbol{\beta}_k \in \mathbb{R}^{q_k}} \left\{ \frac{1}{2} \|\mathbf{z}_k - \boldsymbol{\beta}_k\|_2^2 + \lambda \|\boldsymbol{\beta}_k\|_2 \right\}.$$

**Lemma 6.9** *The group LASSO generalized soft-thresholding operator satisfies for  $\mathbf{z}_k \in \mathbb{R}^{q_k}$*

$$\mathcal{S}_\lambda^{q_k}(\mathbf{z}_k) = \arg \min_{\boldsymbol{\beta}_k \in \mathbb{R}^{q_k}} \left\{ \frac{1}{2} \|\mathbf{z}_k - \boldsymbol{\beta}_k\|_2^2 + \lambda \|\boldsymbol{\beta}_k\|_2 \right\} = \mathbf{z}_k \left( 1 - \frac{\lambda}{\|\mathbf{z}_k\|_2} \right)_+ \in \mathbb{R}^{q_k},$$

and for the generalized projection operator for  $h(\boldsymbol{\beta}) = \lambda \sum_{k=1}^K \|\boldsymbol{\beta}_k\|_2$  we have

$$\text{prox}_h(\mathbf{z}) = \mathcal{S}_\lambda^{\text{group}}(\mathbf{z}) \stackrel{\text{def.}}{=} (\mathbf{z}_0, \mathcal{S}_\lambda^{q_1}(\mathbf{z}_1), \dots, \mathcal{S}_\lambda^{q_K}(\mathbf{z}_K)),$$

for  $\mathbf{z} = (\mathbf{z}_0, \mathbf{z}_1, \dots, \mathbf{z}_K)$  with  $\mathbf{z}_k \in \mathbb{R}^{q_k}$ .

**Proof** We prove this lemma. In a first step we have

$$\arg \min_{\boldsymbol{\beta}_k} \left\{ \frac{1}{2} \|\mathbf{z}_k - \boldsymbol{\beta}_k\|_2^2 + \lambda \|\boldsymbol{\beta}_k\|_2 \right\} = \arg \min_{\boldsymbol{\beta}_k = \varrho \mathbf{z}_k / \|\mathbf{z}_k\|_2, \varrho \geq 0} \left\{ \frac{1}{2} \|\mathbf{z}_k\|_2^2 \left( 1 - \frac{\varrho}{\|\mathbf{z}_k\|_2} \right)^2 + \lambda \varrho \right\},$$

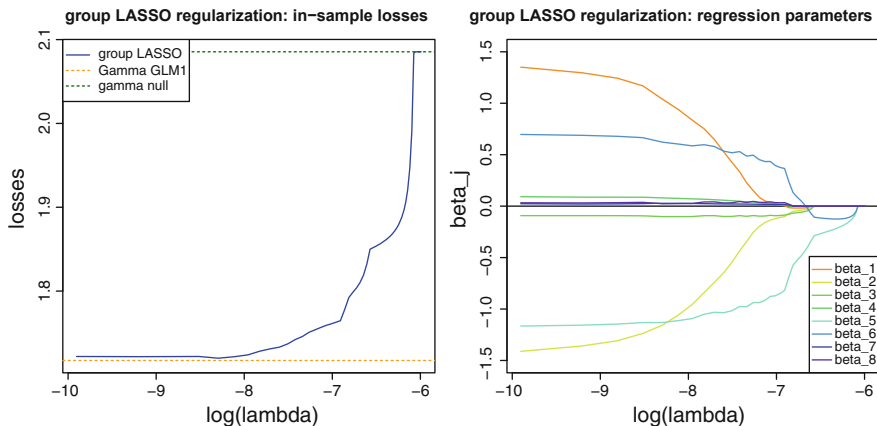
this follows because the square distance  $\|\mathbf{z}_k - \boldsymbol{\beta}_k\|_2^2 = \|\mathbf{z}_k\|_2^2 - 2\langle \mathbf{z}_k, \boldsymbol{\beta}_k \rangle + \|\boldsymbol{\beta}_k\|_2^2$  is minimized if  $\mathbf{z}_k$  and  $\boldsymbol{\beta}_k$  point into the same direction. Thus, there remains the minimization of the objective function in  $\varrho \geq 0$ . The first derivative is given by

$$\frac{\partial}{\partial \varrho} \left( \frac{1}{2} \|\mathbf{z}_k\|_2^2 \left( 1 - \frac{\varrho}{\|\mathbf{z}_k\|_2} \right)^2 + \lambda \varrho \right) = -\|\mathbf{z}_k\|_2 \left( 1 - \frac{\varrho}{\|\mathbf{z}_k\|_2} \right) + \lambda = \lambda - \|\mathbf{z}_k\|_2 + \varrho.$$

If  $\|\mathbf{z}_k\|_2 > \lambda$  we have  $\varrho = \|\mathbf{z}_k\|_2 - \lambda > 0$ , and otherwise we need to set  $\varrho = 0$ . This implies

$$\mathcal{S}_\lambda^{q_k}(\mathbf{z}_k) = (\|\mathbf{z}_k\|_2 - \lambda)_+ \mathbf{z}_k / \|\mathbf{z}_k\|_2.$$

This completes the proof.  $\square$



**Fig. 6.9** Group LASSO regularized MLEs in model Gamma GLM1: (lhs) in-sample losses as a function of the regularization parameter  $\lambda > 0$ , (rhs) resulting  $\tilde{\beta}_j^{\text{group}}(\lambda)$  for  $1 \leq j \leq q$

---

### Proximal gradient descent algorithm for group LASSO

---

1. Make the gradient descent step for a suitable learning rate  $\varrho_{t+1} > 0$

$$\boldsymbol{\beta}^{(t)} \mapsto \tilde{\boldsymbol{\beta}}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\beta}} g(\boldsymbol{\beta}^{(t)}).$$

2. Perform soft-thresholding of the gradient descent solution

$$\tilde{\boldsymbol{\beta}}^{(t+1)} \mapsto \boldsymbol{\beta}^{(t+1)} = \mathcal{S}_{\varrho_{t+1}\lambda}^{\text{group}}(\tilde{\boldsymbol{\beta}}^{(t+1)}),$$

where the latter soft-thresholding function is defined in Lemma 6.9.

3. Iterate these two steps until a stopping criterion is met.
- 

*Example 6.10 (Group LASSO Regression)* We revisit Example 6.8 which considers claim size modeling using model Gamma GLM1. This time we group the variables OwnerAge and OwnerAge<sup>2</sup> ( $\beta_1, \beta_2$ ) as well as VehAge and VehAge<sup>2</sup> ( $\beta_5, \beta_6$ ). The results are shown in Fig. 6.9.

The order in which the parameters are regularized to zero is:  $\beta_4$  (RiskClass),  $\beta_8$  (BonusClass),  $\beta_7$  (GenderMale),  $(\beta_1, \beta_2)$  (OwnerAge, OwnerAge<sup>2</sup>),  $\beta_3$  (AreaGLM) and  $(\beta_5, \beta_6)$  (VehAge, VehAge<sup>2</sup>). This order now reflects more the variable importance as received from the Wald statistics of Listing 5.11, and it shows that grouped features should be regularized jointly in order to determine their importance. ■

## 6.3 Expectation-Maximization Algorithm

### 6.3.1 Mixture Distributions

In many applied problems there does not exist a simple off-the-shelf distribution that is suitable to model the whole range of observations. We think of claim size modeling which may range from small to very large claims; the main body of the data may look like, say, gamma distributed, but the tail of the data being regularly varying. Another related problem is that claims may come from different insurance policy modules. For instance, in property insurance, one can insure water damage, fire, glass and theft claims on the same insurance policy, and feature information about the claim type may not always be available. In such cases, it looks attractive to choose a mixture or a composition of different distributions. In this section we focus on mixtures.

Choose a fixed integer  $K$  bigger than 1 and define the  $(K - 1)$ -unit simplex excluding the edges by

$$\Delta_K = \left\{ \mathbf{p} \in (0, 1)^K; \sum_{k=1}^K p_k = 1 \right\}. \quad (6.22)$$

$\Delta_K$  defines the family of categorical distributions with  $K$  levels (all levels having a strictly positive probability). These distributions belong to the vector-valued parameter EF which we have met in Sects. 2.1.4 and 5.7.

The idea behind mixture distributions is to mix  $K$  different distributions with a mixture probability  $\mathbf{p} \in \Delta_K$ . For instance, we can mix  $K$  different EDF densities  $f_k$  by considering

$$Y \sim \sum_{k=1}^K p_k f_k(y; \theta_k, v/\varphi_k) = \sum_{k=1}^K p_k \exp \left\{ \frac{y\theta_k - \kappa_k(\theta_k)}{\varphi_k/v} + a_k(y; v/\varphi_k) \right\}, \quad (6.23)$$

with cumulant functions  $\theta_k \in \Theta_k \mapsto \kappa_k(\theta_k)$ , exposure  $v > 0$  and dispersion parameters  $\varphi_k > 0$ , for  $1 \leq k \leq K$ .

At the first sight, this does not look very spectacular and parameter estimation seems straightforward. If we consider the log-likelihood of  $n$  independent random variables  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  following mixture density (6.23) we receive log-likelihood function

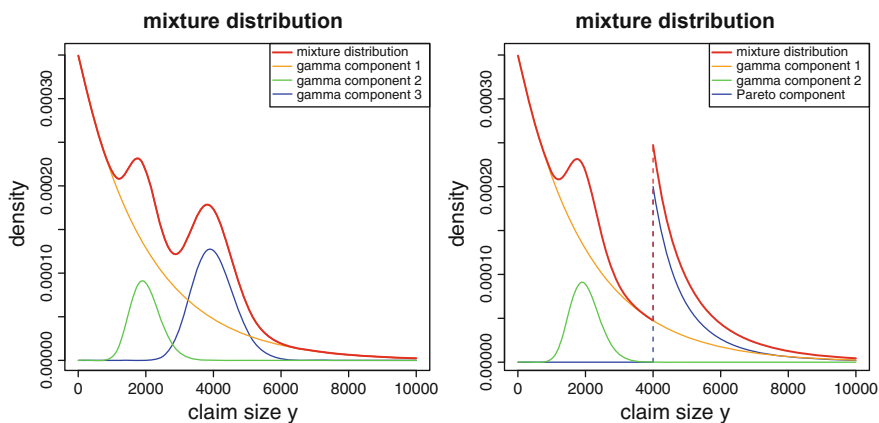
$$(\boldsymbol{\theta}, \mathbf{p}) \mapsto \ell_Y(\boldsymbol{\theta}, \mathbf{p}) = \sum_{i=1}^n \ell_{Y_i}(\boldsymbol{\theta}, \mathbf{p}) = \sum_{i=1}^n \log \left( \sum_{k=1}^K p_k f_k(Y_i; \theta_k, v_i/\varphi_k) \right), \quad (6.24)$$



for canonical parameter  $\boldsymbol{\theta} = (\theta_1, \dots, \theta_K)^\top \in \boldsymbol{\Theta} = \boldsymbol{\Theta}_1 \times \dots \times \boldsymbol{\Theta}_K$  and mixture probability  $\mathbf{p} \in \Delta_K$ . Unfortunately, MLE of  $(\boldsymbol{\theta}, \mathbf{p})$  in (6.24) is not that simple. Note, the summation over  $1 \leq k \leq K$  is inside of the logarithmic function, and the use of the Newton–Raphson algorithm may be cumbersome. The Expectation-Maximization (EM) algorithm presented in Sect. 6.3.3, below, makes parameter estimation feasible. In a nutshell, the EM algorithm leads to a sequence of parameter estimates for  $(\boldsymbol{\theta}, \mathbf{p})$  that monotonically increases the log-likelihood in each iteration of the algorithm. Thus, we can receive an approximation to the MLE of  $(\boldsymbol{\theta}, \mathbf{p})$ .

Nevertheless, model fitting may still be difficult for the following reasons. Firstly, the log-likelihood function of a mixture distribution does not need to be bounded, we highlight this in Example 6.13, below. In that case, MLE is not a well-defined problem. Secondly, even in very simple situations, the log-likelihood function (6.24) can have multiple local maximums. This usually happens if the data is clustered and the clusters are well separated. In that case of multiple local maximums, convergence of the EM algorithm does not guarantee that we have found the global maximum. Thirdly, convergence of the log-likelihood function through the EM algorithm does not guarantee that also the sequence of parameter estimates of  $(\boldsymbol{\theta}, \mathbf{p})$  converges. The latter needs additional examination and regularity conditions.

Figure 6.10 (lhs) shows a density of a mixture distribution mixing  $K = 3$  gamma densities with shape parameters  $\alpha_k = 1, 20, 40$  (orange, green and blue) and mixture probability  $\mathbf{p} = (0.7, 0.1, 0.2)^\top$ ; the mixture components are already multiplied with  $\mathbf{p}$ . The resulting mixture density in red color is continuous. Figure 6.10 (rhs) replaces the blue gamma component of the plot on the left-hand side by a Pareto component (in blue). As a result we observe that the resulting mixture density in red is no longer continuous. This example is often used in practice, however, the discontinuity may be a serious issue in applications and one may use a Lomax (Pareto Type II) component instead, we refer to Sect. 2.2.5.



**Fig. 6.10** (lhs) Mixture distribution mixing three gamma densities, and (rhs) mixture distributions mixing two gamma components and a Pareto component with mixture probabilities  $\mathbf{p} = (0.7, 0.1, 0.2)^\top$  for orange, green and blue components (the density components are already multiplied with  $\mathbf{p}$ )

### 6.3.2 Incomplete and Complete Log-Likelihoods

A mixture distribution can be defined (brute force) by just defining a mixture density as in (6.23). Alternatively, we could define a mixture distribution in a more constructive way. In the following we discuss this constructive derivation which will allow us to efficiently fit mixture distributions to data  $\mathbf{Y}$ . For our outline we focus on (6.23), but all results presented below hold true in much more generality.

Choose a categorical random variable  $Z$  with  $K \geq 2$  levels having probabilities  $\mathbb{P}[Z = k] = p_k > 0$  for  $1 \leq k \leq K$ , that is, with  $\mathbf{p} \in \Delta_K$ . The main idea is to sample in a first step level  $Z = k \in \{1, \dots, K\}$ , and in a second step  $Y|_{\{Z=k\}} \sim f_k(y; \theta_k, v/\varphi_k)$ , based on the selected level  $Z = k$ . The random tuple  $(Y, Z)$  has joint density

$$(Y, Z) \sim f_{\boldsymbol{\theta}, \mathbf{p}}(y, k) = p_k f_k(y; \theta_k, v/\varphi_k),$$

and the marginal density of  $Y$  is exactly given by (6.23). In this interpretation we have a hierarchical model  $(Y, Z)$ . If only  $Y$  is available for parameter estimation, then we are in the situation of *incomplete information* because information about the first hierarchy  $Z$  is missing. If both  $Y$  and  $Z$  are available we say that we have *complete information*.

For the subsequent derivations we use a different coding of the categorical random variable  $Z$ , namely,  $Z$  can be represented in the following one-hot encoding version

$$\mathbf{Z} = (Z_1, \dots, Z_K)^\top = (\mathbb{1}_{\{Z=1\}}, \dots, \mathbb{1}_{\{Z=K\}})^\top, \quad (6.25)$$

these are the  $K$  corners of the  $(K - 1)$ -unit simplex  $\Delta_K$ . One-hot encoding differs from dummy coding (5.21). One-hot encoding does not lead to a full rank design matrix because there is a redundancy, that is, we can drop one component of  $\mathbf{Z}$  and still have the same information. One-hot encoding  $\mathbf{Z}$  of  $Z$  allows us to extend the *incomplete (data) log-likelihood*  $\ell_Y(\boldsymbol{\theta}, \mathbf{p})$ , see (6.23)–(6.24), under complete information  $(Y, \mathbf{Z})$  as follows

$$\begin{aligned} \ell_{(Y, \mathbf{Z})}(\boldsymbol{\theta}, \mathbf{p}) &= \log \left( \prod_{k=1}^K (p_k f_k(Y; \theta_k, v/\varphi_k))^{Z_k} \right) \\ &= \log \left( \prod_{k=1}^K \left( p_k \exp \left\{ \frac{Y\theta_k - \kappa_k(\theta_k)}{\varphi_k/v} + a_k(Y; v/\varphi_k) \right\} \right)^{Z_k} \right) \quad (6.26) \\ &= \sum_{k=1}^K Z_k \left( \log(p_k) + \frac{Y\theta_k - \kappa_k(\theta_k)}{\varphi_k/v} + a_k(Y; v/\varphi_k) \right). \end{aligned}$$

$\ell_{(Y,Z)}(\boldsymbol{\theta}, \mathbf{p})$  is called *complete (data) log-likelihood*. As a consequence of this last expression we observe that under complete information  $(Y_i, \mathbf{Z}_i)_{1 \leq i \leq n}$ , the MLE of  $\boldsymbol{\theta}$  and  $\mathbf{p}$  can be determined completely analogously to above. Namely,  $\theta_k$  is estimated from all observations  $Y_i$  for which  $\mathbf{Z}_i$  belongs to level  $k$ , and the level indicators  $(\mathbf{Z}_i)_{1 \leq i \leq n}$  are used to estimate the mixture probability  $\mathbf{p}$ . Thus, the objective function nicely decouples under complete information into independent parts for  $\theta_k$  and  $\mathbf{p}$  estimation. There remains the question of how to fit this model under incomplete information  $\mathbf{Y}$ . The next section will discuss this problem.

### 6.3.3 Expectation-Maximization Algorithm for Mixtures

The EM algorithm is a general purpose tool for parameter estimation under incomplete information. The EM algorithm has been introduced within the EF by Sundberg [348, 349]. Sundberg's developments have been based on the vector-valued parameter EF with statistics  $S(\mathbf{Y}) \in \mathbb{R}^k$ , see (3.17), and he solved the estimation problem under the assumption that  $S(\mathbf{Y})$  is not fully known. These results have been generalized to MLE under incomplete data in the celebrated work of Dempster et al. [96] and Wu [385]. The monograph of McLachlan–Krishnan [267] gives the theory behind the EM algorithm, and it also provides a historical review in Section 1.8. In actuarial science the EM algorithm is increasingly used to solve various kinds of problems of incomplete data. Mixture models of Erlang distributions are considered in Lee–Lin [240], Yin–Lin [396] and Fung et al. [146, 147]; general Erlang mixtures are universal approximators to positive distributions (in the weak convergence sense), and regularized Erlang mixtures and mixtures of experts models are determined using the EM algorithm to receive approximations to the true underlying model. Miljkovic–Grün [278], Parodi [295] and Fung et al. [148] consider the EM algorithm for mixtures of general distributions, in particular, mixtures of small and large claims distributions. Verbelen et al. [371], Blostein–Miljkovic [40], Grün–Miljkovic [173] and Fung et al. [147] use the EM algorithm for censored and/or truncated observations, and dispersion modeling is performed with the EM algorithm in Tzougas–Karlis [359]. (Inhomogeneous) phase-type and matrix Mittag–Leffler distributions are fitted with the EM algorithm in Asmussen et al. [14], Albrecher et al. [8] and Bladt [37], and the EM algorithm is used to fit mixture density networks (MDNs) in DeLong et al. [95]. Parameter uncertainty is investigated in O'Hagan et al. [289] using the bootstrap method. The present section is mainly based on McLachlan–Krishnan [267].

As mentioned above, the EM algorithm is a general purpose tool for parameter estimation under incomplete data, and we describe the variant of the EM algorithm which is useful for our mixture distribution setup given in (6.26). We give a justification for its functioning below. The EM algorithm is an iterative algorithm that performs a Bayesian expectation step (E-step) to infer the latent variable  $\mathbf{Z}$ , given the model parameters and  $\mathbf{Y}$ . Next, it performs a maximization step (M-step) for MLE of the parameters given the observation  $\mathbf{Y}$  and the estimated latent variable  $\hat{\mathbf{Z}}$ . More specifically, the E-step and the M-step look as follows.

- **E-step.** Calculate the posterior probability of the event that a given observation  $Y$  has been generated from the  $k$ -th component of the mixture distribution. Bayes' rule allows us to infer this posterior probability (for given  $\theta$  and  $p$ ) from (6.26)

$$\mathbb{P}_{\theta, p}[Z_k = 1|Y] = \frac{p_k f_k(Y; \theta_k, v/\varphi_k)}{\sum_{l=1}^K p_l f_l(Y; \theta_l, v/\varphi_l)}.$$

The posterior (Bayesian) estimate for  $Z_k$  after having observed  $Y$  is given by

$$\widehat{Z}_k(\theta, p|Y) \stackrel{\text{def.}}{=} \mathbb{E}_{\theta, p}[Z_k|Y] = \mathbb{P}_{\theta, p}[Z_k = 1|Y] \quad \text{for } 1 \leq k \leq K. \quad (6.27)$$

This posterior mean  $\widehat{\mathbf{Z}} = \widehat{\mathbf{Z}}(\theta, p|Y) = (\widehat{Z}_1(\theta, p|Y), \dots, \widehat{Z}_K(\theta, p|Y))^T \in \Delta_K$  is used as an estimate for the (unobserved) latent variable  $\mathbf{Z}$ ; note that this posterior mean depends on the unknown parameters  $(\theta, p)$ .

- **M-step.** Based on  $Y$  and  $\widehat{\mathbf{Z}}$  the parameters  $\theta$  and  $p$  are estimated with MLE.

Alternation of these two steps provide the following recursive algorithm. We assume to have independent responses  $(Y_i, \mathbf{Z}_i)$ ,  $1 \leq i \leq n$ , following the mixture distribution (6.26), where, for simplicity, we assume that only the volumes  $v_i > 0$  are dependent on  $i$ .

---

#### EM algorithm for mixture distributions

---

(0) Choose an initial parameter  $(\widehat{\theta}^{(0)}, \widehat{p}^{(0)}) \in \Theta \times \Delta_K$ .

(1) Repeat for  $t \geq 1$  until a stopping criterion is met:

- **E-step.** Given parameter  $(\widehat{\theta}^{(t-1)}, \widehat{p}^{(t-1)}) \in \Theta \times \Delta_K$  estimate the latent variables  $\mathbf{Z}_i$ ,  $1 \leq i \leq n$ , by their conditional expectations, see (6.27),

$$\widehat{\mathbf{Z}}_i^{(t)} = \widehat{\mathbf{Z}} \left( \widehat{\theta}^{(t-1)}, \widehat{p}^{(t-1)} \middle| Y_i \right) = \mathbb{E}_{\widehat{\theta}^{(t-1)}, \widehat{p}^{(t-1)}}[\mathbf{Z}_i | Y_i] \in \Delta_K. \quad (6.28)$$

- **M-step.** Calculate the MLE  $(\widehat{\theta}^{(t)}, \widehat{p}^{(t)}) \in \Theta \times \Delta_K$  based on (complete) observations  $((Y_1, \widehat{\mathbf{Z}}_1^{(t)}), \dots, (Y_n, \widehat{\mathbf{Z}}_n^{(t)}))$ , i.e., solve the score equations,

see (6.26),

$$\nabla_{\boldsymbol{\theta}} \left( \sum_{i=1}^n \sum_{k=1}^K \widehat{Z}_{i,k}^{(t)} \frac{Y_i \theta_k - \kappa_k(\theta_k)}{\varphi_k/v_i} \right) = 0, \quad (6.29)$$

$$\nabla_{\mathbf{p}_-} \left( \sum_{i=1}^n \sum_{k=1}^K \widehat{Z}_{i,k}^{(t)} \log(p_k) \right) = 0, \quad (6.30)$$

where  $\mathbf{p}_- = (p_1, \dots, p_{K-1})^\top$  and setting  $p_K = 1 - \sum_{k=1}^{K-1} p_k \in (0, 1)$ .

---

#### Remarks 6.11

- The E-step uses Bayes' rule. This motivates to consider the EM algorithm in this Bayesian chapter; alternatively, it also fits to the MLE chapters.
- We have formulated the M-step in (6.29)–(6.30) in a general way because the canonical parameter  $\boldsymbol{\theta}$  and the mixture probability  $\mathbf{p}$  could be modeled by GLMs, and, henceforth, they may be feature  $\mathbf{x}_i$  dependent. Moreover, (6.29) is formulated for a mixture of single-parameter EDF distributions, but, of course, this holds in much more generality.
- Equations (6.29)–(6.30) are the score equations received from (6.26). There is a subtle point here, namely,  $Z_k \in \{0, 1\}$  in (6.26) are observations, whereas  $\widehat{Z}_{i,k}^{(t)} \in (0, 1)$  in (6.29)–(6.30) are their estimates. Thus, in the EM algorithm the unknown latent variables are replaced by their estimates which, in our setup, results in two different types of variables with disjoint ranges. This may matter in software implementations, for instance, a categorical GLM may ask for a categorical random variable  $Z \in \{1, \dots, K\}$  (of factor type), whereas  $\widehat{\mathbf{Z}}$  is in the interior of the unit simplex  $\Delta_K$ .
- For mixture distributions one can replace the latent variables  $\mathbf{Z}_i$  by their conditionally expected values  $\widehat{\mathbf{Z}}_i$ , see (6.29)–(6.30). In general, this does not hold true in EM algorithm applications: in our case we benefit from the fact that  $Z_k$  influences the complete log-likelihood *linearly*, see (6.26). In the general (non-linear) case of the EM algorithm application, different from mixture distribution problems, one needs to calculate the conditional expectation of the log-likelihood function.

- If we calculate the scores element-wise we receive

$$\frac{\partial}{\partial \theta_k} \sum_{i=1}^n \frac{Y_i \theta_k - \kappa_k(\theta_k)}{\varphi_k / (v_i \widehat{Z}_{i,k}^{(t)})} = 0,$$

$$\frac{\partial}{\partial p_k} \sum_{i=1}^n \left( \widehat{Z}_{i,k}^{(t)} \log(p_k) + \widehat{Z}_{i,K}^{(t)} \log(p_K) \right) = 0,$$

recall normalization  $p_K = 1 - \sum_{k=1}^{K-1} p_k \in (0, 1)$ .

From the first score equation we see that we receive the classical MLE/GLM framework, and all tools introduced above for parameter estimation can directly be used. The only part that changes are the weights  $v_i \mapsto v_i \widehat{Z}_{i,k}^{(t)}$ . In the homogeneous case, i.e., in the null model we have MLE after the  $t$ -th iteration of the EM algorithm

$$\widehat{\theta}_k^{(t)} = h_k \left( \frac{\sum_{i=1}^n v_i \widehat{Z}_{i,k}^{(t)} Y_i}{\sum_{i=1}^n v_i \widehat{Z}_{i,k}^{(t)}} \right),$$

where  $h_k$  is the canonical link that corresponds to cumulant function  $\kappa_k$ .

If we choose the null model for the mixture probabilities we receive MLEs

$$\widehat{p}_k^{(t)} = \frac{1}{n} \sum_{i=1}^n \widehat{Z}_{i,k}^{(t)} \quad \text{for } 1 \leq k \leq K. \quad (6.31)$$

In Sect. 6.3.4, below, we will present an example that uses the null model for the mixture probabilities  $\mathbf{p}$ , and we present an other example that uses a logistic categorical GLM for these mixture probabilities.

**Justification of the EM Algorithm** So far, we have neither given any argument why the EM algorithm is reasonable for parameter estimation nor have we said anything about convergence. The purpose of this paragraph is to justify the above EM algorithm. We aim at solving the incomplete log-likelihood maximization problem, see (6.24),

$$(\widehat{\boldsymbol{\theta}}^{\text{MLE}}, \widehat{\mathbf{p}}^{\text{MLE}}) = \arg \max_{(\boldsymbol{\theta}, \mathbf{p})} \ell_Y(\boldsymbol{\theta}, \mathbf{p}) = \arg \max_{(\boldsymbol{\theta}, \mathbf{p})} \sum_{i=1}^n \log \left( \sum_{k=1}^K p_k f_k(Y_i; \boldsymbol{\theta}_k, v_i / \varphi_k) \right),$$

subject to existence and uniqueness. We introduce some notation. Let  $f(y, \mathbf{z}; \boldsymbol{\theta}, \mathbf{p}) = \exp\{\ell_{(y, \mathbf{z})}(\boldsymbol{\theta}, \mathbf{p})\}$  be the joint density of  $(Y, \mathbf{Z})$  and let  $f(y; \boldsymbol{\theta}, \mathbf{p}) =$

$\exp\{\ell_Y(\boldsymbol{\theta}, \mathbf{p})\}$  be the marginal density of  $Y$ . This allows us to rewrite the incomplete log-likelihood as follows for any value of  $\mathbf{z}$

$$\ell_Y(\boldsymbol{\theta}, \mathbf{p}) = \log f(Y; \boldsymbol{\theta}, \mathbf{p}) = \log \left( \frac{f(Y, \mathbf{z}; \boldsymbol{\theta}, \mathbf{p})}{f(\mathbf{z}|Y; \boldsymbol{\theta}, \mathbf{p})} \right),$$

thus, we bring in the complete log-likelihood by using Bayes' rule. Choose an arbitrary categorical distribution  $\pi \in \Delta_K$  with  $K$  levels. We have using the previous step

$$\begin{aligned} \ell_Y(\boldsymbol{\theta}, \mathbf{p}) &= \log f(Y; \boldsymbol{\theta}, \mathbf{p}) = \sum_{\mathbf{z}} \pi(\mathbf{z}) \log f(Y; \boldsymbol{\theta}, \mathbf{p}) \\ &= \sum_{\mathbf{z}} \pi(\mathbf{z}) \log \left( \frac{f(Y, \mathbf{z}; \boldsymbol{\theta}, \mathbf{p})/\pi(\mathbf{z})}{f(\mathbf{z}|Y; \boldsymbol{\theta}, \mathbf{p})/\pi(\mathbf{z})} \right) \\ &= \sum_{\mathbf{z}} \pi(\mathbf{z}) \log \left( \frac{f(Y, \mathbf{z}; \boldsymbol{\theta}, \mathbf{p})}{\pi(\mathbf{z})} \right) + \sum_{\mathbf{z}} \pi(\mathbf{z}) \log \left( \frac{\pi(\mathbf{z})}{f(\mathbf{z}|Y; \boldsymbol{\theta}, \mathbf{p})} \right) \\ &= \sum_{\mathbf{z}} \pi(\mathbf{z}) \log \left( \frac{f(Y, \mathbf{z}; \boldsymbol{\theta}, \mathbf{p})}{\pi(\mathbf{z})} \right) + D_{\text{KL}}(\pi || f(\cdot|Y; \boldsymbol{\theta}, \mathbf{p})) \quad (6.32) \\ &\geq \sum_{\mathbf{z}} \pi(\mathbf{z}) \log \left( \frac{f(Y, \mathbf{z}; \boldsymbol{\theta}, \mathbf{p})}{\pi(\mathbf{z})} \right), \end{aligned}$$

the inequality follows because the KL divergence is always non-negative, see Lemma 2.21. This provides us with a lower bound for the incomplete log-likelihood  $\ell_Y(\boldsymbol{\theta}, \mathbf{p})$  for any categorical distribution  $\pi \in \Delta_K$  and any  $(\boldsymbol{\theta}, \mathbf{p}) \in \boldsymbol{\Theta} \times \Delta_K$ :

$$\begin{aligned} \ell_Y(\boldsymbol{\theta}, \mathbf{p}) &\geq \sum_{\mathbf{z}} \pi(\mathbf{z}) \log \left( \frac{f(Y, \mathbf{z}; \boldsymbol{\theta}, \mathbf{p})}{\pi(\mathbf{z})} \right) \quad (6.33) \\ &= \mathbb{E}_{\mathbf{Z} \sim \pi} [\ell_{(Y, \mathbf{Z})}(\boldsymbol{\theta}, \mathbf{p}) | Y] - \sum_{\mathbf{z}} \pi(\mathbf{z}) \log(\pi(\mathbf{z})) \stackrel{\text{def.}}{=} \mathcal{Q}(\boldsymbol{\theta}, \mathbf{p}; \pi). \end{aligned}$$

Thus, we have a lower bound  $\mathcal{Q}(\boldsymbol{\theta}, \mathbf{p}; \pi)$  on the incomplete log-likelihood  $\ell_Y(\boldsymbol{\theta}, \mathbf{p})$ . This lower bound is based on the conditionally expected complete log-likelihood  $\ell_{(Y, \mathbf{Z})}(\boldsymbol{\theta}, \mathbf{p})$ , given  $Y$ , and under an arbitrary choice  $\pi$  for  $\mathbf{Z}$ . The difference between this arbitrary  $\pi$  and the true conditional posterior distribution is given by the KL divergence  $D_{\text{KL}}(\pi || f(\cdot|Y; \boldsymbol{\theta}, \mathbf{p}))$ , see (6.32).

The general idea of the EM algorithm is to make this lower bound  $\mathcal{Q}(\boldsymbol{\theta}, \mathbf{p}; \pi)$  as large as possible in  $\boldsymbol{\theta}$ ,  $\mathbf{p}$  and  $\pi$  by iterating the following two alternating steps for  $t \geq 1$ :

$$\hat{\pi}^{(t)} = \arg \max_{\pi} \mathcal{Q}(\hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)}; \pi), \quad (6.34)$$

$$(\hat{\boldsymbol{\theta}}^{(t)}, \hat{\mathbf{p}}^{(t)}) = \arg \max_{\boldsymbol{\theta}, \mathbf{p}} \mathcal{Q}(\boldsymbol{\theta}, \mathbf{p}; \hat{\pi}^{(t)}). \quad (6.35)$$

The first step (6.34) can be solved explicitly and it results in the E-step. Namely, from (6.32) we see that maximizing  $\mathcal{Q}(\hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)}; \pi)$  in  $\pi$  is equivalent to minimizing the KL divergence  $D_{\text{KL}}(\pi \| f(\cdot | Y; \hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)}))$  in  $\pi$  because the left-hand side of (6.32) is independent of  $\pi$ . Thus, we have to solve

$$\hat{\pi}^{(t)} = \arg \max_{\pi} \mathcal{Q}(\hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)}; \pi) = \arg \min_{\pi} D_{\text{KL}}(\pi \| f(\cdot | Y; \hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)})).$$

This optimization is solved by choosing the density  $\hat{\pi}^{(t)} = f(\cdot | Y; \hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)})$ , see Lemma 2.21, and this gives us exactly (6.28) if we calculate the corresponding conditional expectation of the latent variable  $\mathbf{Z}$ . Moreover, importantly, this step provides us with an identity in (6.33):

$$\ell_Y(\hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)}) = \mathcal{Q}(\hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)}; \hat{\pi}^{(t)}). \quad (6.36)$$

The second step (6.35) then increases the right-hand side of (6.36). This second step is equivalent to

$$(\hat{\boldsymbol{\theta}}^{(t)}, \hat{\mathbf{p}}^{(t)}) = \arg \max_{\boldsymbol{\theta}, \mathbf{p}} \mathcal{Q}(\boldsymbol{\theta}, \mathbf{p}; \hat{\pi}^{(t)}) = \arg \max_{\boldsymbol{\theta}, \mathbf{p}} \mathbb{E}_{\mathbf{Z} \sim \hat{\pi}^{(t)}} [\ell_{(Y, \mathbf{Z})}(\boldsymbol{\theta}, \mathbf{p}) | Y], \quad (6.37)$$

and this maximization is solved by the solution of the score equations (6.29)–(6.30) of the M-step. In this step we explicitly use the linearity in  $\mathbf{Z}$  of the log-likelihood  $\ell_{(Y, \mathbf{Z})}$ , which allows us to calculate the objective function in (6.37) explicitly resulting in replacing  $\mathbf{Z}$  by  $\hat{\mathbf{Z}}^{(t)}$ . For other incomplete data problems, where we do not have this linearity, this step will be more complicated.

Summarizing, alternating optimizations (6.34) and (6.35) gives us a sequence of parameters  $(\hat{\boldsymbol{\theta}}^{(t)}, \hat{\mathbf{p}}^{(t)})_{t \geq 0}$  with monotonically increasing incomplete log-likelihoods

$$\dots \leq \ell_Y(\hat{\boldsymbol{\theta}}^{(t-1)}, \hat{\mathbf{p}}^{(t-1)}) \leq \ell_Y(\hat{\boldsymbol{\theta}}^{(t)}, \hat{\mathbf{p}}^{(t)}) \leq \ell_Y(\hat{\boldsymbol{\theta}}^{(t+1)}, \hat{\mathbf{p}}^{(t+1)}) \leq \dots \quad (6.38)$$



Therefore, the EM algorithm converges supposed that the incomplete log-likelihood  $\ell_Y(\boldsymbol{\theta}, \mathbf{p})$  is a bounded function.

### Remarks 6.12

- In general, the log-likelihood function  $(\boldsymbol{\theta}, \mathbf{p}) \mapsto \ell_Y(\boldsymbol{\theta}, \mathbf{p})$  does not need to be bounded. In that case the EM algorithm may not converge (unless it converges to a local maximum). An illustrative example is given in Example 6.13, below, which shows what can go wrong in MLE of mixture distributions.
- Even if the log-likelihood function  $(\boldsymbol{\theta}, \mathbf{p}) \mapsto \ell_Y(\boldsymbol{\theta}, \mathbf{p})$  is bounded, one may not expect a unique solution to the parameter estimation problem with the EM algorithm. Firstly, a monotonically increasing sequence (6.38) only guarantees that we have convergence of that sequence. But the sequence may not converge to the global maximum and different starting points of the algorithm need to be explored. Secondly, convergence of sequence (6.38) does not necessarily imply that the parameters  $(\hat{\boldsymbol{\theta}}^{(t)}, \hat{\mathbf{p}}^{(t)})$  converge for  $t \rightarrow \infty$ . On the one hand, we may have an identifiability issue because the components  $f_k$  of the mixture distribution may be exchangeable, and secondly one needs stronger conditions to ensure that not only the log-likelihoods converge but also their arguments (parameters)  $(\hat{\boldsymbol{\theta}}^{(t)}, \hat{\mathbf{p}}^{(t)})$ . This is the point studied in Wu [385].
- Even in very simple examples of mixture distributions we can have multiple local maximums. In this case the role of the starting point plays a crucial role. It is advantageous that in the starting configuration every component  $k$  shares roughly the same number of observations for the initial estimates  $(\hat{\boldsymbol{\theta}}^{(0)}, \hat{\mathbf{p}}^{(0)})$  and  $\hat{\mathbf{Z}}^{(1)}$ , otherwise one may start in a so-called spurious configuration where only a few observations almost fully determine a component  $k$  of the mixture distribution. This may result in similar singularities as in Example 6.13, below. Therefore, there are three common ways to determine a starting configuration of the EM algorithm, see Miljkovic–Grün [278]: (a) Euclidean distance-based initialization: cluster centers are selected at random, and all observations are allocated to these centers according to the shortest Euclidean distance; (b)  $K$ -means clustering allocation; or (c) completely random allocation to  $K$  bins. Using one of these three options,  $f_k$  and  $\mathbf{p}$  are initialized.
- We have formulated the EM algorithm in the homogeneous situation. However, we can easily expand it to GLMs by, for instance, assuming that the canonical parameters  $\theta_k$  are modeled by linear predictors  $\langle \boldsymbol{\beta}_k, \mathbf{x} \rangle$  and/or likewise for the mixture probabilities  $\mathbf{p}$ . The E-step will not change in this setup. For the M-step, we will solve a different maximization problem, however, this maximization problem respects monotonicity (6.38), and therefore a modified version of the above EM algorithm applies. We emphasize that the crucial point is monotonicity (6.38) that makes the EM algorithm a valid procedure.

### 6.3.4 Lab: Mixture Distribution Applications

In this section we are going to present different mixture distribution examples that use the EM algorithm for parameter estimation. On the one hand this illustrates the functioning of the EM algorithm, and on the other hand it also highlights pitfalls that need to be avoided.

*Example 6.13 (Gaussian Mixture)* We directly fit a mixture model to the observation  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$ . Assume that the log-likelihood of  $\mathbf{Y}$  is given by a mixture of two Gaussian distributions

$$\ell_Y(\boldsymbol{\theta}, \boldsymbol{\sigma}, \mathbf{p}) = \sum_{i=1}^n \log \left( \sum_{k=1}^2 p_k \frac{1}{\sqrt{2\pi} \sigma_k} \exp \left\{ -\frac{1}{2\sigma_k^2} (Y_i - \theta_k)^2 \right\} \right),$$

with  $\mathbf{p} \in \Delta_2$ , mean vector  $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top \in \mathbb{R}^2$  and standard deviations  $\boldsymbol{\sigma} = (\sigma_1, \sigma_2)^\top \in \mathbb{R}_+^2$ . Choose estimate  $\hat{\theta}_1 = Y_1$ , then we have

$$\lim_{\sigma_1 \rightarrow 0} \frac{1}{\sqrt{2\pi} \sigma_1} \exp \left\{ -\frac{1}{2\sigma_1^2} (Y_1 - \hat{\theta}_1)^2 \right\} = \lim_{\sigma_1 \rightarrow 0} \frac{1}{\sqrt{2\pi} \sigma_1} = \infty.$$

For any  $i \neq 1$  we have  $Y_i \neq \hat{\theta}_1$  (note that the Gaussian distribution is absolutely continuous and observations are distinct, a.s.). Henceforth for  $i \neq 1$

$$\lim_{\sigma_1 \rightarrow 0} \frac{1}{\sqrt{2\pi} \sigma_1} \exp \left\{ -\frac{1}{2\sigma_1^2} (Y_i - \hat{\theta}_1)^2 \right\} = \lim_{\sigma_1 \rightarrow 0} \frac{1}{\sqrt{2\pi}} \exp \left\{ -\frac{1}{2\sigma_1^2} (Y_i - \hat{\theta}_1)^2 - \log \sigma_1 \right\} = 0.$$

If we choose any  $\hat{\theta}_2 \in \mathbb{R}$ ,  $\mathbf{p} \in \Delta_2$  and  $\sigma_2 > 0$ , we receive for  $\hat{\theta}_1 = Y_1$

$$\begin{aligned} \lim_{\sigma_1 \rightarrow 0} \ell_Y(\hat{\boldsymbol{\theta}}, \boldsymbol{\sigma}, \mathbf{p}) &= \lim_{\sigma_1 \rightarrow 0} \log \left( \sum_{k=1}^2 p_k \frac{1}{\sqrt{2\pi} \sigma_k} \exp \left\{ -\frac{1}{2\sigma_k^2} (Y_1 - \hat{\theta}_k)^2 \right\} \right) \\ &\quad + \sum_{i=2}^n \log \left( \frac{p_2}{\sqrt{2\pi} \sigma_2} \right) - \frac{1}{2\sigma_2^2} (Y_i - \hat{\theta}_2)^2 = \infty. \end{aligned}$$

Thus, we can make the log-likelihood of this mixture Gaussian model arbitrarily large by fitting a degenerate Gaussian model to one observation in one mixture component, and letting the remaining observations be described by the other mixture component. This shows that the MLE problem may not be well-posed for mixture distributions because the log-likelihood can be unbounded.

If the data has well separated clusters, the log-likelihood of a mixture Gaussian distribution will have multiple local maximums. One can construct for any given

number  $B \in \mathbb{N}$  a data set  $\mathbf{Y}$  such that the number of local maximums exceeds this number  $B$ , see Theorem 3 in Améndola et al. [11]. ■

*Example 6.14 (Gamma Claim Size Modeling)* In this example we consider claim size modeling of the French MTPL example given in Chap. 13.1. In view of Fig. 13.15 this seems quite difficult because we have three modes and heavy-tailedness. We choose a mixture of 5 distribution functions, we choose four gamma distributions and the Lomax distribution

$$Y \sim \sum_{k=1}^4 \left( p_k \frac{\beta_k^{\alpha_k}}{\Gamma(\alpha_k)} y^{\alpha_k-1} \exp\{-\beta_k y\} \right) + p_5 \frac{\beta_5}{M} \left( \frac{y+M}{M} \right)^{-(\beta_5+1)}, \quad (6.39)$$

with shape parameters  $\alpha_k$  and scale parameters  $\beta_k$ ,  $1 \leq k \leq 4$ , for the gamma densities; scale parameter  $M$  and tail parameter  $\beta_5$  for the Lomax density; and with mixture probability  $\mathbf{p} \in \Delta_5$ . The idea behind this choice is that three gamma distributions take care of the three modes of the empirical density, see Fig. 13.15, the fourth gamma distribution models the remaining claims in the body of the distribution, and the Lomax distribution takes care of the regularly varying tail of the data. For the gamma distribution, we refer to Sect. 2.1.3, and for the Lomax distribution, we refer to Sect. 2.2.5.

We choose the null model for both the mixture probabilities  $\mathbf{p} \in \Delta_5$  and the densities  $f_k$ ,  $1 \leq k \leq 5$ . This model can directly be fitted with the EM algorithm as presented above, in particular, we can estimate the mixture probabilities by (6.31). The remaining shape, scale and tail parameters are directly estimated by MLE. To initialize the EM algorithm we use the interpretation of the components as explained above. We partition the entire data into  $K = 5$  bins according to their claim sizes  $Y_i$  being in  $(0, 300]$ ,  $(300, 1'000]$ ,  $(1'000, 1'200]$ ,  $(1'200, 5'000]$  or  $(5'000, \infty)$ . The first three intervals will initialize the three modes of the empirical density, see Fig. 13.15 (lhs). This will correspond to the categorical variable taking values  $Z = 1, 2, 3$ ; the fourth interval will correspond to  $Z = 4$  and it will model the main body of the claims; and the last interval will correspond to  $Z = 5$ , modeling the Lomax tail of the claims. These choices provide the initialization given in Table 6.1 with upper indices  $^{(0)}$ . We remark that we choose a fixed threshold of  $M = 2'000$  for the Lomax distribution, this choice will be further discussed below.

Based on these choices we run the EM algorithm for mixture distributions. We observe convergence after roughly 80 iterations, and the resulting parameters after 100 iterations are presented in Table 6.1. We observe rather large shape parameters  $\hat{\alpha}_k^{(100)}$  for the first three components  $k = 1, 2, 3$ . This indicates that these three components model the three modes of the empirical density and these three modes collect almost  $\hat{p}_1^{(100)} + \hat{p}_2^{(100)} + \hat{p}_3^{(100)} \approx 50\%$  of all claims. The remaining claims are modeled by the gamma density  $k = 4$  having mean 1'304 and by the Lomax distribution having tail parameter  $\hat{\beta}_5^{(100)} = 1.416$ , thus, this tail has finite first moment  $M/(\hat{\beta}_5^{(100)} - 1) = 4'812$  and infinite second moment.

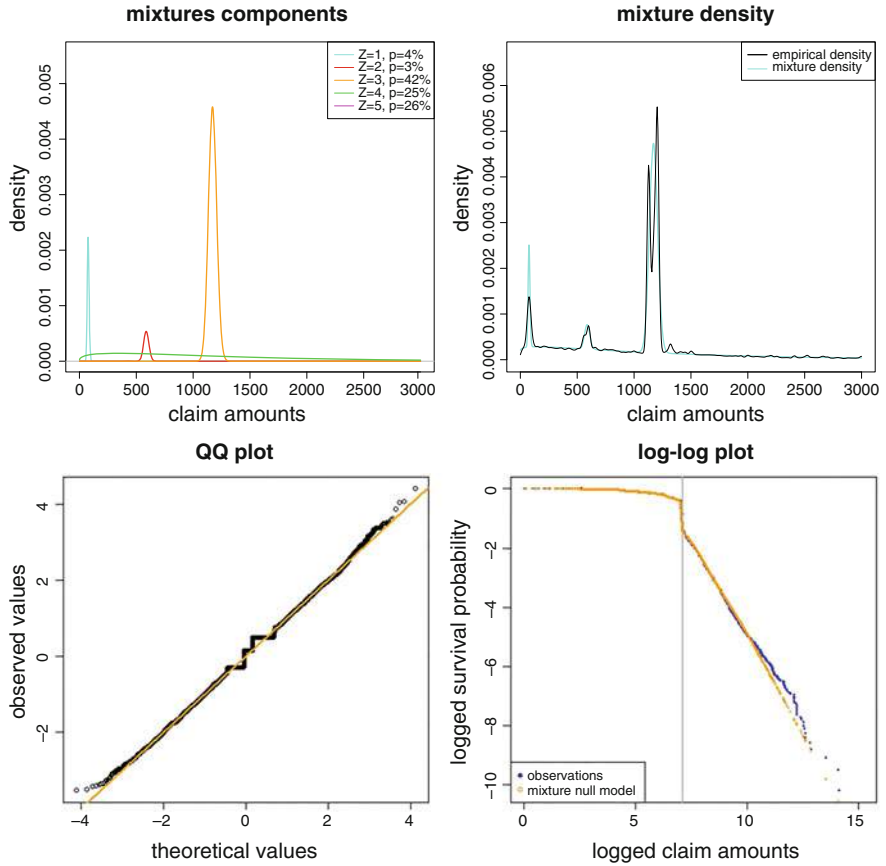
**Table 6.1** Parameter choices in the mixture model (6.39)

	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
$\hat{p}_k^{(0)}$	0.13	0.18	0.25	0.39	0.05
$\hat{\alpha}_k^{(0)}$	2.43	11.24	1'299.44	5.63	–
$\hat{\beta}_k^{(0)}$	0.019	0.018	1.141	0.003	0.517
$\hat{\mu}_k^{(0)} = \hat{\alpha}_k^{(0)} / \hat{\beta}_k^{(0)}$	125	623	1'138	1'763	–
$\hat{p}_k^{(100)}$	0.04	0.03	0.42	0.25	0.26
$\hat{\alpha}_k^{(100)}$	93.05	650.94	1'040.37	1.34	–
$\hat{\beta}_k^{(100)}$	1.207	1.108	0.888	0.001	1.416
$\hat{\mu}_k^{(100)} = \hat{\alpha}_k^{(100)} / \hat{\beta}_k^{(100)}$	77	588	1'172	1'304	–

Figure 6.11 shows the resulting estimated mixture distribution. It gives the individual mixture components (top-lhs), the resulting mixture density (top-rhs), the QQ plot (bottom-lhs) and the log-log plot (bottom-rhs). Overall we find a rather good fit; maybe the first mode is a bit too spiky. However, this plot may also be misleading because the empirical density plot relies on kernel smoothing having a given bandwidth. Thus, the true observations may be more spiky than the plot indicates. The third mode suggests that there are two different values in the observations around 1'100, this is also visible in the QQ plot. Nevertheless, the overall result seems satisfactory. These results (based on 13 estimated parameters) are also summarized in Table 6.2.

We mention a couple of limitations of these results. Firstly, the log-likelihood of this mixture model is unbounded, similarly to Example 6.13 we can precisely fit one degenerate gamma mixture component to an individual observation  $Y_i$  which results in an infinite log-likelihood value. Thus, the found solution corresponds to a local maximum of the log-likelihood function and we should not state AIC values in Table 6.2, see also Remarks 4.28. Secondly, it is crucial to initialize three components to the three modes, if we randomly allocate all claims to 5 bins as initial configuration, the EM algorithm only finds mode  $Z = 3$  but not necessarily the first two modes, at least, in our specifically chosen random initialization this was the case. In fact, the likelihood value of our latter solution was worse than in the first calibration which shows that we ended up in a worse local maximum.

We may be tempted to also estimate the Lomax threshold  $M$  with MLE. In Fig. 6.12 we plot the maximal log-likelihood as a function of  $M$  (if we start the EM algorithm always in the same configuration given in Table 6.1). From this figure a threshold of  $M = 1'600$  seems optimal. Choosing this threshold of  $M = 1'600$  leads to a slightly bigger log-likelihood of  $-199'304$  and a slightly smaller tail parameter of  $\hat{\beta}_5^{(100)} = 1.318$ . However, overall the model is very similar to the one with  $M = 2'000$ . In general, we do *not* recommend to estimate  $M$  with MLE, but this should be treated as a hyper-parameter selected by the modeler. The reason for this recommendation is that this threshold is crucial in deciding for large claims modeling and its estimation from data is, typically, not very robust; we also refer to Remarks 6.15, below.



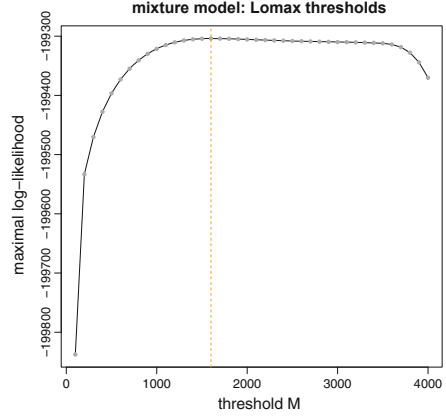
**Fig. 6.11** Mixture null model: (top-lhs) individual estimated gamma components  $f_k(\cdot; \hat{\alpha}_k^{(100)}, \hat{\beta}_k^{(100)})$ ,  $1 \leq k \leq K$ , and Lomax component  $f_5(\cdot; \hat{\beta}_5^{(100)})$ , (top-rhs) estimated mixture density  $\sum_{k=1}^4 \hat{p}_k^{(100)} f_k(\cdot; \hat{\alpha}_k^{(100)}, \hat{\beta}_k^{(100)}) + \hat{p}_5^{(100)} f_5(\cdot; \hat{\beta}_5^{(100)})$ , (bottom-lhs) QQ plot of the estimated model, (bottom-rhs) log-log plot of the estimated model

**Table 6.2** Mixture models for French MTPL claim size modeling

	# Param.	$\ell_Y(\hat{\theta}, \hat{p})$	AIC	$\hat{\mu} = \mathbb{E}_{\hat{\theta}, \hat{p}}[Y]$
Empirical				2'266
Null model ( $M = 2000$ )	13	−199'306	398'637	2'381
Logistic GLM ( $M = 2000$ )	193	−198'404	397'193	2'176

In a next step we enhance the mixture modeling by including feature information  $\mathbf{x}_i$  to explain the responses  $Y_i$ . In view of Fig. 13.17 we have decided to only model the mixture probabilities  $\mathbf{p} = \mathbf{p}(\mathbf{x})$  feature dependent because feature information seems to mainly influence the heights of the peaks. We do not consider features VehPower and VehGas because these features do not seem to contribute, and

**Fig. 6.12** Choice of Lomax threshold  $M$



we do not consider *Density* because of the high co-linearity with *Area*, see Fig. 13.12 (rhs). Thus, we are left with the features *Area*, *VehAge*, *DrivAge*, *BonusMalus*, *VehBrand* and *Region*. Pre-processing of these features is done as in Listing 5.1, except that we keep *Area* categorical. Using these features  $\mathbf{x} \in \mathcal{X} \subset \{1\} \times \mathbb{R}^q$  we choose a logistic categorical GLM for the mixture probabilities

$$\mathbf{x} \mapsto (p_1(\mathbf{x}), \dots, p_{K-1}(\mathbf{x}))^\top = \frac{\exp\{X\boldsymbol{\gamma}\}}{1 + \sum_{l=1}^4 \exp\langle \boldsymbol{\gamma}_l, \mathbf{x} \rangle}, \quad (6.40)$$

that is, we choose  $K = 5$  as reference level, feature matrix  $X \in \mathbb{R}^{(K-1) \times (K-1)(q+1)}$  is defined in (5.71), and with regression parameter  $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1^\top, \dots, \boldsymbol{\gamma}_{K-1}^\top)^\top \in \mathbb{R}^{(K-1)(q+1)}$ ; this regression parameter  $\boldsymbol{\gamma}$  should not be confused with the shape parameters  $\beta_1, \dots, \beta_4$  of the gamma components and the tail parameter  $\beta_5$  of the Lomax component, see (6.39). Note that the notation in this section slightly differs from Sect. 5.7 on the logistic categorical GLM. In this section we consider mixture probabilities  $\mathbf{p}(\mathbf{x}) \in \Delta_{K=5}$  (which corresponds to one-hot encoding), whereas in Sect. 5.7 we model  $(p_1(\mathbf{x}), \dots, p_{K-1}(\mathbf{x}))^\top$  with a categorical GLM (which corresponds to dummy coding), and normalization provides us with  $p_K(\mathbf{x}) = 1 - \sum_{l=1}^{K-1} p_l(\mathbf{x}) \in (0, 1)$ .

This logistic categorical GLM requires that we replace in the M-step the probability estimation (6.31) by Fisher's scoring method for GLMs as outlined in Sect. 5.7.2, but there is a small difference to that section. In the working residuals (5.74) we use dummy coding  $T(Z) \in \{0, 1\}^{K-1}$  of a categorical variable  $Z$ , this now needs to be replaced by the estimated vector  $(\hat{Z}_1(\boldsymbol{\theta}, \mathbf{p}|Y), \dots, \hat{Z}_{K-1}(\boldsymbol{\theta}, \mathbf{p}|Y))^\top \in (0, 1)^{K-1}$  which is used as an estimate for the latent variable  $T(Z)$ . Apart from that everything is done as described in Sect. 5.7.2; in R this can be done with the procedure `multinom` from the package `nnet` [368]. We start the EM algorithm exactly in the final configuration of the

**Table 6.3** Parameter choices in the mixture models: upper part null model, lower part GLM for estimated mixture probabilities  $\hat{\mathbf{p}}(\mathbf{x}_i)$ 

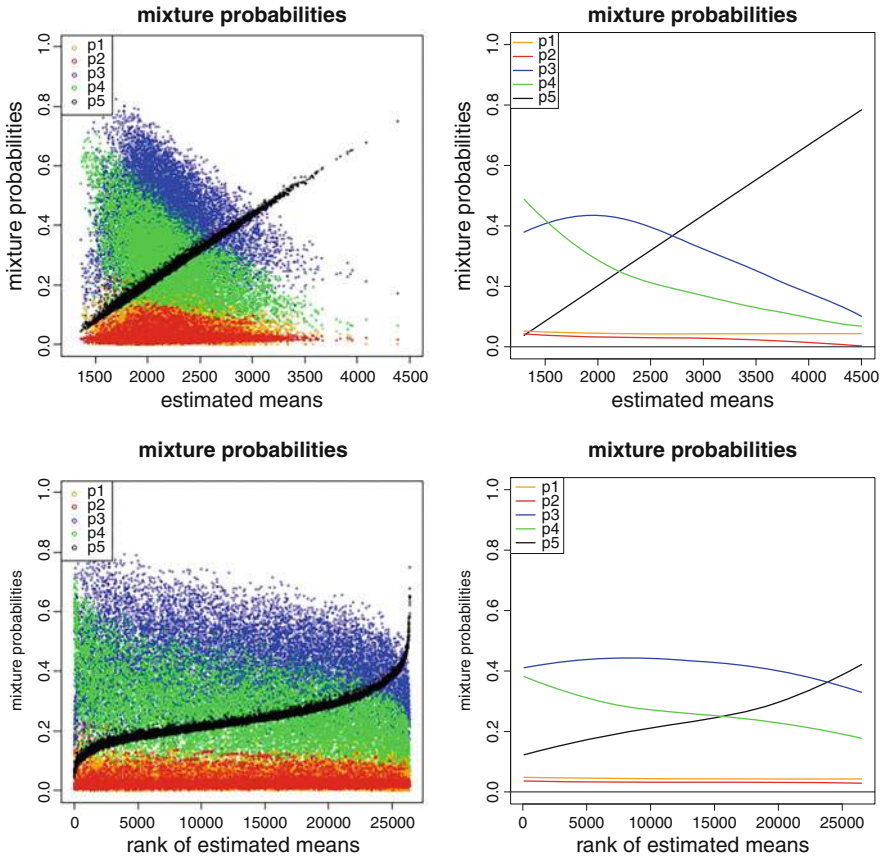
	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
Null: $\hat{p}_k^{(100)}$	0.04	0.03	0.42	0.25	0.26
Null: $\hat{\alpha}_k^{(100)}$	93.05	650.94	1'040.37	1.34	–
Null: $\hat{\beta}_k^{(100)}$	1.207	1.108	0.888	0.001	1.416
Null: $\hat{\mu}_k^{(100)} = \hat{\alpha}_k^{(100)} / \hat{\beta}_k^{(100)}$	77	588	1'172	1'304	–
GLM: average mixture probabilities	0.04	0.03	0.42	0.25	0.26
GLM: $\hat{\alpha}_k^{(100)}$	94.03	597.20	1'043.38	1.28	–
GLM: $\hat{\beta}_k^{(100)}$	1.223	1.019	0.891	0.001	1.365
GLM: $\hat{\mu}_k^{(100)} = \hat{\alpha}_k^{(100)} / \hat{\beta}_k^{(100)}$	77	586	1'172	1'268	–

estimated mixture null model, and we run this algorithm for 20 iterations (which provides convergences).

The resulting parameters are given in the lower part of Table 6.3. We observe that the resulting parameters remain essentially the same, the second mode  $Z = 2$  is a bit less spiky, and the tail parameter is slightly smaller. The summary of this model is given on the last line of Table 6.2. Regression modeling adds another  $4 \cdot 45 = 180$  parameters to the model because we have  $q = 45$  feature components in  $\mathbf{x}$  (different from the intercept component). In view of AIC we give preference to the logistic mixture probability case (though AIC has to be interpreted with care, here, because we do not consider the MLE but rather a local maximum).

Figure 6.13 plots the individual estimated mixture probabilities  $\mathbf{x}_i \mapsto \hat{\mathbf{p}}(\mathbf{x}_i) \in \Delta_5$  over the insurance policies  $1 \leq i \leq n$ ; these plots are inspired by the thesis of Frei [138]. The upper plots consider these probabilities against the estimated claim sizes  $\hat{\mu}(\mathbf{x}_i) = \sum_{k=1}^5 \hat{p}_k(\mathbf{x}_i) \hat{\mu}_k$  and the lower plots against the ranks of  $\hat{\mu}(\mathbf{x}_i)$ , the latter gives a different scaling on the  $x$ -axis because of the heavy-tailedness of the claims. The plots on the left-hand side show all individual policies  $1 \leq i \leq n$ , and the plots on the right-hand side show a quadratic spline fit to these observations. Not surprisingly, we observe that the claim size estimate  $\hat{\mu}(\mathbf{x}_i)$  is mainly driven by the large claims probability  $\hat{p}_5(\mathbf{x}_i)$  describing the Lomax contribution.

In Fig. 6.14 we compare the QQ plots of the mixture null model and the one where we model the mixture probabilities with the logistic categorical GLM. We see that the latter (more complex) model clearly outperforms the more simple one, in fact, this QQ plot looks quite convincing for the French MTPL claim size data. Finally, we perform a Wald test (5.32). We simultaneously treat all parameters that belong to the same feature variable (similar to the ANOVA analysis); for instance, for the 22 Regions the corresponding part of the regression parameter  $\boldsymbol{\gamma}$  contains  $4 \cdot 21 = 84$  components. The resulting  $p$ -values of dropping such components are all close to 0 which says that we should not eliminate one of the feature variables. This closes the example. ■

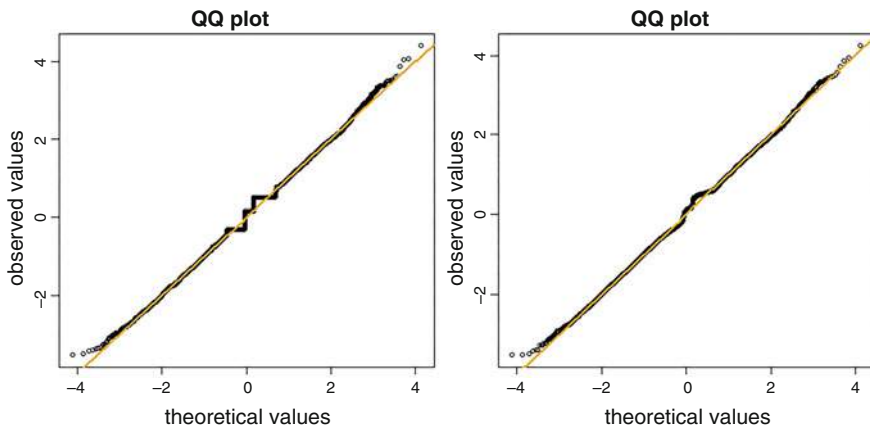


**Fig. 6.13** Mixture probabilities  $x_i \mapsto \hat{p}(x_i)$  on individual policies  $1 \leq i \leq n$ : (top) against the estimated means  $\hat{\mu}(x_i)$  and (bottom) against the ranks of the estimated means  $\hat{\mu}(x_i)$ ; (lhs) over policies  $1 \leq i \leq n$  and (rhs) quadratic spline fit

#### Remarks 6.15

- In Example 6.14 we have chosen a mixture distribution with four gamma components and one Lomax component. The reason for choosing the Lomax component has been two-fold. Firstly, we need a regularly varying tail to model the heavy-tailed property of the data. Secondly, we have preferred the Lomax distribution over the Pareto distribution because this provides us with a continuous density in (6.39). The results in Example 6.14 have been satisfactory. In most practical approaches, however, this approach will not work, even when fixing the threshold  $M$  of the Lomax component. Often, the nature of the data is such that the chosen gamma mixture distribution is not able to fully explain the small data in the body of the distribution, and in that situation the Lomax tail will assist in fitting the small claims. The typical result is that the Lomax part





**Fig. 6.14** QQ plots of the mixture models: (lhs) null model and (rhs) logistic categorical GLM for mixture probabilities

then pays more attention to small claims (through the log-likelihood function of numerous small claims) and the fitting of the tail turns out to be poor (because a few large claims do not sufficiently contribute to the log-likelihood). There are two ways to solve this dilemma. Either one works with composite distributions, see (6.56) below, and one drops the continuity property of the density; this is the approach taken in Fung et al. [148]. Or one fits the Lomax distribution solely to large observations in a first step, and then fixes the parameters of the Lomax distribution during the second step when fitting the full model to all data, this is the approach taken in Frei [138]. Both of these two approaches have been providing good results on real insurance data.

- There is an asymptotic theory for the optimal selection of the number of mixture components, we refer to Khalili–Chen [214] and Khalili [213]. Fung et al. [148] combine this asymptotic theory of mixture component selection with feature selection within these mixture components using LASSO and SCAD regularization.
- In Example 6.14 we have only modeled the mixture probabilities feature dependent, but not the parameters of the gamma mixture components. Introducing regressions for the gamma mixture components needs some care in fitting. For policy independent shape parameters  $\alpha_1, \dots, \alpha_4$ , we can estimate the regression functions for the means of the mixture components without explicitly specifying  $\alpha_k$  because these shape parameters cancel in the score equations. However, these shape parameters will be needed in the E-step, which requires also MLE of  $\alpha_k$ . For more discussion on shape parameter estimation we refer to Sect. 5.3.7 (GLM with constant shape parameter) and Sect. 5.5.4 (double GLM).

## 6.4 Truncated and Censored Data

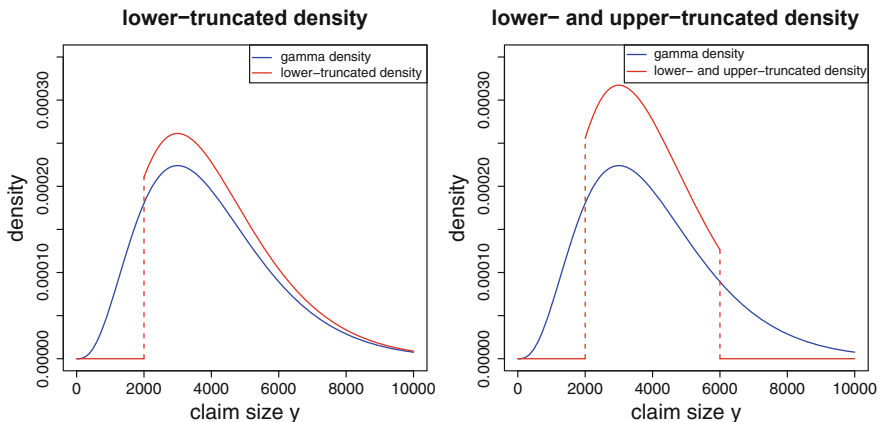
### 6.4.1 Lower-Truncation and Right-Censoring

A common problem in insurance is that we often have truncated or censored observations. Truncation naturally occurs if we sell insurance products that have a deductible  $d > 0$  because in that case only the insurance claim  $(Y - d)_+$  is compensated, and claims below the deductible  $d$  are usually not reported to the insurance company. This case is called *lower-truncation*, because claims below the deductible are not observed. If we lower-truncate an original claim  $Y \sim f(\cdot; \theta)$  with lower-truncation point  $\tau \in \mathbb{R}$  we obtain the density

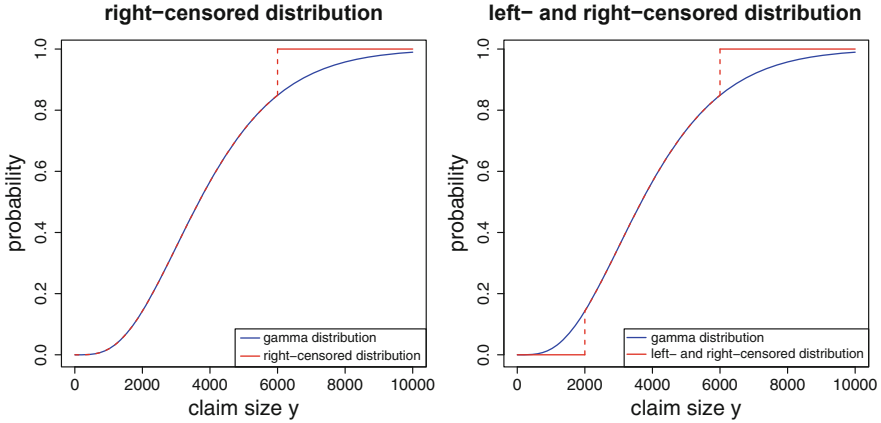
$$f_{(\tau, \infty)}(y; \theta) = \frac{f(y; \theta) \mathbb{1}_{\{y > \tau\}}}{1 - F(\tau, \theta)}, \quad (6.41)$$

if  $F(\cdot; \theta)$  is the distribution function corresponding to the density  $f(\cdot; \theta)$ . The lower-truncated density  $f_{(\tau, \infty)}(y; \theta)$  only considers claims that fall into the interval  $(\tau, \infty)$ . Obviously, we can define upper-truncation completely analogously by considering an interval  $(-\infty, \tau]$  instead. Figure 6.15 (lhs) gives an example of a lower-truncated density, and Fig. 6.15 (rhs) gives an example of a lower- and upper-truncated density.

Censoring occurs by selling insurance products with a maximal cover  $M > 0$  because in that case only the insurance claim  $Y \wedge M = \min\{Y, M\}$  is compensated, and the exact claim size above the maximal cover  $M$  may not be available. This case is called *right-censoring* because the exact claim amount above  $M$  is not known. Right-censoring of an original claim  $Y \sim F(\cdot; \theta)$  with censoring point  $M \in \mathbb{R}$



**Fig. 6.15** (lhs) Lower-truncated gamma density with  $\tau = 2'000$ , and (rhs) lower- and upper-truncated gamma density with truncation points 2'000 and 6'000



**Fig. 6.16** (lhs) Right-censored gamma distribution with  $M = 6'000$ , and (rhs) left- and right-censored gamma distribution with censoring points 2'000 and 6'000

gives the distribution

$$F_{Y \wedge M}(y; \theta) = F(y; \theta) \mathbb{1}_{\{y < M\}} + \mathbb{1}_{\{y \geq M\}},$$

that is, we have a point mass in the censoring point  $M$ . We can define left-censoring analogously by considering the claim  $Y \vee M = \max\{Y, M\}$ . Figure 6.16 (lhs) shows a right-censored gamma distribution with censoring point  $M = 6'000$ , and Fig. 6.16 (rhs) shows a left- and right-censored example with censoring points 2'000 and 6'000.

Often in re-insurance, deductibles (also called retention levels) and maximal covers are combined, for instance, an excess-of-loss (XL) insurance cover of size  $u > 0$  above the retention level  $d > 0$  covers the claim

$$(Y - d)_+ \wedge u = (Y - d) \mathbb{1}_{\{d \leq Y < d+u\}} + u \mathbb{1}_{\{Y \geq d+u\}} = (Y - d)_+ - (Y - (d + u))_+.$$

Obviously, truncation and censoring pose some challenges in regression modeling because at the same time we need to consider the density  $f(\cdot; \theta)$  and the distribution function  $F(\cdot; \theta)$  to estimate a parameter  $\theta$ . Both cases can be understood as missing data problems, with censoring providing the number of claims but not necessarily the exact claim size, and with truncation leaving also the number of claims unknown. These two cases are studied in Fung et al. [147] within the mixture of experts models using a variant of the EM algorithm. We use their techniques within the EDF framework for right-censored or lower-truncated data. This is done in the next sections.

### 6.4.2 Parameter Estimation Under Right-Censoring

Assume we have a fixed censoring point  $M > 0$  that applies to independent observations  $Y_i$  following EDF densities  $f(\cdot; \theta_i, v_i/\varphi)$ ; for simplicity we assume to work with an absolutely continuous EDF in this section. The (incomplete) log-likelihood function of canonical parameters  $\theta = (\theta_i)_{1 \leq i \leq n}$  for observations  $Y \wedge M$  is given by

$$\ell_{Y \wedge M}(\theta) = \sum_{i: Y_i < M} \log f(Y_i; \theta_i, v_i/\varphi) + \sum_{i: Y_i \wedge M = M} \log(1 - F(M; \theta_i, v_i/\varphi)). \quad (6.42)$$

We interpret this as an incomplete data problem because the claim sizes  $Y_i$  above the censoring point  $M$  are not known. The complete log-likelihood is given by

$$\ell_Y(\theta) = \sum_{i=1}^n \log f(Y_i; \theta_i, v_i/\varphi).$$

Similarly to (6.32) we calculate a lower bound to the incomplete log-likelihood. We focus on one component of  $Y$  and drop the lower index  $i$  in  $Y_i$  for this consideration. Firstly, if  $Y \wedge M < M$  we are in the situation of full claim size information and, obviously, we have log-likelihood in that case  $Y < M$

$$\ell_{Y \wedge M}(\theta) = \ell_Y(\theta) = \frac{Y\theta - \kappa(\theta)}{\varphi/v} + a(Y; v/\varphi). \quad (6.43)$$

In the second case  $Y \wedge M = M$  we do not have precise claim size information. In that case we have conditional density of claim  $Y|_{\{Y \wedge M = M\}} = Y|_{\{Y \geq M\}}$  above  $M$

$$f(z|Y \geq M; \theta, v/\varphi) = \frac{f(z; \theta, v/\varphi)\mathbb{1}_{\{z \geq M\}}}{1 - F(M; \theta, v/\varphi)} = \frac{f(z; \theta, v/\varphi)\mathbb{1}_{\{z \geq M\}}}{\exp\{\ell_{Y \wedge M}(\theta)\}}, \quad (6.44)$$

the latter follows because  $Y \wedge M = M$  has the corresponding point mass in censoring point  $M$  (we work with an absolutely continuous EDF here). Choose an arbitrary density  $\pi$  having the same support as  $Y|_{\{Y \geq M\}}$ , and consider a random variable  $Z \sim \pi$ . Using (6.44) and the EDF structure on the last line, we have for  $Y \geq M$

$$\begin{aligned} \ell_{Y \wedge M}(\theta) &= \int \pi(z) \ell_{Y \wedge M}(\theta) dv(z) \\ &= \int \pi(z) \log \left( \frac{f(z; \theta, v/\varphi)/\pi(z)}{f(z|Y \geq M; \theta, v/\varphi)/\pi(z)} \right) dv(z) \\ &= \int \pi(z) \log \left( \frac{f(z; \theta, v/\varphi)}{\pi(z)} \right) dv(z) + D_{\text{KL}}(\pi || f(\cdot|Y \geq M; \theta, v/\varphi)) \end{aligned}$$

$$\begin{aligned}
&\geq \int \pi(z) \log \left( \frac{f(z; \theta, v/\varphi)}{\pi(z)} \right) dv(z) \\
&= \frac{\mathbb{E}_\pi [Z] \theta - \kappa(\theta)}{\varphi/v} + \mathbb{E}_\pi [a(Z; v/\varphi)] - \mathbb{E}_\pi [\log \pi(Z)] \stackrel{\text{def.}}{=} \mathcal{Q}(\theta; \pi).
\end{aligned}$$

This allows us to explore the E-step and the M-step similarly to (6.34) and (6.35).

The **E-step** in the case  $Y \geq M$  for given canonical parameter estimate  $\hat{\theta}^{(t-1)}$  reads as

$$\begin{aligned}
\hat{\pi}^{(t)} &= \arg \max_{\pi} \mathcal{Q}(\hat{\theta}^{(t-1)}; \pi) = \arg \min_{\pi} D_{\text{KL}} \left( \pi \parallel f(\cdot | Y \geq M; \hat{\theta}^{(t-1)}, v/\varphi) \right) \\
&= f(\cdot | Y \geq M; \hat{\theta}^{(t-1)}, v/\varphi).
\end{aligned}$$

This allows us to calculate the estimation of the claim size above  $M$ , i.e., under  $\hat{\pi}^{(t)}$

$$\hat{Y}^{(t)} = \mathbb{E}_{\hat{\pi}^{(t)}} [Z] = \int z f(z | Y \geq M; \hat{\theta}^{(t-1)}, v/\varphi) dv(z). \quad (6.45)$$

Note that this is an estimate of the censored claim  $Y |_{\{Y \geq M\}}$ . This completes the E-step.

The **M-step** considers in the EDF case for censored claim sizes  $Y \geq M$

$$\begin{aligned}
\hat{\theta}^{(t)} &= \arg \max_{\theta} \mathcal{Q}(\theta; \hat{\pi}^{(t)}) = \arg \max_{\theta} \frac{\mathbb{E}_{\hat{\pi}^{(t)}} [Z] \theta - \kappa(\theta)}{\varphi/v} \\
&= \arg \max_{\theta} \ell_{\hat{Y}^{(t)}}(\theta),
\end{aligned} \quad (6.46)$$

the latter uses that the normalizing term  $a(\cdot; v/\varphi)$  is not relevant for the MLE of  $\theta$ . That is, (6.46) describes the regular MLE step under the observation  $\hat{Y}^{(t)}$  in the case of a censored observation  $Y \geq M$ ; and if  $Y < M$  we simply use the log-likelihood (6.43).

---

#### EM algorithm for right-censored data within the EDF

---

(0) Choose an initial parameter  $\hat{\theta}^{(0)} = (\hat{\theta}_i^{(0)})_{1 \leq i \leq n}$ .

(1) Repeat for  $t \geq 1$ :

- **E-step.** Given parameter  $\hat{\theta}^{(t-1)} = (\hat{\theta}_i^{(t-1)})_{1 \leq i \leq n}$ , estimate for the right-censored claims  $Y_i \geq M$  their sizes by, see (6.45),

$$\hat{Y}_i^{(t)} = \int z f(z | Y_i \geq M; \hat{\theta}_i^{(t-1)}, v_i/\varphi) dv(z).$$

This provides us with an estimated observation

$$\widehat{\mathbf{Y}}^{(t)} = \left( Y_i \mathbb{1}_{\{Y_i < M\}} + \widehat{Y}_i^{(t)} \mathbb{1}_{\{Y_i \geq M\}} \right)_{1 \leq i \leq n}^\top.$$

- **M-step.** Calculate the MLE  $\widehat{\boldsymbol{\theta}}^{(t)} = (\widehat{\theta}_i^{(t)})_{1 \leq i \leq n}$  based on observation  $\widehat{\mathbf{Y}}^{(t)}$ , i.e., solve

$$\widehat{\boldsymbol{\theta}}^{(t)} = \arg \max_{\boldsymbol{\theta}} \ell_{\widehat{\mathbf{Y}}^{(t)}}(\boldsymbol{\theta}).$$

Note that the above EM algorithm uses that the log-likelihood  $\ell_Y(\boldsymbol{\theta})$  of the EDF is linear in the observations that interact with parameter  $\boldsymbol{\theta}$ . We revisit the gamma claim size example of Sect. 5.3.7.

*Example 6.16 (Right-Censored Gamma Claim Sizes)* We revisit the gamma claim size GLM introduced in Sect. 5.3.7. The claim sizes are illustrated in Fig. 13.22. In total we have  $n = 656$  observations  $Y_i$ , and they range from 16 SEK to 211'254 SEK. We right-censor this data at  $M = 50'000$ , this results in 545 uncensored observations and 111 censored observations equal to  $M$ . Thus, for the 17% largest claims we assume to not have any knowledge about the exact claim sizes. We use the EM algorithm for right-censored data to fit a GLM to this problem.

In order to calculate the E-step we need to evaluate the conditional expectation (6.45) under the gamma model

$$\begin{aligned} \widehat{Y}^{(t)} &= \int z f(z|Y \geq M; \widehat{\theta}^{(t-1)}, v/\varphi) dv(z) \\ &= \int_M^\infty z \frac{\frac{\beta^\alpha}{\Gamma(\alpha)} z^{\alpha-1} \exp\{-\beta z\}}{1 - \mathcal{G}(\alpha, \beta M)} dz = \frac{\alpha}{\beta} \frac{1 - \mathcal{G}(\alpha + 1, \beta M)}{1 - \mathcal{G}(\alpha, \beta M)}, \end{aligned} \quad (6.47)$$

with shape parameter  $\alpha = v/\varphi$ , scale parameter  $\beta = -\widehat{\theta}^{(t-1)}v/\varphi$ , see (5.45), and scaled incomplete gamma function

$$\mathcal{G}(\alpha, y) = \frac{1}{\Gamma(\alpha)} \int_0^y z^{\alpha-1} \exp\{-z\} dz \in (0, 1) \quad \text{for } y \in (0, \infty). \quad (6.48)$$

Thus, we receive a simple formula that allows us to efficiently calculate the E-step, and the M-step is exactly the gamma GLM explained in Sect. 5.3.7 for the (estimated) data  $\widehat{\mathbf{Y}}^{(t)}$ .

For the modeling we choose exactly the features as used for model Gamma GLM2, this gives  $q + 1 = 7$  regression parameter components and additionally we set for the dispersion parameter  $\widehat{\varphi}^{\text{MLE}} = 1.427$ , this is the MLE in model Gamma

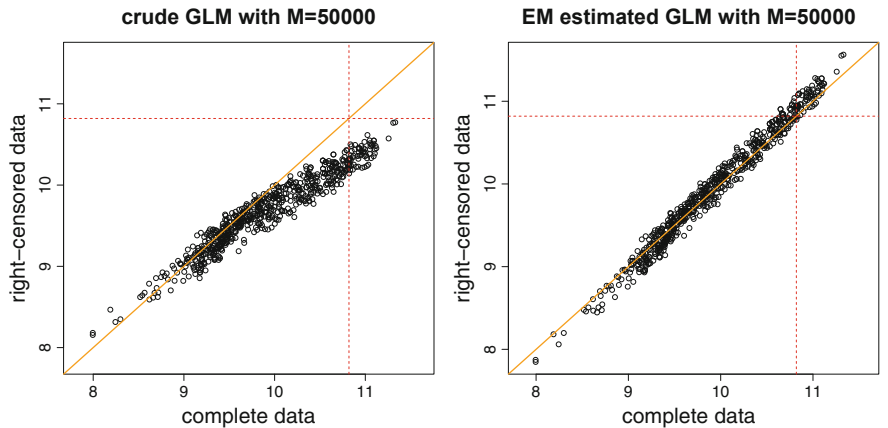
**Table 6.4** Comparison of the complete log-likelihood and the incomplete log-likelihood (right-censoring  $M = 50'000$ ) results

	# Param.	Log-likelihood $\ell_Y(\hat{\theta}^{\text{MLE}}, \hat{\varphi}^{\text{MLE}})$	Dispersion est. $\hat{\varphi}^{\text{MLE}}$	Average amount	Rel. change
Gamma GLM2 (complete data)	7 + 1	-7'129	1.427	25'130	
Crude GLM2 (right-censored)	7 + 1	-7'158		18'068	-28%
EM est. GLM2 (right-censored)	7 + 1	-7'132		26'687	+6%

GLM2. This dispersion parameter we keep fixed in all our models studied in this example. In a first step we simply fit a gamma GLM to the right-censored data  $Y_i \wedge M$ . We call this model ‘crude GLM2’, and it underestimates the empirical claim sizes by 28% because it ignores the fact of having right-censored data.

To initialize the EM algorithm for right-censored data we use the model crude GLM2. We then iterate the algorithm for 15 steps which provides convergence. The results are presented in Table 6.4. We observe that the resulting log-likelihood of the model fitted on the censored data and evaluated on the complete data  $\ell_Y$  (which is available here) is almost the same as for model Gamma GLM2, which has been estimated on the complete data. Moreover, this right-censored EM algorithm fitted model slightly over-estimates the average claim sizes.

Figure 6.17 shows the estimated means  $\hat{\mu}_i$  on an individual claims level. The  $x$ -axis always gives the estimates from the complete log-likelihood model Gamma GLM2. The  $y$ -axis on the left-hand side shows the estimates from the crude GLM and the right-hand side the estimates from the EM algorithm fitted counterpart (fitted on the right-censored data). We observe that the crude model underestimates the claims (being below the diagonal), and the largest estimate lies below  $M = 50'000$



**Fig. 6.17** Comparison of the estimated means  $\hat{\mu}_i$  in model Gamma GLM2 against (lhs) the crude GLM and (rhs) the EM fitted right-censored model; both axis are on the log-scale, the dotted lines shows the censoring point  $\log(M)$

in our example (horizontal dotted line). The EM algorithm fitted model, considering the fact that we have right-censored data, corrects for the censoring, and the resulting estimates resemble the ones from the complete log-likelihood model quite well. In fact, we probably slightly over-estimate under right-censoring, here. Note that all these considerations have been done under an identical dispersion parameter estimate  $\hat{\varphi}^{\text{MLE}}$ . For the complete log-likelihood case, this is not really needed for mean estimation because it cancels in the score equations for mean estimation. However, a reasonable dispersion parameter estimate is crucial for the incomplete case as it enters  $\hat{Y}^{(t)}$  in the E-step, see (6.47), thus, the caveat here is that we need a reasonable dispersion estimate from the right-censored data (which we did not discuss, here, and which requires further research). ■

### 6.4.3 Parameter Estimation Under Lower-Truncation

Compared to censoring we have less information under truncation because not only the claim sizes below the lower-truncation point are unknown, but we also do not know how many claims there are below that truncation point  $\tau$ . Assume we work with responses belonging to the EDF. The incomplete log-likelihood is given by

$$\ell_{Y>\tau}(\theta) = \sum_{i=1}^n \log f(Y_i; \theta_i, v_i/\varphi) - \log(1 - F(\tau; \theta_i, v_i/\varphi)),$$

assuming that  $\mathbf{Y} = (Y_i)_{1 \leq i \leq n} > \tau$  collects all claims above the truncation point  $Y_i > \tau$ , see (6.41). We proceed as in Fung et al. [147] to construct a complete log-likelihood; there are different ways to do so, but this proposal is convenient for parameter estimation. Firstly, we equip each observed claim  $Y_i > \tau$  with an independent count random variable  $K_i \sim p(\cdot; \theta_i, v_i/\varphi)$  that determines the number of claims below the truncation point that correspond to claim  $i$  above the truncation point. Secondly, we assume that these claims are given by independent observations  $Z_{i,1}, \dots, Z_{i,K_i} \leq \tau$ , a.s., with a distribution obtained from an un-truncated version of  $Y_i$ , i.e., we consider the upper-truncated version of  $f(\cdot; \theta_i, v_i/\varphi)$  for  $Z_{i,j}$ . This gives us the complete log-likelihood

$$\begin{aligned} \ell_{(Y,K,Z)}(\theta) = \sum_{i=1}^n & \left( \log \left( \frac{f(Y_i; \theta_i, v_i/\varphi)}{1 - F(\tau; \theta_i, v_i/\varphi)} \right) \right. \\ & \left. + \log p(K_i; \theta_i, v_i/\varphi) + \sum_{j=1}^{K_i} \log \left( \frac{f(Z_{i,j}; \theta_i, v_i/\varphi)}{F(\tau; \theta_i, v_i/\varphi)} \right) \right), \end{aligned} \quad (6.49)$$



with  $\mathbf{K} = (K_i)_{1 \leq i \leq n}$ , and  $\mathbf{Z}$  collects all (latent) claims  $Z_{i,j} \leq \tau$ , an empty sum is set equal to zero. Next, we assume that  $K_i$  is following the geometric distribution

$$\mathbb{P}_{\theta_i}[K_i = k] = p(k; \theta_i, v_i/\varphi) = F(\tau; \theta_i, v_i/\varphi)^k (1 - F(\tau; \theta_i, v_i/\varphi)). \quad (6.50)$$

As emphasized in Fung et al. [147], this complete log-likelihood is an artificial construct that supports parameter estimation of lower-truncated data. It does *not* claim that the true un-truncated data follows this model (6.49) but it provides a distributional extension below the truncation point  $\tau > 0$  that is convenient for parameter estimation. Namely, inserting this geometric distribution assumption into (6.49) gives us complete log-likelihood

$$\ell_{(Y, \mathbf{K}, \mathbf{Z})}(\theta) = \sum_{i=1}^n \left( \log f(Y_i; \theta_i, v_i/\varphi) + \sum_{j=1}^{K_i} \log f(Z_{i,j}; \theta_i, v_i/\varphi) \right). \quad (6.51)$$

Within the EDF this allows us to do the same EM algorithm considerations as above; note that this expression no longer involves the distribution function. We consider one observation  $Y_i > \tau$  and we drop the lower index  $i$ . This gives us complete observation  $(Y, K, Z = (Z_j)_{1 \leq j \leq K})$  and conditional density

$$f(k, z|y; \theta, v/\varphi) = \frac{f(y, k, z; \theta, v/\varphi)}{f(\tau, \infty)(y; \theta, v/\varphi)} = \frac{f(y, k, z; \theta, v/\varphi)}{\exp\{\ell_{Y=y>\tau}(\theta)\}},$$

where  $\ell_{Y>\tau}(\theta)$  is the log-likelihood of the lower-truncated datum  $Y > \tau$ . Choose an arbitrary density  $\pi$  modeling the random vector  $(K, Z)$  below the truncation point  $\tau$ . This gives us for the random vector  $(K, Z) \sim \pi$

$$\begin{aligned} \ell_{Y>\tau}(\theta) &= \int \pi(k, z) \ell_{Y>\tau}(\theta) d\nu(k, z) \\ &= \int \pi(k, z) \log \left( \frac{f(Y, k, z; \theta, v/\varphi)/\pi(k, z)}{f(k, z|Y; \theta, v/\varphi)/\pi(k, z)} \right) d\nu(k, z) \\ &= \int \pi(k, z) \log \left( \frac{f(Y, k, z; \theta, v/\varphi)}{\pi(k, z)} \right) d\nu(k, z) + D_{\text{KL}}(\pi || f(\cdot|Y; \theta, v/\varphi)) \\ &\geq \int \pi(k, z) \log \left( \frac{f(Y, k, z; \theta, v/\varphi)}{\pi(k, z)} \right) d\nu(k, z) \\ &= \mathbb{E}_{\pi} [\ell_{(Y, K, Z)}(\theta) | Y] - \mathbb{E}_{\pi} [\log \pi(K, Z)] \\ &= \log f(Y; \theta, v/\varphi) + \mathbb{E}_{\pi} \left[ \sum_{j=1}^K \log f(Z_j; \theta, v/\varphi) \right] - \mathbb{E}_{\pi} [\log \pi(K, Z)] \\ &\stackrel{\text{def.}}{=} \mathcal{Q}(\theta; \pi), \end{aligned}$$

where the second last identity uses that the log-likelihood (6.51) has a simple form under the geometric distribution chosen for  $K$ ; this is exactly the step where we benefit from this specific choice of the probability extension below the truncation point. There is a subtle point here. Namely,  $\ell_{Y>\tau}(\theta)$  is the log-likelihood of the lower-truncated datum  $Y > \tau$ , whereas  $\log f(Y; \theta, v/\varphi)$  is the log-likelihood not using any lower-truncation.

The **E-step** for given canonical parameter estimate  $\widehat{\theta}^{(t-1)}$  reads as

$$\begin{aligned}\widehat{\pi}^{(t)} &= \arg \max_{\pi} \mathcal{Q}(\widehat{\theta}^{(t-1)}; \pi) = \arg \min_{\pi} D_{\text{KL}}(\pi \parallel f(\cdot|Y; \widehat{\theta}^{(t-1)}, v/\varphi)) \\ &= f(\cdot | Y; \widehat{\theta}^{(t-1)}, v/\varphi) \\ &= p(\cdot; \widehat{\theta}^{(t-1)}, v/\varphi) \prod_{j=1}^{\cdot} \frac{f(\cdot; j; \widehat{\theta}^{(t-1)}, v/\varphi)}{F(\tau; \widehat{\theta}^{(t-1)}, v/\varphi)}.\end{aligned}$$

The latter describes a compound distribution for  $\sum_{j=1}^K Z_j$  with a geometric count random variable  $K$  and independent i.i.d. random variables  $Z_1, Z_2, \dots$ , having upper-truncated densities  $f_{(-\infty, \tau]}(\cdot; \widehat{\theta}^{(t-1)}, v/\varphi)$ . This allows us to calculate the expected compound claim below the truncation point

$$\begin{aligned}\widehat{Y}_{\leq \tau}^{(t)} &= \mathbb{E}_{\widehat{\pi}^{(t)}} \left[ \sum_{j=1}^K Z_j \right] = \mathbb{E}_{\widehat{\pi}^{(t)}} [K] \mathbb{E}_{\widehat{\pi}^{(t)}} [Z_1] \\ &= \frac{F(\tau; \widehat{\theta}^{(t-1)}, v/\varphi)}{1 - F(\tau; \widehat{\theta}^{(t-1)}, v/\varphi)} \int z f_{(-\infty, \tau]}(z; \widehat{\theta}^{(t-1)}, v/\varphi) dv(z).\end{aligned}$$

This completes the E-step.

The **M-step** considers within the EDF

$$\begin{aligned}\widehat{\theta}^{(t)} &= \arg \max_{\theta} \mathcal{Q}(\theta; \widehat{\pi}^{(t)}) \\ &= \arg \max_{\theta} \frac{\left( Y + \mathbb{E}_{\widehat{\pi}^{(t)}} \left[ \sum_{j=1}^K Z_j \right] \right) \theta - (1 + \mathbb{E}_{\widehat{\pi}^{(t)}} [K]) \kappa(\theta)}{\varphi/v} \\ &= \arg \max_{\theta} \frac{v(1 + \mathbb{E}_{\widehat{\pi}^{(t)}} [K])}{\varphi} \left[ \left( \frac{Y + \widehat{Y}_{\leq \tau}^{(t)}}{1 + \mathbb{E}_{\widehat{\pi}^{(t)}} [K]} \right) \theta - \kappa(\theta) \right].\end{aligned}$$

That is, the M-step applies the classical MLE step, we only need to change weights and observations

$$v \mapsto v^{(t)} = v \left( 1 + \mathbb{E}_{\widehat{\pi}^{(t)}} [K] \right) = \frac{v}{1 - F(\tau; \widehat{\theta}^{(t-1)}, v/\varphi)},$$

$$Y \mapsto \widehat{Y}^{(t)} = \frac{Y + \widehat{Y}_{\leq \tau}^{(t)}}{1 + \mathbb{E}_{\widehat{\pi}^{(t)}} [K]} = \frac{Y + \mathbb{E}_{\widehat{\pi}^{(t)}} [K] \mathbb{E}_{\widehat{\pi}^{(t)}} [Z_1]}{1 + \mathbb{E}_{\widehat{\pi}^{(t)}} [K]}.$$

Note that this uses the specific structure of the EDF, in particular, we benefit from linearity here which allows for closed-form solutions.

---

EM algorithm for lower-truncated data within the EDF

---

(0) Choose an initial parameter  $\widehat{\theta}^{(0)} = (\widehat{\theta}_i^{(0)})_{1 \leq i \leq n}$ .

(1) Repeat for  $t \geq 1$ :

- **E-step.** Given parameter  $\widehat{\theta}^{(t-1)} = (\widehat{\theta}_i^{(t-1)})_{1 \leq i \leq n}$ , estimate the number of claims  $K$  and the corresponding claim sizes  $Z_{i,j}$  by

$$\widehat{K}_i^{(t)} = \frac{F(\tau; \widehat{\theta}_i^{(t-1)}, v_i/\varphi)}{1 - F(\tau; \widehat{\theta}_i^{(t-1)}, v_i/\varphi)},$$

$$\widehat{Z}_{i,1}^{(t)} = \int z f_{(-\infty, \tau]}(z; \widehat{\theta}_i^{(t-1)}, v_i/\varphi) dv(z). \quad (6.52)$$

This provides us with estimated weights and observations for  $1 \leq i \leq n$

$$v_i^{(t)} = v_i \left( 1 + \widehat{K}_i^{(t)} \right) \quad \text{and} \quad \widehat{Y}_i^{(t)} = \frac{Y_i + \widehat{K}_i^{(t)} \widehat{Z}_{i,1}^{(t)}}{1 + \widehat{K}_i^{(t)}}.$$

- **M-step.** Calculate the MLE  $\widehat{\theta}^{(t)} = (\widehat{\theta}_i^{(t)})_{1 \leq i \leq n}$  based on observations  $\widehat{\mathbf{Y}}^{(t)} = (\widehat{Y}_i^{(t)})_{1 \leq i \leq n}^\top$  and weights  $\widehat{\mathbf{v}}^{(t)} = (\widehat{v}_i^{(t)})_{1 \leq i \leq n}^\top$ , i.e., solve

$$\widehat{\theta}^{(t)} = \arg \max_{\theta} \ell_{\widehat{\mathbf{Y}}^{(t)}}(\theta; \widehat{\mathbf{v}}^{(t)}/\varphi) = \arg \max_{\theta} \sum_{i=1}^n \log f(\widehat{Y}_i^{(t)}; \theta_i, \widehat{v}_i^{(t)}/\varphi).$$


---

*Remarks 6.17* Essentially, the above algorithm uses that the MLE in the EDF is based on a sufficient statistics of the observations, and in our case this sufficient statistics is  $\widehat{Y}_i^{(t)}$ .

*Example 6.18 (Lower-Truncated Claim Sizes)* We revisit the gamma claim size GLM introduced in Sect. 5.3.7, see also Example 6.16 on right-censored claims. We

choose as lower-truncation point  $\tau = 1'000$ , i.e., we get rid of the very small claims that mainly generate administrative expenses at a rather small claim compensation. We have 70 claims below this truncation point, and there remain  $n = 586$  claims above the truncation point that can be used for model fitting in the lower-truncated case. We use the EM algorithm for lower-truncated data to fit a GLM to this problem.

In order to calculate the E-step we need to evaluate the conditional expectation (6.52) under the gamma model for truncation probability

$$F(\tau; \widehat{\theta}^{(t-1)}, v/\varphi) = \int_0^\tau \frac{\beta^\alpha}{\Gamma(\alpha)} z^{\alpha-1} \exp\{-\beta z\} dz = \mathcal{G}(\alpha, \beta\tau),$$

with shape parameter  $\alpha = v/\varphi$  and scale parameter  $\beta = -\widehat{\theta}^{(t-1)}v/\varphi$ . In complete analogy to (6.47) we have

$$\widehat{Z}_1^{(t)} = \int z f_{(\infty, \tau]}(z; \widehat{\theta}^{(t-1)}, v/\varphi) dv(z) = \frac{\alpha \mathcal{G}(\alpha + 1, \beta\tau)}{\beta \mathcal{G}(\alpha, \beta\tau)}.$$

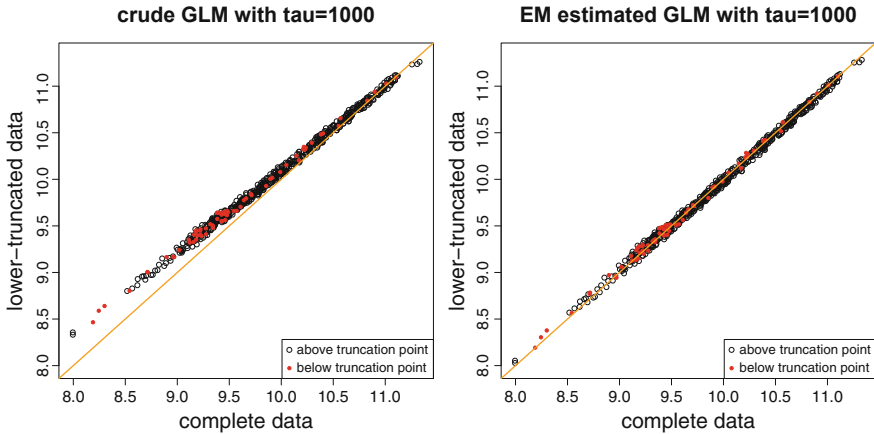
For the modeling we choose again the features as used for model Gamma GLM2, this gives  $q+1 = 7$  regression parameter components and additionally we set for the dispersion parameter  $\widehat{\varphi}^{\text{MLE}} = 1.427$ . This dispersion parameter we keep fixed in all the models studied in this example. In a first step we simply fit a gamma GLM to the lower-truncated data  $Y_i > \tau$ . We call this model ‘crude GLM2’, and it overestimates the true claim sizes because it ignores the fact of having lower-truncated data.

To initialize the EM algorithm for lower-truncated data we use the model crude GLM2. We then iterate the algorithm for 10 steps which provides convergence. The results are presented in Table 6.5. We observe that the resulting log-likelihood fitted on the lower-truncated data and evaluated on the complete data  $\ell_Y$  (which is available here) is the same as for model Gamma GLM2 which has been estimated on the complete data. Moreover, this lower-truncated EM algorithm fitted model slightly under-estimates the average claim sizes.

Figure 6.18 shows the estimated means  $\widehat{\mu}_i$  on an individual claims level. The  $x$ -axis always gives the estimates from the complete log-likelihood model Gamma GLM2. The  $y$ -axis on the left-hand side shows the estimates from the crude GLM and the right-hand side the estimates from the EM algorithm fitted counterpart (fitted on the lower-truncated data). We observe that the crude model overestimates

**Table 6.5** Comparison of the complete log-likelihood and the incomplete log-likelihood (lower-truncation  $\tau = 1'000$ ) results

	# Param.	Log-likelihood $\ell_Y(\widehat{\theta}^{\text{MLE}}, \widehat{\varphi}^{\text{MLE}})$	Dispersion est. $\widehat{\varphi}^{\text{MLE}}$	Average amount	Rel. change
Gamma GLM2 (complete data)	7 + 1	−7'129	1.427	25'130	
Crude GLM2 (lower-truncated)	7 + 1	−7'133		26'879	+7%
EM est. GLM2 (lower-truncated)	7 + 1	−7'129		24'900	−1%



**Fig. 6.18** Comparison of the estimated means  $\hat{\mu}_i$  in model Gamma GLM2 against (lhs) the crude GLM and (rhs) the EM fitted lower-truncated model; both axis are on the log-scale

the claims (being above the orange diagonal), in particular, this applies to claims with lower expected claim amounts. The EM algorithm fitted model, considering the fact that we have lower-truncated data, corrects for the truncation, and the resulting estimates almost completely coincide with the ones from the complete log-likelihood model. Again we remark that we use an identical dispersion parameter estimate  $\hat{\varphi}^{\text{MLE}}$ , and it is an open problem to select a reasonable value from lower-truncated data. ■

*Example 6.19 (Zero-Truncated Claim Counts and the Hurdle Poisson Model)* In Sect. 5.3.6, we have been studying the ZIP model that has assigned an additional probability weight to the event  $\{N = 0\}$  of having zero claims. This model can be understood as a hierarchical model with a latent variable  $Z$  indicating whether we have an excess zero claim or not, see (5.41). In that situation we have a mixture distribution of a Poisson distribution and a degenerate distribution. Fitting in Example 5.25 has been done brute force by using a general purpose optimizer, but we could also use the EM algorithm for mixture distributions.

An alternative way of modeling excess zeros is the hurdle approach which combines a lower-truncated count distribution with a point mass in zero. For the Poisson case this reads as, see (5.42),

$$f_{\text{hurdle Poisson}}(k; \lambda, v, \pi_0) = \begin{cases} \pi_0 & \text{for } k = 0, \\ (1 - \pi_0) \frac{e^{-v\lambda} (v\lambda)^k}{1 - e^{-v\lambda}} & \text{for } k \in \mathbb{N}, \end{cases} \quad (6.53)$$

for  $\pi_0 \in (0, 1)$  and  $\lambda, v > 0$ . If we ignore any observation  $\{N = 0\}$  we obtain a lower-truncated Poisson model, also called zero-truncated Poisson (ZTP) model. This ZTP model can be fitted with the EM algorithm for lower-truncated data. In the following we only consider insurance policies  $i$  with  $N_i > 0$ . The log-likelihood of

the ZTP model  $N > 0$  is given by (we consider one single component only and drop the lower index in the notation)

$$\theta \mapsto \ell_{N>0}(\theta) = N\theta - ve^\theta - \log(N!) + N \log(v) - \log(1 - e^{-ve^\theta}), \quad (6.54)$$

with exposure  $v > 0$  and canonical parameter  $\theta \in \Theta = \mathbb{R}$  such that  $\lambda = \exp\{\theta\}$ . The ZTP model provides for the random variable  $K$  the following geometric distribution (for the number of claims below the truncation point), see (6.50),

$$\mathbb{P}_\theta[K = k] = \mathbb{P}_\theta[N = 0]^k \mathbb{P}_\theta[N > 0] = e^{-kve^\theta} (1 - e^{-ve^\theta}).$$

In view of (6.51), this gives us complete log-likelihood (note that  $Z_j = 0$  for all  $j$ )

$$\begin{aligned} \ell_{(N,K,Z)}(\theta) &= N\theta - ve^\theta - \log(N!) + N \log(v) + \sum_{j=1}^K (Z_j\theta - ve^\theta - \log(Z_j!) + Z_j \log(v)) \\ &= N\theta - (1 + K)ve^\theta - \log(N!) + N \log(v). \end{aligned}$$

We can now directly apply a simplified version of the EM algorithm for lower-truncated data. For the E-step we have, given parameter  $\hat{\theta}^{(t-1)}$ ,

$$\hat{K}^{(t)} = \frac{\mathbb{P}_{\hat{\theta}^{(t-1)}}[N = 0]}{1 - \mathbb{P}_{\hat{\theta}^{(t-1)}}[N = 0]} = \frac{e^{-v e^{\hat{\theta}^{(t-1)}}}}{1 - e^{-v e^{\hat{\theta}^{(t-1)}}}} \quad \text{and} \quad \hat{Z}_1^{(t)} = 0.$$

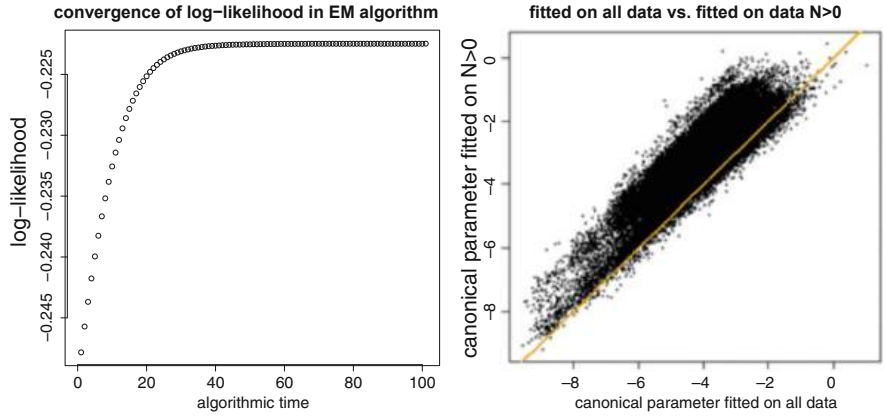
This provides us with the estimated weights and observations (set  $Y = N/v$ )

$$v^{(t)} = v \left(1 + \hat{K}^{(t)}\right) = \frac{v}{1 - e^{-v e^{\hat{\theta}^{(t-1)}}}} \quad \text{and} \quad \hat{Y}^{(t)} = \frac{Y}{1 + \hat{K}^{(t)}} = \frac{N}{v^{(t)}}. \quad (6.55)$$

Thus, the EM algorithm iterates Poisson MLEs, and the E-Step modifies the weights  $v^{(t)}$  in each step of the loop correspondingly. We remark that the ZTP model has an EF representation which allows one to directly estimate the corresponding parameters without using the EM algorithm, see Remark 6.20, below.

We revisit the French MTPL claim frequency data, and, in particular, we use model Poisson GLM3 as a benchmark, we refer to Tables 5.5 and 5.10. The feature engineering is done exactly as in model Poisson GLM3. We then select only the insurance policies from the learning data  $\mathcal{L}$  that have suffered at least one claim, i.e.,  $N_i > 0$ . These are  $m = 22'434$  out of  $n = 610'206$  insurance policies. Thus, we only consider  $m/n = 3.68\%$  of all insurance policies, and we fit the lower-truncated log-likelihood (ZTP model) to this data

$$\ell_{N>0}(\beta) = \sum_{i=1}^m N_i \theta_i - v_i e^{\theta_i} - \log(N_i!) + N_i \log(v_i) - \log(1 - e^{-v_i e^{\theta_i}}),$$



**Fig. 6.19** (lhs) Convergence of the EM algorithm for the lower-truncated data in the Poisson hurdle case; (rhs) canonical parameters of the Poisson GLMs fitted on all data  $\mathcal{L}$  vs. fitted only on policies with  $N_i > 0$

**Table 6.6** Run times, number of parameters, AICs, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model and the Poisson, negative-binomial, ZIP and hurdle Poisson GLMs

	Run time	# Param.	AIC	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	199'506	25.213	25.445	7.36%
Poisson GLM3	15 s	50	192'716	24.084	24.102	7.36%
NB GLM3 $\hat{\alpha}_{NB}^{MLE} = 1.810$	85 s	51	192'113	20.722	20.674	7.38%
ZIP GLM3 (null $\pi_0$ )	270 s	51	192'393	–	–	7.37%
Hurdle Poisson GLM3	300 s	100	<b>191'851</b>	–	–	7.39%

where  $1 \leq i \leq m$  runs over all insurance policies with at least one claim and where the canonical parameter  $\theta_i$  is given by the linear predictor  $\theta_i = \langle \boldsymbol{\beta}, \mathbf{x}_i \rangle$ . We fit this model using the EM algorithm for lower-truncated data. In each loop this requires that the offset  $o_i^{(t)} = \log(v_i^{(t)})$  is adjusted according to (6.55); for the discussion of offsets we refer to Sect. 5.2.3. Convergence of the EM algorithm is achieved after roughly 75 iterations, see Fig. 6.19 (lhs).

In our first analysis we do not consider the Poisson hurdle model, but we simply consider model Poisson GLM3. However, this Poisson model with regression parameter  $\boldsymbol{\beta}$  is fitted only on the data  $N_i > 0$  (exactly using the results of the EM algorithm for lower-truncated data  $N_i > 0$ ). The resulting predictive model is presented in Table 6.7. We observe that model Poisson GLM3 that is only fitted on the data  $N_i > 0$  is clearly not competitive, i.e., we cannot simply extrapolate this estimated model to  $\{N_i = 0\}$ . This extrapolation results in a Poisson GLM that has a much too large average frequency of 15.11%, see last column of Table 6.7; this bias can clearly be seen in Fig. 6.19 (rhs) where we compare the two fits. From this we conclude that either the Poisson model assumption in general does not

**Table 6.7** Number of parameters, in-sample and out-of-sample deviance losses on all data (units are in  $10^{-2}$ ), out-of-sample lower-truncated log-likelihood  $\ell_{N>0}$  and in-sample average frequency of the Poisson null model and model Poisson GLM3 fitted on all data  $\mathcal{L}$  and fitted on the data  $N_i > 0$  only

	# Param.	In-sample	Out-of-sample		Aver.
		Loss on $\mathcal{L}$	Loss on $\mathcal{T}$	$\ell_{N>0}$	freq.
Poisson null	1	25.213	25.445	–	7.36%
Poisson GLM3 fitted on all data	50	24.084	24.102	–0.2278	7.36%
Poisson GLM3 fitted on $N_i > 0$	50	28.064	28.211	–0.2195	15.11%

match the data, or that we have excess zeros (which do not influence the estimation procedure if we only consider the policies with at least one claim). Let us compare the lower-truncated log-likelihood  $\ell_{N>0}$  out-of-sample only on the policies with at least one claim (ZTP model). We observe that the EM fitted model provides a better description of the data, as we have a bigger log-likelihood than the model fitted on all data  $\mathcal{L}$  (i.e.  $-0.2195$  vs.  $-0.2278$  for the ZTP log-likelihood). Thus, the lower-truncated fitting procedure finds a better model on  $\{N_i > 0\}$  when only fitted on these lower-truncated claim counts.

This analysis concludes that we need to fit the full hurdle Poisson model (6.53). That is, we cannot simply extrapolate the model fitted on the ZTP log-likelihood  $\ell_{N>0}$  because, typically,  $\pi_0(\mathbf{x}_i) \neq \exp\{-v_i e^{\langle \tilde{\beta}, \mathbf{x}_i \rangle}\}$ , the latter coming from the Poisson GLM with regression parameter  $\tilde{\beta}$ . We model the zero claim probability  $\pi_0(\mathbf{x}_i)$  by the logistic Bernoulli GLM indicating whether we have claims or not. We set up the logistic GLM for  $p(\mathbf{x}_i) = 1 - \pi_0(\mathbf{x}_i)$  of describing the indicator  $Y_i = \mathbb{1}_{\{N_i > 0\}}$  of having claims. The difficulty compared to the Poisson model is that we cannot easily integrate the time exposure  $v_i$  as a pro rata temporis variable like in the Poisson case. We therefore make the following considerations. The canonical link in the logistic Bernoulli GLM is the logit function  $p \mapsto \text{logit}(p) = \log(p/(1-p)) = \log(p) - \log(1-p)$  for  $p \in (0, 1)$ . Typically, in our application,  $p \ll 1$  is fairly small because claims are rare events. This implies  $\log(p/(1-p)) \approx \log(p)$ , i.e., the logit link behaves similarly to the log-link for small default probabilities  $p$ . This motivates to integrate the logged exposures  $\log v_i$  as offsets into the logistic probabilities. That is, we make the following model assumption

$$(\mathbf{x}, v) \mapsto \text{logit}(p(\mathbf{x}_i, v_i)) = \log(v_i) + \langle \tilde{\beta}, \mathbf{x}_i \rangle,$$

with offset  $o_i = \log(v_i)$  and regression parameter  $\tilde{\beta} \in \mathbb{R}^{q+1}$ . We fit this model using the R command `glm` using `family=binomial()`. The results then allow us to define the estimated hurdle Poisson model by, recall  $p(\mathbf{x}_i, v_i) = 1 - \pi_0(\mathbf{x}_i, v_i)$ ,

$$f_{\text{hurdle Poisson}}(k; \mathbf{x}_i, v_i) = \begin{cases} 1 - p(\mathbf{x}_i, v_i) = (1 + \exp\{\log(v_i) + \langle \tilde{\beta}, \mathbf{x}_i \rangle\})^{-1} & \text{for } k = 0, \\ \frac{p(\mathbf{x}_i, v_i)}{1 - e^{-\mu(\mathbf{x}_i, v_i)}} e^{-\mu(\mathbf{x}_i, v_i)} \frac{\mu(\mathbf{x}_i, v_i)^k}{k!} & \text{for } k \in \mathbb{N}, \end{cases}$$



**Table 6.8** Contingency table of the observed numbers of policies against predicted numbers of policies with given claim counts `ClaimNb` (in-sample)

	Numbers of claims <code>ClaimNb</code>					
	0	1	2	3	4	5
Observed number of policies	587'772	21'198	1'174	57	4	1
Poisson predicted number of policies	587'325	22'064	779	34	3	0.3
NB predicted number of policies	587'902	20'982	1'200	100	15	4
ZIP predicted number of policies	587'829	21'094	1'191	79	9	4
Hurdle Poisson predicted number of policies	587'772	21'119	1'233	76	6	1

where  $\tilde{\beta} \in \mathbb{R}^{q+1}$  is the regression parameter from the logistic Bernoulli GLM, and where  $\mu(\mathbf{x}_i, v_i) = v_i \exp\langle \beta, \mathbf{x}_i \rangle$  is the Poisson GLM estimated with the EM algorithm on the lower-truncated data  $N_i > 0$  (ZTP model). The results are presented in Table 6.6.

Table 6.6 compares the hurdle Poisson model to the approaches studied in Table 5.10. Firstly, fitting the hurdle Poisson model is more time intensive, the EM algorithm takes some time and we need to fit the Bernoulli logistic GLM which is of a similar complexity as fitting model Poisson GLM3. The results in terms of AIC look convincing. The hurdle Poisson model provides an excellent model for the indicator of having a claim (here it outperforms model ZIP GLM3). It also tries to optimally fit a ZTP model to all insurance policies having at least one claim. This can also be seen from Table 6.8 which determines the expected number of policies that suffer the different numbers of claims.

We close this example by concluding that the hurdle Poisson model provides the best description, at the price of using more parameters. The ZIP model could be lifted to a similar level, however, we consider fitting the hurdle approach to be more convenient, see also Remark 6.20, below. In particular, feature engineering seems simpler in the hurdle approach because the different effects are clearly separated, whereas in the ZIP approach it is more difficult to suitably model the excess zeros, see also Listing 5.10. This closes this example. ■

*Remark 6.20* In (6.54) we have been considering the ZTP model for different exposures  $v > 0$ . If we set these exposures to  $v = 1$ , we obtain the ZTP log-likelihood

$$\ell_{N>0}(\theta) = N\theta - \left(e^\theta + \log(1 - e^{-e^\theta})\right) - \log(N!).$$

Note that this describes a single-parameter linear EF with cumulant function

$$\kappa(\theta) = e^\theta + \log(1 - e^{-e^\theta}),$$

for canonical parameter in the effective domain  $\theta \in \Theta = \mathbb{R}$ . The mean of this EF model is given by

$$\mu = \mathbb{E}_\theta[N] = \kappa'(\theta) = \frac{e^\theta}{1 - e^{-e^\theta}} = \frac{\lambda}{1 - e^{-\lambda}},$$

where we set  $\lambda = e^\theta$ . The variance is given by

$$\text{Var}_\theta(N) = \kappa''(\theta) = \mu \left( \frac{e^\lambda - (1 + \lambda)}{e^\lambda - 1} \right) = \mu (1 - \mu e^{-\lambda}) > 0.$$

Note that the term in brackets is positive but less than one. The latter implies that the ZTP model has under-dispersion. Alternatively to the EM algorithm, we can also directly fit a GLM to this ZTP model. The only difficulty is that we need to appropriately integrate the time exposures. The original Poisson model suggests that if we choose the canonical parameter being equal to the linear predictor, we should integrate the logged exposures as offsets into the linear predictors. Along these lines, if we choose the canonical link  $h = (\kappa')^{-1}$  of the ZTP model, we receive that the canonical parameter  $\theta$  is equal to the linear predictor  $\langle \boldsymbol{\beta}, \mathbf{x} \rangle$ , and we can directly integrate the logged exposures as offsets into the canonical parameters, see (5.25). This then allows us to directly fit this ZTP model with exposures using Fisher's scoring method. In this case of a concave log-likelihood function, the result will be identical to the solution of the EM algorithm found in Example 6.19, and, in fact, this direct approach is more straightforward and more time-efficient. Similar considerations can be done for other hurdle models.

#### 6.4.4 Composite Models

In Sect. 6.3.1 we have promoted to mix distributions in cases where the data cannot be modeled by a single EDF distribution. Alternatively, one can also consider to compose densities which leads to so-called *composite models* (also called splicing models). This idea has been introduced to the actuarial literature by Cooray–Ananda [81] and Scollnik [332]. Assume we have two absolutely continuous densities  $f^{(i)}(\cdot; \theta_i)$  with corresponding distribution functions  $F^{(i)}(\cdot; \theta_i)$ ,  $i = 1, 2$ . These two densities can easily be composed at a splicing value  $\tau$  and with weight  $p \in (0, 1)$  by considering the following composite density

$$f(y; p, \theta_1, \theta_2) = p \frac{f^{(1)}(y; \theta_1) \mathbb{1}_{\{y \leq \tau\}}}{F^{(1)}(\tau; \theta_1)} + (1 - p) \frac{f^{(2)}(y; \theta_2) \mathbb{1}_{\{y > \tau\}}}{1 - F^{(2)}(\tau; \theta_2)}, \quad (6.56)$$

supposed that both denominators are non-zero. In this notation we treat splicing value  $\tau$  as a hyper-parameter that is chosen by the modeler, and is not estimated

from data. In view of (6.41) we can rewrite this in terms for lower- and upper-truncated densities

$$f(y; p, \theta_1, \theta_2) = p f_{(-\infty, \tau]}^{(1)}(y; \theta_1) + (1 - p) f_{(\tau, \infty)}^{(2)}(y; \theta_2).$$

In this notation, we see that a composite model can also be interpreted as a mixture model with mixture probability  $p \in (0, 1)$  and mixing densities  $f_{(-\infty, \tau]}^{(1)}$  and  $f_{(\tau, \infty)}^{(2)}$  having disjoint supports  $(-\infty, \tau]$  and  $(\tau, \infty)$ , respectively.

These disjoint supports allow for simpler MLE, i.e., we do not need to rely on the ‘EM algorithm for mixture distributions’ to fit this model. The log-likelihood of  $Y \sim f(y; p, \theta_1, \theta_2)$  is given by

$$\begin{aligned} \ell_Y(p, \theta_1, \theta_2) = & \left( \log(p) + \log f_{(-\infty, \tau]}^{(1)}(Y; \theta_1) \right) \mathbb{1}_{\{Y \leq \tau\}} \\ & + \left( \log(1 - p) + \log f_{(\tau, \infty)}^{(2)}(Y; \theta_2) \right) \mathbb{1}_{\{Y > \tau\}}. \end{aligned}$$

This shows that the log-likelihood nicely decouples in the composite case and all parameters can directly be estimated with MLE: parameter  $\theta_1$  uses all observations smaller or equal to  $\tau$ , parameter  $\theta_2$  uses all observations bigger than  $\tau$ , and  $p$  is estimated by the proportions of claims below and above the splicing point  $\tau$ . This holds for a null model as well as for a GLM approach for  $\theta_1$ ,  $\theta_2$  and  $p$ .

Nevertheless, the EM algorithm may still be used for parameter estimation, namely, truncation may ask for the ‘EM algorithm for truncated data’. Alternatively, we could also use the ‘EM algorithm for censored data’ to estimate the truncated densities, because we have knowledge of the number of claims above and below the splicing point  $\tau$ , thus, we could right- or left-censor these claims. The latter may lead to more stability in the estimation procedure since we use more information in parameter estimation, i.e., the two truncated densities will not be independent because they simultaneously consider all claim counts (but not identical claim sizes due to censoring).

For composite models one sometimes requires more regularity in the densities, we may, e.g., require continuity in the density in the splicing point which provides mixture probability

$$p = \frac{f^{(2)}(\tau; \theta_2)F^{(1)}(\tau; \theta_1)}{f^{(1)}(\tau; \theta_1)(1 - F^{(2)}(\tau; \theta_2)) + f^{(2)}(\tau; \theta_2)F^{(1)}(\tau; \theta_1)}.$$

This reduces the number of parameters to be estimated but complicates the score equations. If we require a differential condition in  $\tau$  we receive requirement

$$p = \frac{f_y^{(2)}(\tau; \theta_2)F^{(1)}(\tau; \theta_1)}{f_y^{(1)}(\tau; \theta_1)(1 - F^{(2)}(\tau; \theta_2)) + f_y^{(2)}(\tau; \theta_2)F^{(1)}(\tau; \theta_1)},$$

where  $f_y^{(i)}(y; \theta_i)$  denotes the first derivative w.r.t.  $y$ . Together with the continuity this provides requirement for having differentiability in  $\tau$

$$\frac{f^{(2)}(\tau; \theta_2)}{f^{(1)}(\tau; \theta_1)} = \frac{f_y^{(2)}(\tau; \theta_2)}{f_y^{(1)}(\tau; \theta_1)}.$$

Again this reduces the degrees of freedom in parameter estimation but complicates the score equations. We refrain from giving an example and close this section; we will consider a deep composite regression model in Sect. 11.3.2, below, where we replace the fixed splicing point by a quantile for a fixed quantile level.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.



# Chapter 7

## Deep Learning



In the sequel, we introduce deep learning models. In this chapter these deep learning models will be based on fully-connected feed-forward neural networks. We present these networks as an extension of GLMs. These networks perform feature engineering themselves. We discuss how networks achieve this, and we explain how networks are used for predictive modeling. There is a vastly growing literature on deep learning with networks, the classical reference is the book of Goodfellow et al. [166], but also the numerous tutorials around the open-source deep learning libraries TensorFlow [2], Keras [77] or PyTorch [296] give an excellent overview of the state-of-the-art in this field.

### 7.1 Deep Learning and Representation Learning

In Chap. 5 on GLMs, we have been modeling the mean structure of the responses  $Y$ , given features  $\mathbf{x}$ , by the following regression function, see (5.6),

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = \mathbb{E}_{\theta(\mathbf{x})}[Y] = g^{-1} \langle \boldsymbol{\beta}, \mathbf{x} \rangle. \quad (7.1)$$

The crucial assumption has been that the regression function (7.1) provides a reasonable functional description of the expected value  $\mathbb{E}_{\theta(\mathbf{x})}[Y]$  of datum  $(Y, \mathbf{x})$ . As described in Sect. 5.2.2, this typically requires *manual feature engineering* of  $\mathbf{x}$ , bringing feature information into the right structural form.

In contrast to manual feature engineering, deep learning aims at performing an *automated feature engineering* within the statistical model by massaging information through different transformations. Deep learning uses a finite sequence of functions  $(\mathbf{z}^{(m)})_{1 \leq m \leq d}$ , called *layers*,

$$\mathbf{z}^{(m)} : \{1\} \times \mathbb{R}^{q_{m-1}} \rightarrow \{1\} \times \mathbb{R}^{q_m},$$

of (fixed) dimensions  $q_m \in \mathbb{N}$ ,  $1 \leq m \leq d$ , and initialization  $q_0 = q$  being the dimension of the (raw) feature information  $\mathbf{x} \in \mathcal{X} \subset \{1\} \times \mathbb{R}^q$ . Each of these layers presents a new *representation of the features*, that is, after layer  $m$  we have a  $q_m$ -dimensional representation of the raw feature  $\mathbf{x} \in \mathcal{X}$

$$\mathbf{z}^{(m:1)}(\mathbf{x}) \stackrel{\text{def.}}{=} \left( \mathbf{z}^{(m)} \circ \dots \circ \mathbf{z}^{(1)} \right)(\mathbf{x}) \in \{1\} \times \mathbb{R}^{q_m}. \quad (7.2)$$

Note that the first component is always identically equal to 1. For this reason we call the representation  $\mathbf{z}^{(m:1)}(\mathbf{x}) \in \{1\} \times \mathbb{R}^{q_m}$  of  $\mathbf{x}$  to be  $q_m$ -dimensional.

*Deep learning* now assumes that we have  $d \in \mathbb{N}$  appropriate transformations (layers)  $\mathbf{z}^{(m)}$ ,  $1 \leq m \leq d$ , such that  $\mathbf{z}^{(d:1)}(\mathbf{x})$  provides a suitable  $q_d$ -dimensional representation of the raw feature  $\mathbf{x} \in \mathcal{X}$ , that then enters a GLM

$$\mu(\mathbf{x}) = \mathbb{E}_{\theta(\mathbf{x})} [Y] = g^{-1} \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle, \quad (7.3)$$

with link function  $g : \mathcal{M} \rightarrow \mathbb{R}$  and regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q_d+1}$ . This regression architecture is called a *feed-forward network* of *depth*  $d \in \mathbb{N}$  because information  $\mathbf{x}$  is processed in a directed acyclic (feed-forward) path through the  $d$  layers  $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(d)}$  before entering the final GLM.

Each layer  $\mathbf{z}^{(m)}$  involves parameters. Successful deep learning simultaneously fits these parameters as well as the regression parameter  $\boldsymbol{\beta}$  to the available learning data  $\mathcal{L}$  so that we obtain an optimal predictive model on the test data  $\mathcal{T}$ . That is, the learned model should optimally generalize to unseen data, we refer to Chap. 4 on predictive modeling. Thus, the process of optimal representation learning is also part of the model fitting procedure. In contrast to GLMs, the resulting log-likelihood functions are non-concave in their parameters because, typically, each layer involves non-linear transformations. This makes model fitting a challenge. State-of-the-art model fitting in deep learning uses variants of the gradient descent algorithm which we have already met in Sect. 6.2.4.

*Remark 7.1* Representation learning  $\mathbf{x} \mapsto \mathbf{z}^{(d:1)}(\mathbf{x})$  is closely related to Mercer's kernel [272]. If we have a portfolio with features  $\mathbf{x}_1, \dots, \mathbf{x}_n$ , we obtain a Mercer's kernel by considering the matrix

$$\mathbf{K} = \left( K(\mathbf{x}_i, \mathbf{x}_j) \right)_{1 \leq i, j \leq n} = \left( \left\langle \mathbf{z}^{(d:1)}(\mathbf{x}_i), \mathbf{z}^{(d:1)}(\mathbf{x}_j) \right\rangle \right)_{1 \leq i, j \leq n} \in \mathbb{R}^{n \times n}. \quad (7.4)$$

In many regression problems it can be shown that one can equivalently work with the design matrix  $\mathbf{Z} = (\mathbf{z}^{(d:1)}(\mathbf{x}_1), \dots, \mathbf{z}^{(d:1)}(\mathbf{x}_n))^\top \in \mathbb{R}^{n \times (q_d+1)}$  or with

Mercer’s kernel  $\mathbf{K} \in \mathbb{R}^{n \times n}$ . Mercer’s kernel does not require the full knowledge of the learned representations  $\mathbf{z}^{(d:1)}(\mathbf{x}_i)$ , but it suffices to know the discrepancies between  $\mathbf{z}^{(d:1)}(\mathbf{x}_i)$  and  $\mathbf{z}^{(d:1)}(\mathbf{x}_j)$  measured by the scalar products  $K(\mathbf{x}_i, \mathbf{x}_j)$ . This is also closely related to the cosine similarity in word embeddings, see (10.11). This approach then results in replacing the search for an optimal representation learning by a search of the optimal Mercer’s kernel for the given data; this is called the kernel trick in machine learning.

7.2 Generic Feed-Forward Neural Networks

Feed-forward neural (FN) networks use special layers  $\mathbf{z}^{(m)}$  in (7.2)–(7.3), whose components are called *neurons*. This is discussed and studied in detail in this section.

7.2.1 Construction of Feed-Forward Neural Networks

FN networks are regression functions of type (7.3) where each neuron  $z_j^{(m)}$ ,  $1 \leq j \leq q_m$ , of the layers  $\mathbf{z}^{(m)} = (1, z_1^{(m)}, \dots, z_{q_m}^{(m)})^\top$ ,  $1 \leq m \leq d$ , has the structure of a GLM; the first component  $z_0^{(m)} = 1$  always plays the role of the intercept and does not need any modeling.

A first important choice is the *activation function*  $\phi : \mathbb{R} \rightarrow \mathbb{R}$  which plays the role of the inverse link function  $g^{-1}$ . To perform non-linear representation learning, this activation function should be non-linear, too. The most popular choices of activation functions are listed in Table 7.1.

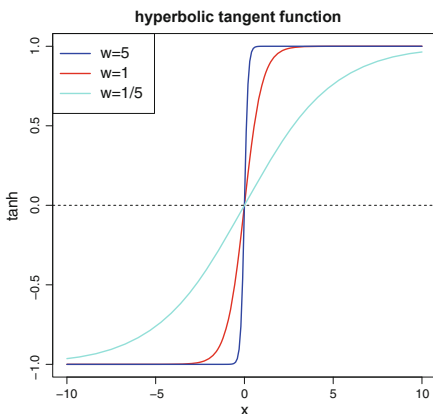
The first three examples in Table 7.1 are smooth functions with simple derivatives, see the last column of Table 7.1. Having simple derivatives is an advantage in gradient descent algorithms for model fitting. The derivative of the ReLU activation function for  $x \neq 0$  is given by the step function activation, and in 0 one typically considers a sub-gradient. We briefly comment on these activation functions.

**Table 7.1** Popular choices of non-linear activation functions and their derivatives; the last two examples are not strictly monotone

	Activation function	Derivative
Sigmoid (logistic) activation	$\phi(x) = (1 + e^{-x})^{-1}$	$\phi' = \phi(1 - \phi)$
Hyperbolic tangent activation	$\phi(x) = \tanh(x)$	$\phi' = 1 - \phi^2$
Exponential activation	$\phi(x) = \exp(x)$	$\phi' = \phi$
Step function activation	$\phi(x) = \mathbb{1}_{\{x \geq 0\}}$	
Rectified linear unit (ReLU) activation	$\phi(x) = x \mathbb{1}_{\{x \geq 0\}}$	

**Fig. 7.1** Hyperbolic tangent activation function

$x \mapsto \tanh(wx) \in (-1, 1)$  for (fixed) weights  $w \in \{1/5, 1, 5\}$  and  $x \in (-10, 10)$



- We are mainly going to use the *hyperbolic tangent activation* function

$$x \mapsto \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = 2 \left( 1 + e^{-2x} \right)^{-1} - 1 \in (-1, 1).$$

Figure 7.1 illustrates the hyperbolic tangent activation function.

The hyperbolic tangent activation function is anti-symmetric w.r.t. the origin with range  $(-1, 1)$ . This anti-symmetry and boundedness is an advantage in fitting deep FN network architectures. For this reason we usually prefer the hyperbolic tangent over other activation functions.

- The *sigmoid activation* function corresponds to the logistic function that was used in the Bernoulli and the categorical EFs, see Sects. 2.1.2 and 5.7. The sigmoid activation function can be obtained from the hyperbolic tangent activation function by setting  $\phi(x) = (\tanh(x/2) + 1)/2$ .
- The *step function activation* is not really used in applications. However, it allows for nice interpretations, and it links FN networks to the theory of regression and classification trees (CARTs); see Breiman et al. [54] for CARTs.
- The *exponential activation* function is a nice differentiable choice whenever the range should be one-sided bounded.
- The *ReLU activation* function is also called hinge function or ramp function. This is the preferred choice in the machine learning community. However, typically, we will not use it because in our experience it is less robust in fitting compared to the hyperbolic tangent activation function. This may be for two reasons, firstly, the ReLU activation is unbounded, and secondly, it is identically equal to zero for  $x < 0$ , which implies that there is no sensitivity in negative choices of  $x$ .



A FN layer with activation function  $\phi$  is a mapping

$$\mathbf{z}^{(m)} : \{1\} \times \mathbb{R}^{q_{m-1}} \rightarrow \{1\} \times \mathbb{R}^{q_m} \quad (7.5)$$

$$\mathbf{z} \mapsto \mathbf{z}^{(m)}(\mathbf{z}) = \left(1, z_1^{(m)}(\mathbf{z}), \dots, z_{q_m}^{(m)}(\mathbf{z})\right)^\top,$$

having neurons for  $1 \leq j \leq q_m$

$$z_j^{(m)}(\mathbf{z}) = \phi(\mathbf{w}_j^{(m)}, \mathbf{z}) = \phi\left(\sum_{l=0}^{q_{m-1}} w_{l,j}^{(m)} z_l\right), \quad (7.6)$$

with given network weights  $\mathbf{w}_j^{(m)} = (w_{l,j}^{(m)})_{0 \leq l \leq q_{m-1}} \in \mathbb{R}^{q_{m-1}+1}$ .

**Interpretation** Every neuron  $\mathbf{z} \mapsto z_j^{(m)}(\mathbf{z})$  describes a GLM regression function with link function  $\phi^{-1}$  and regression parameter  $\mathbf{w}_j^{(m)} \in \mathbb{R}^{q_{m-1}+1}$  for features  $\mathbf{z} \in \{1\} \times \mathbb{R}^{q_{m-1}}$ . These GLM regression functions can be interpreted as data compression, i.e., in each neuron the  $q_{m-1}$ -dimensional feature  $\mathbf{z}$  is projected to a real number  $\langle \mathbf{w}_j^{(m)}, \mathbf{z} \rangle \in \mathbb{R}$  which is then (non-linearly) activated by  $\phi$ . Since this leads to a substantial loss of information, we perform this procedure of data compression  $q_m$  times in FN layer  $\mathbf{z}^{(m)}$ , so that each neuron in  $(z_j^{(m)}(\mathbf{z}))_{1 \leq j \leq q_m}$  represents a different projection of input  $\mathbf{z}$ . Choosing suitable weights  $\mathbf{w}_j^{(m)}$  will allow us to extract the crucial feature information from  $\mathbf{z}$  to receive good explanatory variables for the regression task at hand.

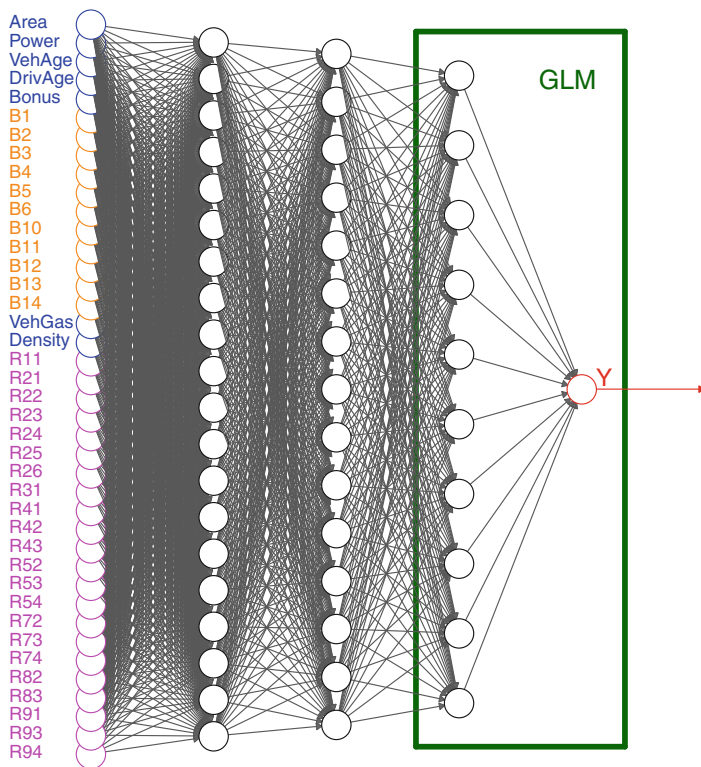
A FN network of depth  $d \in \mathbb{N}$  is obtained by composing  $d$  FN layers  $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(d)}$  to receive the mapping

$$\mathbf{z}^{(d:1)} : \{1\} \times \mathbb{R}^{q_0=q} \rightarrow \{1\} \times \mathbb{R}^{q_d} \quad (7.7)$$

$$\mathbf{x} \mapsto \mathbf{z}^{(d:1)}(\mathbf{x}) = \left(\mathbf{z}^{(d)} \circ \dots \circ \mathbf{z}^{(1)}\right)(\mathbf{x}).$$

Choosing a strictly monotone and smooth link function  $g$  and a regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q_d+1}$  we receive the FN network regression function

$$\mathbf{x} \in \mathcal{X} \mapsto \mu(\mathbf{x}) = g^{-1}(\boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x})). \quad (7.8)$$



**Fig. 7.2** FN network of depth  $d = 3$ , with number of neurons  $(q_1, q_2, q_3) = (20, 15, 10)$  and input dimension  $q_0 = 40$ . This gives us a network parameter  $\boldsymbol{\vartheta} \in \mathbb{R}^r$  of dimension  $r = 1'306$

This FN network regression function (7.8) has a *network parameter*  $\boldsymbol{\vartheta} = (\mathbf{w}_1^{(1)}, \dots, \mathbf{w}_{q_d}^{(d)}, \boldsymbol{\beta})^\top \in \mathbb{R}^r$  of dimension

$$r = \sum_{m=1}^d q_m(q_{m-1} + 1) + (q_d + 1).$$

In Fig. 7.2 we illustrate a FN network of depth  $d = 3$ , FN layers of dimensions  $(q_1, q_2, q_3) = (20, 15, 10)$  and input dimension  $q_0 = 40$ .<sup>1</sup> This gives us a network parameter  $\boldsymbol{\vartheta} \in \mathbb{R}^r$  of dimension  $r = 1'306$ . On the left-hand side we have the raw features  $\mathbf{x} \in \mathcal{X} \subset \{1\} \times \mathbb{R}^{q_0}$ , these are processed through the three FN layers, where the black circles illustrate the neurons  $z_j^{(m)}$ . The third FN layer  $z^{(3)}$  has dimension

<sup>1</sup> Figures 7.2 and 7.9 are similar to Figure 1 in [122], and all FN network plots have been created with modified versions of the plot functions of the R package `neuralnet` [144].

$q_3 = 10$  providing the learned representation  $\mathbf{z}^{(3:1)}(\mathbf{x}) \in \{1\} \times \mathbb{R}^{q_3}$  of  $\mathbf{x}$ . This is used in the final GLM step (7.8) in the green box of Fig. 7.2.

### Remarks 7.2

- One distinguishes between FN networks of depth  $d = 1$ , called *shallow networks*, and FN networks of depth  $d > 1$ , called *deep networks*. In this sense, deep learning means that we learn suitable feature representations through multiple FN layers  $d > 1$ . We come back to this in Sect. 7.2.2, below. Remark that some people would only call a network deep if  $d \gg 1$ , here  $d > 1$  will be chosen for the definition of deep (which is also a precise definition).
- There are two ways of receiving a GLM. If we have a (trivial) FN network of depth  $d = 0$ , this naturally corresponds to a GLM, see Fig. 7.2. In that case, one works with the original features  $\mathbf{x} \in \mathcal{X}$  in (7.8). The second way of receiving a GLM is given by choosing the identity function as activation function  $\phi(x) = x$ . This implies that  $\mathbf{x} \mapsto \mathbf{z}^{(d:1)}(\mathbf{x}) = A\mathbf{x}$  is a linear function for some matrix  $A \in \mathbb{R}^{(q_d+1) \times (q+1)}$  and, henceforth, we receive a GLM.
- Under the above interpretation of the representation learning structure (7.7), we may also give a different intuition for the FN layers. Typically, we expect that the first FN layers decompose feature information  $\mathbf{x}$  into bits and pieces, which are then recomposed in a suitable way for the prediction task. In this sense, we typically choose a larger dimension for the early FN layers otherwise we may lose too much information already from the very beginning.
- The neural network introduced in (7.7) is called FN network because the signals propagate from one layer to the next (directed acyclic graph). If the network has loops it is called a *recurrent neural* (RN) network. RN networks have been applied very successfully in image and speech recognition, for instance, long short-term memory (LSTM) networks are very useful for time-series analysis. We study RN networks in Chap. 8, below. A third type of neural networks are *convolutional neural* (CN) networks which are very successfully applied to image recognition because they are capable to detect similar structures at different places in images, i.e., CN networks learn local representations. We will discuss CN network architectures in Chap. 9, below.
- The generic FN network architecture (7.8) can be complemented by drop-out layers, normalization layers, skip connections, embedding layers, etc. Such layers are special purpose layers, for instance, taking care of over-fitting. We introduce and discuss these below.
- The regression function (7.8) has a one-dimensional output for regression modeling. Of course, categorical classification can be done completely analogously by choosing a link function  $g$  suitable for classification, see Sect. 5.7. A similar approach also works if, for instance, we want to model simultaneously the mean and the dispersion of the data with a two-dimensional output function  $g^{-1}$ .

### 7.2.2 Universality Theorems

The use of FN networks for representation learning is motivated by the so-called *universality theorems* which say that any compactly supported continuous (regression) function can be approximated arbitrarily well by a suitably large FN network. As such, we can understand the FN network framework as an approximation tool which, of course, is useful far beyond statistical modeling. In Chapter 12 we give some proofs of selected universality statements to illustrate the flavor of such results. In particular, Cybenko [86], Hornik et al. [192], Hornik [191], Leshno et al. [247], Park–Sandberg [293, 294], Petrushev [302] and Isenbeck–Rüschendorf [198] have shown (under mild conditions on the activation function) that shallow FN networks can approximate any compactly supported continuous function arbitrarily well (in supremum norm or in  $L^2$ -norm), if we allow for an arbitrary number of neurons  $q_1 \in \mathbb{N}$  in the single FN layer. Roughly speaking, such a result for shallow FN networks holds true if and only if the chosen activation function is non-polynomial, see Leshno et al. [247]. Such results are proved either by algebraic methods of Stone–Weierstrass type or by Wiener–Tauberian denseness type arguments. Moreover, approximation results are studied in Barron [25, 26], Yukich et al. [399], Makavoz [262], Pinkus [303] and Döhler–Rüschendorf [108].

The above stated universality theorems say that shallow FN networks are sufficient from an approximation point of view. Nevertheless, we will mainly use deep (multiple layers) FN networks, below. These have better convergence properties to given function classes because they more easily promote interactions in feature components compared to shallow ones. Such questions have been studied, e.g., by Elbrächter et al. [120], Kidger–Lyons [215], Lu et al. [260] or Cheridito et al. [75]. For instance, Elbrächter et al. [120] compare finite-depth wide networks to finite-width deep networks (under the choice of the ReLU activation function), and they conclude that for many function classes deep networks lead to exponential approximation rates, whereas shallow networks only provide polynomial approximation rates at the same number of network parameters. This motivates to consider sufficiently deep FN networks for representation learning because these typically have a better approximation capacity compared to shallow ones.

We motivate this by two simple examples. For this motivation we use the step function activation  $\phi(x) = \mathbb{1}_{\{x \geq 0\}} \in \{0, 1\}$ . If we have the step function activation, each neuron partitions  $\mathbb{R}^{q_{m-1}}$  along a hyperplane, i.e.,

$$z \mapsto z_j^{(m)}(z) = \phi\langle \mathbf{w}_j^{(m)}, \mathbf{z} \rangle = \mathbb{1}_{\left\{ \sum_{l=1}^{q_{m-1}} w_{l,j}^{(m)} z_l \geq -w_{0,j}^{(m)} \right\}} \in \{0, 1\}. \quad (7.9)$$

For a shallow FN network we can study the question of the maximal complexity of the resulting partition of the feature space  $\mathcal{X} \subset \{1\} \times \mathbb{R}^{q_0}$  when considering  $q_1$

neurons (7.9) in the single FN layer  $z^{(1)}$ . Zaslavsky [400] proved that  $q_1$  hyperplanes can partition the Euclidean space  $\mathbb{R}^{q_0}$  in at most

$$\sum_{j=0}^{\min\{q_0, q_1\}} \binom{q_1}{j} \quad \text{disjoint sets.} \quad (7.10)$$

This number (7.10) can be seen as a maximal upper complexity bound for shallow FN networks with step function activation. It grows exponentially for  $q_1 \leq q_0$ , and it slows down to a polynomial growth for  $q_1 > q_0$ . Thus, the complexity of shallow FN networks grows comparably slow as the width  $q_1$  of the network exceeds  $q_0$ , and therefore we often need a huge network to receive a good approximation.

This result (7.10) should be contrasted to Theorem 4 in Montúfar et al. [280] who give a lower bound on the complexity of regression functions of deep FN networks (under the ReLU activation function). Assume  $q_m \geq q_0$  for all  $1 \leq m \leq d$ . The maximal complexity is bounded below by

$$\left( \prod_{m=1}^{d-1} \left\lfloor \frac{q_m}{q_0} \right\rfloor^{q_0} \right) \sum_{j=0}^{q_0} \binom{q_d}{j} \quad \text{disjoint linear regions.} \quad (7.11)$$

If we choose as an example a FN network with fixed width  $q_m = 4$  for all  $m \geq 1$  and an input of dimension  $q_0 = 2$ , we receive from (7.11) a lower bound of

$$4^{d-1} \left( \binom{4}{0} + \binom{4}{1} + \binom{4}{2} \right) = \frac{11}{4} \exp\{d \log(4)\}.$$

Thus, we have an exponential growth in depth  $d \rightarrow \infty$ . This contrasts the polynomial complexity growth (7.10) of shallow FN networks.

*Example 7.3 (Shallow vs. Deep Networks: Partitions)* We give a second more explicit example that compares shallow and deep FN networks. Choose  $q_0 = 2$  and assume we want to describe a regression function

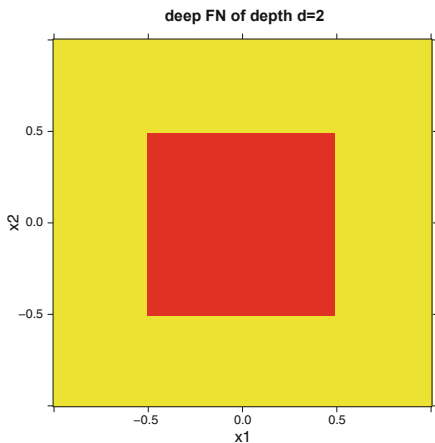
$$\mu : \mathbb{R}^2 \rightarrow \mathbb{R}, \quad \mathbf{x} \mapsto \mu(\mathbf{x}).$$

If we think of a tool box of basis functions to build regression function  $\mu$  we may want to choose indicator functions  $\mathbf{x} \mapsto \chi_A(\mathbf{x}) \in \{0, 1\}$  for arbitrary rectangles  $A = [x_1^-, x_1^+] \times [x_2^-, x_2^+] \subset \mathbb{R}^2$ . We show that we can easily construct such indicator functions  $\chi_A(\mathbf{x})$  for given rectangles  $A \subset \mathbb{R}^2$  with FN networks of depth  $d = 2$ , but not with shallow FN networks.

For illustrative purposes, we fix a square  $A = [-1/2, 1/2] \times [-1/2, 1/2] \subset \mathbb{R}^2$ , and we want to construct  $\chi_A(\mathbf{x})$  with a network of depth  $d = 2$ . This indicator function  $\chi_A$  is illustrated in Fig. 7.3.

**Fig. 7.3** Indicator function

$\chi_A(\mathbf{x})$  for square  
 $A = [-1/2, 1/2) \times$   
 $[-1/2, 1/2) \subset \mathbb{R}^2$



We choose the step function activation for  $\phi$  and a first FN layer with  $q_1 = 4$  neurons

$$\begin{aligned} \mathbf{x} \mapsto \mathbf{z}^{(1)}(\mathbf{x}) &= \left(1, z_1^{(1)}(\mathbf{x}), \dots, z_4^{(1)}(\mathbf{x})\right)^\top \\ &= \left(1, \mathbb{1}_{\{x_1 \geq -1/2\}}, \mathbb{1}_{\{x_2 \geq -1/2\}}, \mathbb{1}_{\{x_1 \geq 1/2\}}, \mathbb{1}_{\{x_2 \geq 1/2\}}\right)^\top \in \{1\} \times \{0, 1\}^4. \end{aligned}$$

This FN layer has a network parameter, see also (7.9),

$$\left(\mathbf{w}_1^{(1)}, \dots, \mathbf{w}_4^{(1)}\right) = \left(\begin{pmatrix} 1/2 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1/2 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} -1/2 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1/2 \\ 0 \\ 1 \end{pmatrix}\right), \quad (7.12)$$

having dimension  $q_1(q_0 + 1) = 12$ . For the second FN layer with  $q_2 = 4$  neurons we choose the step function activation and

$$\begin{aligned} \mathbf{z} \mapsto \mathbf{z}^{(2)}(\mathbf{z}) &= \left(1, z_1^{(2)}(\mathbf{z}), \dots, z_4^{(2)}(\mathbf{z})\right)^\top \\ &= \left(1, \mathbb{1}_{\{z_1+z_2 \geq 3/2\}}, \mathbb{1}_{\{z_2+z_3 \geq 3/2\}}, \mathbb{1}_{\{z_1+z_4 \geq 3/2\}}, \mathbb{1}_{\{z_3+z_4 \geq 3/2\}}\right)^\top. \end{aligned}$$

This FN layer has a network parameter

$$\left(\mathbf{w}_1^{(2)}, \dots, \mathbf{w}_4^{(2)}\right) = \left(\begin{pmatrix} -3/2 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -3/2 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -3/2 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} -3/2 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}\right),$$

having dimension  $q_2(q_1 + 1) = 20$ . For the output layer we choose the identity link  $g(x) = x$ , and the regression parameter  $\beta = (0, 1, -1, -1, 1)^\top \in \mathbb{R}^5$ . As a result, we obtain

$$\chi_A(x) = \langle \beta, z^{(2:1)}(x) \rangle. \quad (7.13)$$

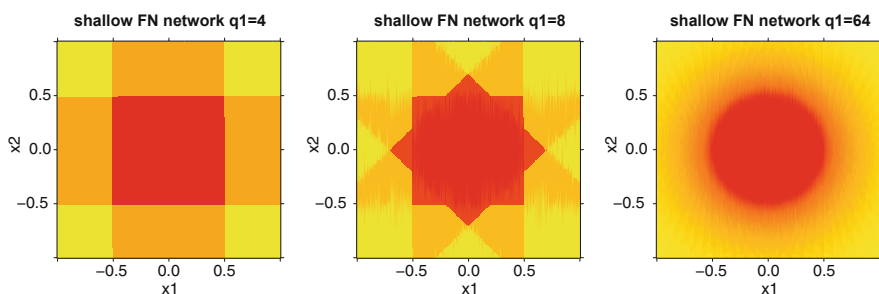
That is, this network of depth  $d = 2$ , number of neurons  $(q_1, q_2) = (4, 4)$ , step function activation and identity link can perfectly replicate the indicator function for the square  $A = [-1/2, 1/2) \times [-1/2, 1/2)$ , see Fig. 7.3. This network has  $r = 37$  parameters.

We now consider a shallow FN network with  $q_1$  neurons. The resulting regression function with identity link is given by

$$\begin{aligned} x \mapsto \langle \beta, z^{(1:1)}(x) \rangle &= \langle \beta, (1, z_1^{(1)}(x), \dots, z_{q_1}^{(1)}(x))^\top \rangle \\ &= \left\langle \beta, \left( 1, \mathbb{1}_{\{\langle w_1^{(1)}, x \rangle \geq 0\}}, \dots, \mathbb{1}_{\{\langle w_{q_1}^{(1)}, x \rangle \geq 0\}} \right)^\top \right\rangle, \end{aligned}$$

where we have used the step function activation  $\phi(x) = \mathbb{1}_{\{x \geq 0\}}$ . As in (7.9), each of these neurons leads to a partition of the space  $\mathbb{R}^2$  with a straight line. Importantly these straight lines go *across* the *entire feature space*, and, therefore, we cannot exactly construct the indicator function of Fig. 7.3 with a shallow FN network. This can nicely be seen in Fig. 7.4 (lhs), where we consider a shallow FN network with  $q_1 = 4$  neurons, weights (7.12), and  $\beta = (0, 1/2, 1/2, -1/2, -1/2)^\top$ .

However, from the universality theorems we know that shallow FN networks can approximate any compactly supported (continuous) function arbitrarily well for sufficiently large  $q_1$ . In this example we can introduce additional neurons and let the resulting hyperplanes rotate around the origin. In Fig. 7.4 (middle, rhs) we show this for  $q_1 = 8$  and  $q_1 = 64$  neurons. We observe that this allows us to approximate a circle, see Fig. 7.4 (rhs), and having circles of different sizes at



**Fig. 7.4** Shallow FN networks with  $q_1 = 4$  (lhs),  $q_1 = 8$  (middle) and  $q_1 = 64$  (rhs)

different locations will allow us to approximate the square  $A$  considered above. However, of course, this is a much less efficient way compared to the deep FN network (7.13).

Intuitively speaking, shallow FN networks act like additions where we add more and more separating hyperplanes for  $q_1 \rightarrow \infty$  (*superposition of basis functions*). In contrast to that, going deep allows us to not only use additions but to also use multiplications (*composition of basis functions*). This is the reason, why we can easily construct the indicator function  $\chi_A$  in the deep case (where we multiply zero's along the boundary of  $A$ ), but not in the shallow case. ■

### 7.2.3 Gradient Descent Methods

We describe gradient descent methods in this section. These are used to fit FN networks. Gradient descent algorithms have already been used in Sect. 6.2.4 for fitting LASSO regularized regression models. We will give the full methodological part here, without relying on Sect. 6.2.4.

#### Plain Vanilla Gradient Descent Algorithm

Assume we have independent instances  $(Y_i, \mathbf{x}_i)$ ,  $1 \leq i \leq n$ , that follow the same member of the EDF. We choose a regression function

$$\mathbf{x}_i \mapsto \mu(\mathbf{x}_i) = \mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i) = \mathbb{E}_{\theta(\mathbf{x}_i)}[Y_i] = g^{-1} \left( \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}_i) \right),$$

for a strictly monotone and smooth link function  $g$ , and a FN network  $\mathbf{z}^{(d:1)}$  with network parameter  $\boldsymbol{\vartheta} \in \mathbb{R}^r$ . We assume that the chosen activation function  $\phi$  is differentiable. We highlight in the notation that the mean functional  $\mu_{\boldsymbol{\vartheta}}(\cdot)$  depends on the network parameter  $\boldsymbol{\vartheta}$ . The canonical parameter of the response  $Y_i$  is given by  $\theta(\mathbf{x}_i) = h(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i)) \in \boldsymbol{\Theta}$ , where  $h = (\kappa')^{-1}$  is the canonical link and  $\kappa$  the cumulant function of the chosen member of the EDF. This gives us (under constant dispersion  $\varphi$ ) the log-likelihood function, for given data  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$ ,

$$\boldsymbol{\vartheta} \mapsto \ell_Y(\boldsymbol{\vartheta}) = \sum_{i=1}^n \frac{v_i}{\varphi} \left[ Y_i h(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i)) - \kappa(h(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i))) \right] + a(Y_i; v_i/\varphi).$$

The deviance loss function in this model is given by, see (4.9) and (4.8),

$$\mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta}) = \frac{2}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \left( Y_i h(Y_i) - \kappa(h(Y_i)) - Y_i h(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i)) + \kappa(h(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i))) \right) \geq 0. \quad (7.14)$$



The MLE of  $\boldsymbol{\vartheta}$  is found by either maximizing the log-likelihood function or by minimizing the deviance loss function in  $\boldsymbol{\vartheta}$ . This problem cannot be solved in general because of complexity. Typically, the deviance loss function is non-convex in  $\boldsymbol{\vartheta}$  and it may have many local minimums. This is one of the reasons, why we are less ambitious here, and why we just try to find a network parameter  $\hat{\boldsymbol{\vartheta}}$  which provides a “small” deviance loss  $\mathfrak{D}(\mathbf{Y}, \hat{\boldsymbol{\vartheta}})$  for the given data  $\mathbf{Y}$ . We discuss this further, below, in fact, this is a crucial point in FN network fitting that is related to *in-sample over-fitting* and, therefore, this point will require a broader discussion.

For the moment, we just try to find a network parameter  $\hat{\boldsymbol{\vartheta}}$  that provides a small deviance loss  $\mathfrak{D}(\mathbf{Y}, \hat{\boldsymbol{\vartheta}})$  for the given data  $\mathbf{Y}$ . Gradient descent algorithms suggest that we try to step-wise locally improve our current position by changing the network parameter into the direction of the maximal local decrease of the deviance loss function. By assumption, our deviance loss function is differentiable in  $\boldsymbol{\vartheta}$ . This allows us to consider the following first order Taylor expansion in  $\boldsymbol{\vartheta}$

$$\mathfrak{D}(\mathbf{Y}, \tilde{\boldsymbol{\vartheta}}) = \mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta}) + \nabla_{\boldsymbol{\vartheta}} \mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta})^\top (\tilde{\boldsymbol{\vartheta}} - \boldsymbol{\vartheta}) + o(\|\tilde{\boldsymbol{\vartheta}} - \boldsymbol{\vartheta}\|_2) \quad \text{as } \|\tilde{\boldsymbol{\vartheta}} - \boldsymbol{\vartheta}\|_2 \rightarrow 0.$$

This shows that the locally optimal change  $\boldsymbol{\vartheta} \mapsto \tilde{\boldsymbol{\vartheta}}$  points into the opposite direction of the gradient of the deviance loss function. This motivates the following gradient descent step.

Assume that at algorithmic time  $t \in \mathbb{N}$  we have a network parameter  $\boldsymbol{\vartheta}^{(t)} \in \mathbb{R}^r$ . Choose a suitable *learning rate*  $\varrho_{t+1} > 0$ , and consider the gradient descent update

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}). \quad (7.15)$$

This gradient descent update gives us the new (smaller) deviance loss at algorithmic time  $t + 1$

$$\mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t+1)}) = \mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}) - \varrho_{t+1} \left\| \nabla_{\boldsymbol{\vartheta}} \mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}) \right\|_2^2 + o(\varrho_{t+1}) \quad \text{for } \varrho_{t+1} \downarrow 0.$$

Under suitably tempered learning rates  $(\varrho_t)_{t \geq 1}$ , this algorithm will converge to a local minimum of the deviance loss function as  $t \rightarrow \infty$  (supposed that we do not get trapped in a saddlepoint).

*Remarks 7.4* We give a couple of (preliminary) remarks on the gradient descent algorithm (7.15), more explanation, further derivations, and variants of the gradient descent algorithm will be discussed below.

- In the applications we will *early stop* the gradient descent algorithm before reaching a local minimum (to prevent from over-fitting). This is going to be discussed in the next paragraphs.
- Fine-tuning the learning rate  $(\varrho_t)_t$  is important, in particular, there is a trade-off between smaller and bigger learning rates: they need to be sufficiently small so that the first order Taylor expansion is still a valid approximation, and they should be sufficiently big otherwise the convergence of the algorithm will be very slow because it needs many iterations.
- The gradient descent algorithm is a first order algorithm, and one is tempted to study higher order approximations, e.g., leading to the Newton–Raphson algorithm. Unfortunately, higher order derivatives are computationally not feasible if the size  $n$  of the data  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  and the dimension  $r$  of the network parameter  $\boldsymbol{\vartheta}$  are large. In fact, even the calculation of the first order derivatives may be challenging and, therefore, stochastic gradient descent methods are considered below. Nevertheless, it is beneficial to have a notion of a second order term. Momentum-based methods originate from approximating the second order terms, these will be studied in (7.19)–(7.20), below.
- The gradient descent step (7.15) solves an unconstrained local optimization. Similarly to (6.15)–(6.16) we could change the gradient descent algorithm to a constraint optimization problem, e.g., involving a LASSO constraint that can be solved with the generalized projection operator (6.17).

### Gradient Calculation via Back-Propagation

Fast gradient descent algorithms essentially rely on fast gradient calculations of the deviance loss function. Under the EDF setup we have gradient w.r.t.  $\boldsymbol{\vartheta}$

$$\begin{aligned}\nabla_{\boldsymbol{\vartheta}} \mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta}) &= \frac{2}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \left( \mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i) - Y_i \right) h'(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i)) \nabla_{\boldsymbol{\vartheta}} \mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i) \\ &= \frac{2}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \frac{\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i) - Y_i}{V(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i))} \frac{1}{g'(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i))} \nabla_{\boldsymbol{\vartheta}} \left\langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}_i) \right\rangle,\end{aligned}\quad (7.16)$$

where the last step uses the variance function  $V(\cdot)$  of the chosen EDF, we also refer to (5.9). The main difficulty is the calculation of the gradient

$$\nabla_{\boldsymbol{\vartheta}} \left\langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \right\rangle = \nabla_{\boldsymbol{\vartheta}} \left\langle \boldsymbol{\beta}, \left( \mathbf{z}^{(d)} \circ \dots \circ \mathbf{z}^{(1)} \right)(\mathbf{x}) \right\rangle,$$

w.r.t. the network parameter  $\boldsymbol{\vartheta} = (\mathbf{w}_1^{(1)}, \dots, \mathbf{w}_{q_d}^{(d)}, \boldsymbol{\beta})^\top \in \mathbb{R}^r$ , and where each FN layer  $\mathbf{z}^{(m)}$  involves the weights  $\mathcal{W}^{(m)} = (\mathbf{w}_1^{(m)}, \dots, \mathbf{w}_{q_m}^{(m)}) \in \mathbb{R}^{(q_{m-1}+1) \times q_m}$ . The workhorse for these gradient calculations is the back-propagation method of Rumelhart et al. [324]. Basically, the back-propagation method is a clever

reparametrization of the problem so that the gradients can be calculated more easily. We therefore modify the weight matrices  $\mathcal{W}^{(m)}$  by dropping the first row containing the intercept parameters  $w_{0,j}^{(m)}$ ,  $1 \leq j \leq q_m$ . Define for  $1 \leq m \leq d+1$

$$\mathcal{W}_{(-0)}^{(m)} = \left( w_{j_{m-1}, j_m}^{(m)} \right)_{1 \leq j_{m-1} \leq q_{m-1}; 1 \leq j_m \leq q_m} \in \mathbb{R}^{q_{m-1} \times q_m},$$

where  $w_{j_{m-1}, j_m}^{(m)}$  denotes component  $j_{m-1}$  of  $\mathbf{w}_{j_m}^{(m)}$ , and where we set  $q_{d+1} = 1$  (output dimension) and  $w_{j_d, 1}^{(d+1)} = \beta_{j_d}$  for  $0 \leq j_d \leq q_d$ .

**Proposition 7.5 (Back-Propagation for the Hyperbolic Tangent Activation)** Choose a FN network of depth  $d \in \mathbb{N}$  and with hyperbolic tangent activation function  $\phi(x) = \tanh(x)$ .

- Define recursively
  - initialize  $q_{d+1} = 1$  and  $\delta^{(d+1)}(\mathbf{x}) = \mathbf{1} \in \mathbb{R}^{q_{d+1}}$ ;
  - iterate for  $d \geq m \geq 1$

$$\delta^{(m)}(\mathbf{x}) = \text{diag} \left( 1 - \left( z_{j_m}^{(m:1)}(\mathbf{x}) \right)^2 \right)_{1 \leq j_m \leq q_m} \mathcal{W}_{(-0)}^{(m+1)} \delta^{(m+1)}(\mathbf{x}) \in \mathbb{R}^{q_m}.$$

- We obtain for  $0 \leq m \leq d$

$$\left( \frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial w_{j_m, j_{m+1}}^{(m+1)}} \right)_{0 \leq j_m \leq q_m; 1 \leq j_{m+1} \leq q_{m+1}} = \mathbf{z}^{(m:1)}(\mathbf{x}) \delta^{(m+1)}(\mathbf{x})^\top \in \mathbb{R}^{(q_m+1) \times q_{m+1}},$$

where  $\mathbf{z}^{(0:1)}(\mathbf{x}) = \mathbf{x} \in \mathbb{R}^{q_0+1}$  and  $\mathbf{w}_1^{(d+1)} = \boldsymbol{\beta} \in \mathbb{R}^{q_d+1}$ .

**Proof of Proposition 7.5** Choose  $1 \leq m \leq d$  and define for the neurons  $1 \leq j_m \leq q_m$  the variables

$$\zeta_{j_m}^{(m)}(\mathbf{x}) = \left\langle \mathbf{w}_{j_m}^{(m)}, \mathbf{z}^{(m-1:1)}(\mathbf{x}) \right\rangle.$$

The learned representation in the  $m$ -th FN layer is obtained by activating these variables

$$\mathbf{z}^{(m:1)}(\mathbf{x}) = \left( 1, \phi \left( \zeta_1^{(m)}(\mathbf{x}) \right), \dots, \phi \left( \zeta_{q_m}^{(m)}(\mathbf{x}) \right) \right)^\top \in \mathbb{R}^{q_m+1}.$$

For the output we define

$$\zeta_1^{(d+1)}(\mathbf{x}) = \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle.$$

The main idea is to calculate the derivatives of  $\langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle$  w.r.t. these new variables  $\zeta_j^{(m)}(\mathbf{x})$ .

*Initialization for  $m = d + 1$*  This provides for  $m = d + 1$  and  $1 \leq j_{d+1} \leq q_{d+1} = 1$

$$\frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \zeta_1^{(d+1)}(\mathbf{x})} = 1 = \delta_1^{(d+1)}(\mathbf{x}).$$

*Recursion for  $m < d + 1$*  Next, we calculate the derivatives w.r.t.  $\zeta_{j_d}^{(d)}(\mathbf{x})$ , for  $m = d$  and  $1 \leq j_d \leq q_d$ . They are given by (note  $q_{d+1} = 1$ )

$$\begin{aligned} \frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \zeta_{j_d}^{(d)}(\mathbf{x})} &= \frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \zeta_1^{(d+1)}(\mathbf{x})} \frac{\partial \zeta_1^{(d+1)}(\mathbf{x})}{\partial \zeta_{j_d}^{(d)}(\mathbf{x})} \\ &= \delta_1^{(d+1)}(\mathbf{x}) \beta_{j_d} \phi'(\zeta_{j_d}^{(d)}(\mathbf{x})) \\ &= \delta_1^{(d+1)}(\mathbf{x}) w_{j_d,1}^{(d+1)} \left(1 - (z_{j_d}^{(d:1)}(\mathbf{x}))^2\right) = \delta_{j_d}^{(d)}(\mathbf{x}), \end{aligned} \quad (7.17)$$

where we have used  $w_{j_d,1}^{(d+1)} = \beta_{j_d}$  and for the hyperbolic tangent activation function  $\phi' = 1 - \phi^2$ . Continuing recursively for  $d > m \geq 1$  and  $1 \leq j_m \leq q_m$  we obtain

$$\begin{aligned} \frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \zeta_{j_m}^{(m)}(\mathbf{x})} &= \sum_{j_{m+1}=1}^{q_{m+1}} \frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \zeta_{j_{m+1}}^{(m+1)}(\mathbf{x})} \frac{\partial \zeta_{j_{m+1}}^{(m+1)}(\mathbf{x})}{\partial \zeta_{j_m}^{(m)}(\mathbf{x})} \\ &= \sum_{j_{m+1}=1}^{q_{m+1}} \delta_{j_{m+1}}^{(m+1)}(\mathbf{x}) w_{j_m, j_{m+1}}^{(m+1)} \left(1 - (z_{j_m}^{(m:1)}(\mathbf{x}))^2\right) = \delta_{j_m}^{(m)}(\mathbf{x}). \end{aligned}$$

Thus, the vectors  $\boldsymbol{\delta}^{(m)}(\mathbf{x}) = (\delta_1^{(m)}(\mathbf{x}), \dots, \delta_{q_m}^{(m)}(\mathbf{x}))^\top$  are calculated recursively in  $d \geq m \geq 1$  with initialization  $\boldsymbol{\delta}^{(d+1)}(\mathbf{x}) = \mathbf{1}$  and the recursion

$$\boldsymbol{\delta}^{(m)}(\mathbf{x}) = \text{diag} \left( 1 - (z_{j_m}^{(m:1)}(\mathbf{x}))^2 \right)_{1 \leq j_m \leq q_m} \mathcal{W}_{(-0)}^{(m+1)} \boldsymbol{\delta}^{(m+1)}(\mathbf{x}) \in \mathbb{R}^{q_m}.$$

Finally, we need to show how these derivatives are related to the original derivatives in the gradient descent method. We have for  $0 \leq j_d \leq q_d$  and  $j_{d+1} = 1$

$$\frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \beta_{j_d}} = \frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \zeta_1^{(d+1)}(\mathbf{x})} \frac{\partial \zeta_1^{(d+1)}(\mathbf{x})}{\partial \beta_{j_d}} = \delta_{j_{d+1}}^{(d+1)}(\mathbf{x}) z_{j_d}^{(d:1)}(\mathbf{x}).$$

For  $1 \leq m < d$ , and  $0 \leq j_m \leq q_m$  and  $1 \leq j_{m+1} \leq q_{m+1}$  we have

$$\frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial w_{j_m, j_{m+1}}^{(m+1)}} = \frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \zeta_{j_{m+1}}^{(m+1)}(\mathbf{x})} \frac{\partial \zeta_{j_{m+1}}^{(m+1)}(\mathbf{x})}{\partial w_{j_m, j_{m+1}}^{(m+1)}} = \delta_{j_{m+1}}^{(m+1)}(\mathbf{x}) z_{j_m}^{(m:1)}(\mathbf{x}).$$

For  $m = 0$ , and  $0 \leq l \leq q_0$  and  $1 \leq j_1 \leq q_1$  we have

$$\frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial w_{l, j_1}^{(1)}} = \frac{\partial \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle}{\partial \zeta_{j_1}^{(1)}(\mathbf{x})} \frac{\partial \zeta_{j_1}^{(1)}(\mathbf{x})}{\partial w_{l, j_1}^{(1)}} = \delta_{j_1}^{(1)}(\mathbf{x}) x_l.$$

This completes the proof of Proposition 7.5.  $\square$

*Remark 7.6* Proposition 7.5 gives the back-propagation method for the hyperbolic tangent activation function which has derivative  $\phi' = 1 - \phi^2$ . This becomes visible in the definition of  $\delta^{(m)}(\mathbf{x})$  where we consider the diagonal matrix

$$\text{diag} \left( 1 - \left( z_{j_m}^{(m:1)}(\mathbf{x}) \right)^2 \right)_{1 \leq j_m \leq q_m}.$$

For a general differentiable activation function  $\phi$  this needs to be replaced by, see (7.17),

$$\text{diag} \left( \phi' \left( \mathbf{w}_{j_m}^{(m)} \cdot \mathbf{z}^{(m-1:1)}(\mathbf{x}) \right) \right)_{1 \leq j_m \leq q_m}.$$

In the case of the sigmoid activation function this gives us, see also Table 7.1,

$$\text{diag} \left( z_{j_m}^{(m:1)}(\mathbf{x}) \left( 1 - z_{j_m}^{(m:1)}(\mathbf{x}) \right) \right)_{1 \leq j_m \leq q_m}.$$

#### Plain vanilla gradient descent algorithm for FN networks

1. Choose an initial network parameter  $\boldsymbol{\vartheta}^{(0)} \in \mathbb{R}^r$ .
2. Iterate for  $t \geq 0$  until a stopping criterion is met:
  - (a) Calculate the gradient  $\nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta})$  in network parameter  $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}^{(t)}$  using (7.16) and the back-propagation method of Proposition 7.5 (for the hyperbolic tangent activation function).
  - (b) Make the gradient descent step for a suitable learning rate  $\varrho_{t+1} > 0$

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}).$$

*Remark 7.7* The initialization  $\boldsymbol{\vartheta}^{(0)} \in \mathbb{R}^r$  of the gradient descent algorithm needs some care. A FN network has many symmetries, for instance, we can permute neurons within a FN layer and we receive the same predictive model. For this reason, the initial network weights  $\mathcal{W}^{(m)} = (\mathbf{w}_1^{(m)}, \dots, \mathbf{w}_{q_m}^{(m)}) \in \mathbb{R}^{(q_{m-1}+1) \times q_m}$ ,  $1 \leq m \leq d$ , should not be chosen with identical components because this will result in a saddlepoint of the corresponding objective function, and gradient descent will not work. For this reason, these weights are initialized randomly either using a uniform or a Gaussian distribution. The former is related to the `glorot_uniform` initializer in `keras`,<sup>2</sup> see (16) in Glorot–Bengio [160]. This initializer scales the support of the uniform distribution with the sizes of the FN layers that are connected by the corresponding weights  $\mathbf{w}_j^{(m)}$ .

For the output parameter we usually set as initial value  $\boldsymbol{\beta}^{(0)} = (\widehat{\beta}_0^{(0)}, 0, \dots, 0)^\top \in \mathbb{R}^{q_d+1}$ , where  $\widehat{\beta}_0^{(0)}$  is the MLE in the corresponding null model (not considering any features) and transformed to the chosen link  $g$ . This choice implies that the gradient descent algorithm starts in the null model, and any decrease in deviance loss can be seen as an improved in-sample loss of using the FN network regression structure over the null model.

## Stochastic Gradient Descent

The gradient in (7.16) has two parts. We have a vector

$$\mathbf{v}(Y) = \left( \frac{v_i}{\varphi} \left( \mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i) - Y_i \right) \frac{1}{V(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i))} \frac{1}{g'(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i))} \right)_{1 \leq i \leq n}^\top \in \mathbb{R}^n,$$

and we have a matrix

$$\mathbf{M} = \left( \nabla_{\boldsymbol{\vartheta}} \left\langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}_1) \right\rangle, \dots, \nabla_{\boldsymbol{\vartheta}} \left\langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}_n) \right\rangle \right) \in \mathbb{R}^{r \times n}.$$

The gradient of the deviance loss function is obtained by the matrix multiplication

$$\nabla_{\boldsymbol{\vartheta}} \mathcal{D}(Y, \boldsymbol{\vartheta}) = \frac{2}{n} \mathbf{M} \mathbf{v}(Y).$$

Matrix multiplication can be very slow in numerical implementations if the sample size  $n$  is large. For this reason, one typically uses the *stochastic gradient descent* (SGD) method that does not consider the entire data  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  simultaneously.

<sup>2</sup> For our examples we use the `R` library `keras` [77] which is an API to TensorFlow [2].

For the SGD method one chooses a fixed *batch size*  $b \in \mathbb{N}$ , and one randomly partitions the entire data  $\mathbf{Y}$  into (*mini-*)*batches*  $\mathbf{Y}_1, \dots, \mathbf{Y}_{\lfloor n/b \rfloor}$  of approximately the same size  $b$  (up to cardinality). Each gradient descent update

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} - \eta_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}_s, \boldsymbol{\vartheta}^{(t)}),$$

is then only based on the observations  $\mathbf{Y}_s$  in the corresponding batch  $1 \leq s \leq \lfloor n/b \rfloor$ . Typically, one sequentially visits all batches, and screening each batch once is called an *epoch*. Thus, if we run the SGD algorithm over  $K$  epochs on batches of size  $b \leq n$ , then we perform  $K \lfloor n/b \rfloor$  gradient descent steps.

Choosing batches of size  $b$  reduces the complexity of the matrix multiplication from  $n$  to  $b$ , and, henceforth, leads to much faster run times in one gradient descent step. On the other hand, batches should have a minimal size so that the gradient descent updates are not too erratic, i.e., if the batches are too small, the randomness in the data may point too often into a (completely) wrong direction for the optimal gradient descent step. For this reason, optimal batch sizes should be chosen carefully. For instance, if we study a low frequency claims count problem, say, with an expected frequency of  $\lambda = 10\%$ , we can determine confidence bounds for parameter estimation. This will provide an estimate of a minimal batch size  $b$  for a reliable parameter estimate.

To have a few erratic steps in SGD, however, can also be beneficial, as long as there are not too many of those. Sometimes, the algorithm gets trapped in saddlepoints or in flat areas of the objective function (vanishing gradient problem). If this is the case, an erratic step may be beneficial because it may perturb the algorithm out of its bottleneck. In fact, often SGD has a better performance than the plain vanilla gradient descent algorithm that is based on the entire data  $\mathbf{Y}$  because of these noisy contributions.

### Momentum-Based Gradient Descent Methods

The gradient descent method only considers a first order Taylor expansion and one is tempted to consider higher order terms to improve the approximation. For instance, Newton's method uses a second order Taylor term by updating

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} - \left( \nabla_{\boldsymbol{\vartheta}}^2 \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}) \right)^{-1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}). \quad (7.18)$$

In many practical applications this calculation is not feasible as the Hessian  $\nabla_{\boldsymbol{\vartheta}}^2 \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)})$  cannot be calculated in a reasonable amount of time. Another (simple) way of considering the changes in the gradients is the *momentum-based gradient descent method* of Rumelhart et al. [324]. This is inspired by mechanics in physics and it is achieved by considering the gradients over several iterations of the algorithm (with exponentially decaying weights). Choose a momentum coefficient  $\nu \in [0, 1)$  and define the initial speed  $\mathbf{v}^{(0)} = \mathbf{0} \in \mathbb{R}^r$ .

Replace the gradient descent update (7.15) by

$$\mathbf{v}^{(t)} \mapsto \mathbf{v}^{(t+1)} = \nu \mathbf{v}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}), \quad (7.19)$$

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} + \mathbf{v}^{(t+1)}. \quad (7.20)$$

For  $\nu = 0$  we have the plain vanilla gradient descent method, for  $\nu > 0$  we also memorize the previous gradients (with exponentially decaying weights). Typically this leads to better convergence properties.

Nesterov [284] has noticed that for convex functions the gradient descent updates may have a zig-zag behavior. Therefore, he proposed the so-called Nesterov-accelerated version

$$\begin{aligned} \mathbf{v}^{(t)} \mapsto \mathbf{v}^{(t+1)} &= \nu \mathbf{v}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)} + \nu \mathbf{v}^{(t)}), \\ \boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} &= \boldsymbol{\vartheta}^{(t)} + \mathbf{v}^{(t+1)}. \end{aligned} \quad (7.21)$$

Thus, the calculation of the momentum  $\mathbf{v}^{(t+1)}$  uses a look-ahead  $\boldsymbol{\vartheta}^{(t)} + \nu \mathbf{v}^{(t)}$  in the gradient calculation (anticipating part of the next step). This provides for the update (7.21) the following equivalent versions, under reparametrization  $\tilde{\boldsymbol{\vartheta}}^{(t)} = \boldsymbol{\vartheta}^{(t)} + \nu \mathbf{v}^{(t)}$ ,

$$\begin{aligned} \boldsymbol{\vartheta}^{(t+1)} &= \boldsymbol{\vartheta}^{(t)} + \left( \nu \mathbf{v}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)} + \nu \mathbf{v}^{(t)}) \right) \\ &= \boldsymbol{\vartheta}^{(t)} + \left( \nu \mathbf{v}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \tilde{\boldsymbol{\vartheta}}^{(t)}) \right) \\ &= \tilde{\boldsymbol{\vartheta}}^{(t)} + \left( \nu \mathbf{v}^{(t+1)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \tilde{\boldsymbol{\vartheta}}^{(t)}) \right) - \nu \mathbf{v}^{(t+1)}. \end{aligned} \quad (7.22)$$

For the Nesterov accelerated update we can also study, we use the last line of (7.22),

$$\begin{aligned} \mathbf{v}^{(t)} \mapsto \mathbf{v}^{(t+1)} &= \nu \mathbf{v}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \tilde{\boldsymbol{\vartheta}}^{(t)}), \\ \tilde{\boldsymbol{\vartheta}}^{(t)} \mapsto \tilde{\boldsymbol{\vartheta}}^{(t+1)} &= \tilde{\boldsymbol{\vartheta}}^{(t)} + \left( \nu \mathbf{v}^{(t+1)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \tilde{\boldsymbol{\vartheta}}^{(t)}) \right). \end{aligned} \quad (7.23)$$

Compared to (7.19)–(7.20), we just shift the index by 1 in the momentum  $\mathbf{v}^{(t)}$  in the round brackets of (7.23). The typical way how the Nesterov-acceleration is formulated is, yet, another equivalent formulation, namely, only in terms of  $\boldsymbol{\vartheta}^{(t)}$  and  $\tilde{\boldsymbol{\vartheta}}^{(t)}$ . From the second line of (7.22) and (7.21) we have the updates

$$\begin{aligned} \boldsymbol{\vartheta}^{(t+1)} &= \tilde{\boldsymbol{\vartheta}}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \tilde{\boldsymbol{\vartheta}}^{(t)}), \\ \tilde{\boldsymbol{\vartheta}}^{(t+1)} &= \boldsymbol{\vartheta}^{(t+1)} + \nu \left( \boldsymbol{\vartheta}^{(t+1)} - \boldsymbol{\vartheta}^{(t)} \right). \end{aligned} \quad (7.24)$$



Typically, one chooses the momentum coefficient  $\nu$  in (7.24) time-dependent by setting  $\nu_t = t/(t + 3)$ .

In our applications we will use the **R** interface to the **keras** library [77]. This library has a couple of standard momentum-based gradient descent methods implemented which use pre-defined learning rates and momentum coefficients. In our analysis we are mainly relying on the variants **rmsprop** and the Nesterov-accelerated version of **adam**, called **nadam**. Therefore, we briefly describe these three variants, and for more information we refer to Sections 8.3 and 8.5 in Goodfellow et al. [166].

### Predefined Gradient Descent Methods

- **rmsprop** stands for ‘root mean square propagation’, and its origin can be found in a lecture of Hinton et al. [187]. Denote by  $\odot$  the Hadamard product that computes the component-wise products of two matrices. Choose a weight  $\alpha \in (0, 1)$  and calculate the accumulated squared gradients, set  $\mathbf{r}^{(0)} = \mathbf{0} \in \mathbb{R}^r$ ,

$$\mathbf{r}^{(t)} \mapsto \mathbf{r}^{(t+1)} = \alpha \mathbf{r}^{(t)} + (1 - \alpha) \left( \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}) \odot \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}) \right) \in \mathbb{R}^r.$$

The sequence  $(\mathbf{r}^{(t)})_{t \geq 1}$  memorizes the (squared) magnitudes of the components of the gradients  $\nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)})$ ,  $t \geq 1$ . This is done individually for each component because we may have directional differences in magnitudes (and momentum). In contrast to (7.19),  $\mathbf{r}^{(t)}$  does not model the speed, but rather an inverse weight. This then motivates the gradient descent update

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} - \frac{\varrho}{\sqrt{\varepsilon + \mathbf{r}^{(t+1)}}} \odot \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}),$$

where the square-root is taken component-wise, for a global decay rate  $\varrho > 0$ , and for a small positive constant  $\varepsilon > 0$  to ensure that everything is well-defined.

- **adam** stands for ‘adaptive moment’ estimation, and it has been proposed by Kingma–Ba [216]. The momentum is determined by the first two moments in **adam**, namely, we set  $\mathbf{v}^{(0)} = \mathbf{r}^{(0)} = \mathbf{0} \in \mathbb{R}^r$  and we consider

$$\mathbf{v}^{(t)} \mapsto \mathbf{v}^{(t+1)} = \nu \mathbf{v}^{(t)} + (1 - \nu) \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}), \quad (7.25)$$

$$\mathbf{r}^{(t)} \mapsto \mathbf{r}^{(t+1)} = \alpha \mathbf{r}^{(t)} + (1 - \alpha) \left( \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}) \odot \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)}) \right), \quad (7.26)$$

for given weights  $\nu, \alpha \in (0, 1)$ . Similar to Bayesian credibility theory,  $\mathbf{v}^{(t)}$  and  $\mathbf{r}^{(t)}$  are biased because these two processes have been initialized in zero. Therefore, they are rescaled by  $1/(1 - \nu^t)$  and  $1/(1 - \alpha^t)$ , respectively. This gives us the gradient descent update

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} - \frac{\varrho}{\varepsilon + \sqrt{\frac{\mathbf{r}^{(t+1)}}{1 - \alpha^t}}} \odot \frac{\mathbf{v}^{(t+1)}}{1 - \nu^t},$$

where the square-root is taken component-wise, for a global decay rate  $\varrho > 0$ , and for a small positive constant  $\varepsilon > 0$  to ensure that everything is well-defined.

- `nadam` is the Nesterov-accelerated [284] version of `adam`. Similarly as when going from (7.19)–(7.20) to (7.23), the acceleration is obtained by a shift of 1 in the velocity parameter, thus, consider the Nesterov-accelerated `adam` update

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} - \frac{\varrho}{\varepsilon + \sqrt{\frac{\mathbf{r}^{(t+1)}}{1-\alpha^t}}} \odot \frac{\nu \mathbf{v}^{(t+1)} + (1-\nu) \nabla_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}^{(t)})}{1-\nu^t},$$

using (7.25) and (7.26).

## Maximum Likelihood Estimation and Over-fitting

As explained above, we model the mean of the datum  $(Y, \mathbf{x})$  by a deep FN network

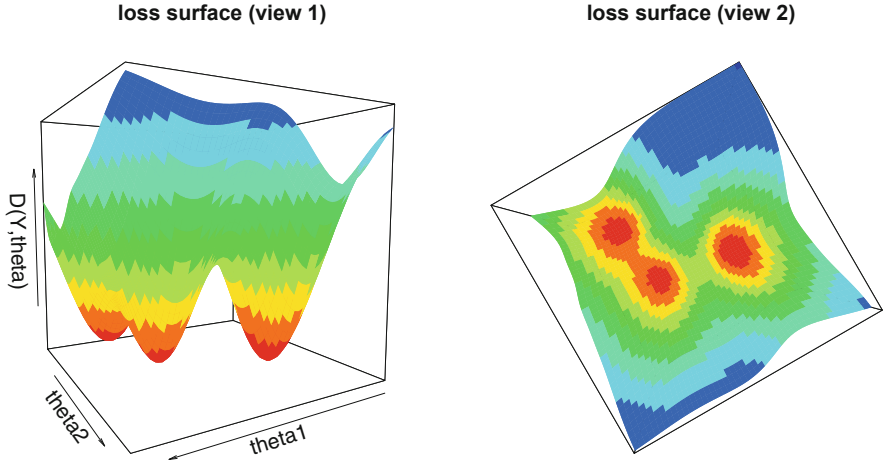
$$\mathbf{x} \mapsto \mu(\mathbf{x}) = \mu_{\boldsymbol{\vartheta}}(\mathbf{x}) = \mathbb{E}_{\theta(\mathbf{x})}[Y] = g^{-1} \left( \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \right),$$

for a network parameter  $\boldsymbol{\vartheta} \in \mathbb{R}^r$ . MLE of this network parameter requires solving for given data  $\mathbf{Y}$

$$\hat{\boldsymbol{\vartheta}}^{\text{MLE}} = \arg \min_{\boldsymbol{\vartheta}} \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta}).$$

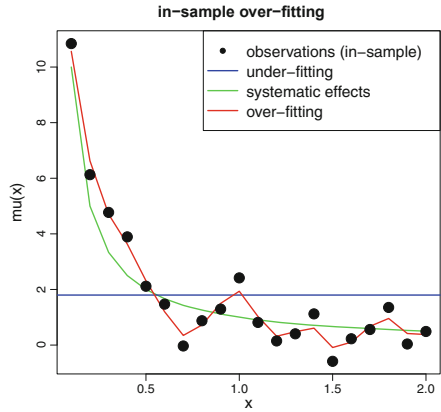
In Fig. 7.5 we give a schematic figure of a loss surface  $\boldsymbol{\vartheta} \mapsto \mathcal{D}(\mathbf{Y}, \boldsymbol{\vartheta})$  for a (low-dimensional) example  $\boldsymbol{\vartheta} \in \mathbb{R}^2$ . The two plots show the same loss surface from two different angles. This loss surface has three (local) minimums (red color), and the smallest one (global minimum) gives the MLE  $\hat{\boldsymbol{\vartheta}}^{\text{MLE}}$ .

In general, this global minimum cannot be found for more complex network architectures because the loss surface typically has a complicated structure for high-dimensional parameter spaces. Is this a problem in FN network fitting? Not really! We are going to explain why. The universality theorems in Sect. 7.2.2 state that more complex FN networks have an excellent approximation capacity. If we translate this to our statistical modeling problem it means that the observations  $\mathbf{Y}$  can be approximated arbitrarily well by sufficiently complex FN networks. In particular, for a given complex network architecture, the MLE  $\hat{\boldsymbol{\vartheta}}^{\text{MLE}}$  will provide the optimal fit of this architecture to the data  $\mathbf{Y}$ , and, as a result, this network does not only reflect the systematic effects in the data but also the noisy part. This behavior is called (*in-sample*) *over-fitting* to the learning data  $\mathcal{L}$ . It implies that such statistical models typically have a poor generalization to unseen (out-of-sample) test data  $\mathcal{T}$ ; this is illustrated by the red color in Fig. 7.6. For this reason, in general, we are not interested in finding the MLE  $\hat{\boldsymbol{\vartheta}}^{\text{MLE}}$  of  $\boldsymbol{\vartheta}$  in FN network regression modeling, but we would like to find a parameter estimate  $\hat{\boldsymbol{\vartheta}}$  that (only) extracts the systematic effects from the learning data  $\mathcal{L}$ . This is illustrated by the different colors in Figs. 7.5



**Fig. 7.5** Schematic figure of a loss surface  $\vartheta \mapsto \mathcal{D}(Y, \vartheta)$  from two different angles for a two-dimensional parameter  $\vartheta \in \mathbb{R}^2$

**Fig. 7.6** Schematic figure of in-sample over-fitting (red), under-fitting (blue) and extracting systematic effects (green)



and 7.6, where we assume: (a) red color provides models with a poor generalization power due to over-fitting, (b) blue color provides models with a poor generalization power, too, because these parametrizations do not explain the systematic effects in the data at all (called under-fitting), and (c) green color gives good parametrizations that explain the systematic effects in the data and generalize well to unseen data. Thus, the aim is to find parametrizations that are in the green area of Fig. 7.5. This green area emphasizes that we lose the notion of uniqueness because there are infinitely many models in the green area that have a comparable generalization

power. Next we explain how we can exploit the gradient descent algorithm to make it useful for finding parametrizations in the green area.

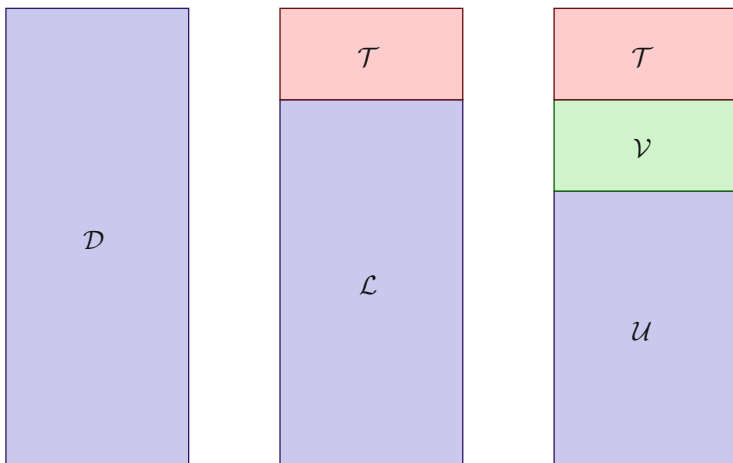
*Remark 7.8* The loss surface considerations in Fig. 7.5 are based on a fixed network architecture. Recent research promotes the so-called Graph HyperNetwork (GHN) that is a (hyper-)network which tries to find the optimal network architecture and its parametrization by an additional network, we refer to Zhang et al. [402] and Knyazev et al. [219].

## Regularization Through Early Stopping

As stated above, if we run the gradient descent algorithm with properly tempered learning rates it will converge to a local minimum of the loss function, which means that the resulting FN network over-fits to the learning data. For this reason we need to *early stop* the gradient descent algorithm beforehand. Coming back to Fig. 7.5, typically, we start the gradient descent algorithm somewhere in the blue area of the loss surface (supposed that the red area is a sparse set on the loss surface). Visually speaking, the gradient descent algorithm then walks down the valley (green, yellow and red area) by exploiting locally optimal steps. Since at the early stage of the algorithm the systematic effects play a dominant role over the noisy part, the gradient descent algorithm learns these systematic effects at this first stage (blue area in Fig. 7.5). When the algorithm arrives at the green area the noisy part in the data starts to increasingly influence the model calibration (gradient descent steps), and, henceforth, at this stage the algorithm should be stopped, and the learned parameter should be selected for predictive modeling. This early stopping is an implicit way of regularization, because it implies that we stop the parameter fitting before the parameters start to learn very individual features of the (noisy) data (and take extreme values).

This early stopping point is determined by doing an out-of-sample analysis. This requires the learning data  $\mathcal{L}$  to be further split into *training data*  $\mathcal{U}$  and *validation data*  $\mathcal{V}$ . The training data  $\mathcal{U}$  is used for gradient descent parameter learning, and the validation data  $\mathcal{V}$  is used for tracking the over-fitting by an instantaneous (out-of-sample) validation analysis. This partition is illustrated in Fig. 7.7, which also highlights that the validation data  $\mathcal{V}$  is disjoint from the test data  $\mathcal{T}$ , the latter only being used in the final step for comparing different statistical models (e.g., a GLM vs. a FN network). That is, model comparison is done in a proper out-of-sample manner on  $\mathcal{T}$ , and each of these models is only fit on  $\mathcal{U}$  and  $\mathcal{V}$ . Thus, for FN network fitting with early stopping we need a reasonable amount of data that can be split into 3 sufficiently large data sets so that each is suitable for its purpose.

For early stopping we partition the learning data  $\mathcal{L}$  into training data  $\mathcal{U}$  and validation data  $\mathcal{V}$ . The plain vanilla gradient descent algorithm can then be changed as follows.



**Fig. 7.7** Partition of entire data  $\mathcal{D}$  (lhs) into learning data  $\mathcal{L}$  and test data  $\mathcal{T}$  (middle), and into training data  $\mathcal{U}$ , validation data  $\mathcal{V}$  and test data  $\mathcal{T}$  (rhs)

---

### Plain vanilla gradient descent algorithm with early stopping

---

1. Choose an initial network parameter  $\boldsymbol{\vartheta}^{(0)} \in \mathbb{R}^r$ .
2. Iterate for  $t \geq 0$  until the early stopping criterion is met:
  - (a) Calculate the gradient  $\nabla_{\boldsymbol{\vartheta}} \mathfrak{D}(\mathcal{U}, \boldsymbol{\vartheta})$  in network parameter  $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}^{(t)}$  on the training data  $\mathcal{U}$  using (7.16) and the back-propagation method of Proposition 7.5 (for the hyperbolic tangent activation function).
  - (b) Make the gradient descent step for a suitable learning rate  $\varrho_{t+1} > 0$

$$\boldsymbol{\vartheta}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t+1)} = \boldsymbol{\vartheta}^{(t)} - \varrho_{t+1} \nabla_{\boldsymbol{\vartheta}} \mathfrak{D}(\mathcal{U}, \boldsymbol{\vartheta}^{(t)}).$$

- (c) Calculate the validation loss  $\mathfrak{D}(\mathcal{V}, \boldsymbol{\vartheta}^{(t)})$  on the validation data  $\mathcal{V}$ .
- (d) Stop the algorithm if the validation loss increases, i.e., if

$$\mathfrak{D}(\mathcal{V}, \boldsymbol{\vartheta}^{(t)}) > \mathfrak{D}(\mathcal{V}, \boldsymbol{\vartheta}^{(t-1)}), \quad (7.27)$$

and return the learned parameter (estimate)  $\widehat{\boldsymbol{\vartheta}} = \boldsymbol{\vartheta}^{(t-1)}$ .

---

In applications we use the SGD algorithm that can also have erratic steps because not all random (mini-)batches are necessarily typical representations of the data. In such cases we should use more sophisticated stopping criteria than (7.27), for instance, early stop if the validation loss increases five times in a row.

**Fig. 7.8** Training loss  $\mathcal{D}(\mathcal{U}, \boldsymbol{\vartheta}^{(t)})$  vs. validation loss  $\mathcal{D}(\mathcal{V}, \boldsymbol{\vartheta}^{(t)})$  over different iterations  $t \geq 0$  of the SGD algorithm

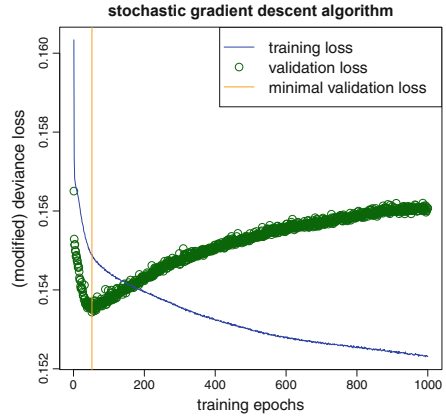


Figure 7.8 provides an example of the application of the SGD algorithm on training data  $\mathcal{U}$  and validation data  $\mathcal{V}$ . The training loss is in blue color and the validation loss in green color. We observe that the validation loss has its minimum after 52 epochs (orange vertical line), and hence the fitting algorithm should be stopped at this point. We give a couple of remarks concerning Fig. 7.8:

- The learning data  $\mathcal{L}$  exactly corresponds to the claims frequency data of Sect. 5.2.4, see also Table 5.2. We take 10% as validation data which gives  $|\mathcal{U}| = 549'185$  and  $|\mathcal{V}| = 61'021$ . For the SGD algorithm we use batches of size 10'000 which implies that one epoch corresponds to  $\lfloor 549'185/10'000 \rfloor = 54$  gradient descent steps. For batches of size 10'000 we expect an approximate estimation precision on an average frequency of  $\bar{\lambda} = 7.36\%$  in the Poisson model of

$$\left[ \bar{\lambda} - 2 \sqrt{\frac{\bar{\lambda}}{10'000\bar{v}}}, \bar{\lambda} + 2 \sqrt{\frac{\bar{\lambda}}{10'000\bar{v}}} \right] = [6.62\%, 8.11\%],$$

with an average exposure  $\bar{v} = 0.5283$  on our learning data, we also refer to Example 3.22.

- The FN network architecture used in Fig. 7.8 is the one shown in Fig. 7.2 using one-hot encoding for categorical variables, see Sect. 7.3.1, below, and the responses are modeled by a Poisson distribution.
- The training loss  $\mathcal{D}(\mathcal{U}, \boldsymbol{\vartheta}^{(t)})$ , blue curve in Fig. 7.8, is a bit wiggly which comes from the fact that we use a SGD where not every batch leads to the optimal decrease in loss. Remark that the loss figures in the graph correspond to average losses over an entire epoch, i.e., in our case an average over 54 SGD steps. Also remark that the y-scale does not show the Poisson deviance loss: we use the loss figures provided by *keras* [77] and these figures drop all terms of the deviance loss that are not relevant for parameter estimation.

We close this section with remarks.

*Remarks 7.9*

- We perform early stopping because otherwise a complex FN network would in-sample over-fit to the learning data. At this stage, one could be tempted to choose a smaller network to prevent from over-fitting. In general, this is not a sensible thing to do because the network needs sufficient flexibility to be able to be fitted to the data. That is, we need some redundancy in the model to be able to successfully apply the SGD algorithm, otherwise the algorithm may get trapped in saddlepoints or bottlenecks. Thus, the chosen network architecture should be above the bound of a necessary minimal complexity, and different architectures above this bound will provide similar accuracy (without a clear winner).
- The chosen network will contain certain elements of randomness, and different runs of the SGD algorithm will provide different solutions. Firstly, the initialization  $\boldsymbol{\vartheta}^{(0)} \in \mathbb{R}^r$  of the algorithm is chosen at random, and since we early stop the algorithm and because we do not have a unique optimal point, the chosen solution will depend on this random initialization. Secondly, the split between training and validation data is done at random, and thirdly the partitioning of the training data into mini-batches is done at random. All these random elements make the early stopping SGD solution non-unique.
- Early stopping implies that the chosen network parameter estimate  $\hat{\boldsymbol{\vartheta}}$  does not correspond to a solution of the score equations and, henceforth, asymptotic results about MLEs do not apply, see Theorem 3.28.

## 7.3 Feed-Forward Neural Network Examples

### 7.3.1 Feature Pre-processing

Similarly to GLMs, we also need to pre-process the feature components in FN network regression modeling. The former Sect. 5.2.2 for GLMs has been called ‘feature engineering’ because we need to bring the feature components into an appropriate functional form w.r.t. the given regression task. The present section is called ‘feature pre-processing’ because we do not need to engineer the features for FN networks. We only need to bring them into a suitable (tabular) form to enter the network, and the network will then do an automated feature engineering through representation learning.

#### Categorical Feature Components: One-Hot Encoding

The categorical features have been treated by dummy coding within GLMs. Dummy coding provides full rank design matrices. For FN network regression modeling the

**Table 7.2** One-hot encoding example mapping the  $K = 11$  levels (colors) to the unit vectors of the 11-dimensional Euclidean space  $\mathbb{R}^{11}$  showing the resulting encoding vectors  $\mathbf{x}_j^\top$  as row vectors

$a_1 = \text{white}$	1	0	0	0	0	0	0	0	0	0	0
$a_2 = \text{yellow}$	0	1	0	0	0	0	0	0	0	0	0
$a_3 = \text{orange}$	0	0	1	0	0	0	0	0	0	0	0
$a_4 = \text{red}$	0	0	0	1	0	0	0	0	0	0	0
$a_5 = \text{magenta}$	0	0	0	0	1	0	0	0	0	0	0
$a_6 = \text{violet}$	0	0	0	0	0	1	0	0	0	0	0
$a_7 = \text{blue}$	0	0	0	0	0	0	1	0	0	0	0
$a_8 = \text{cyan}$	0	0	0	0	0	0	0	1	0	0	0
$a_9 = \text{green}$	0	0	0	0	0	0	0	0	1	0	0
$a_{10} = \text{beige}$	0	0	0	0	0	0	0	0	0	1	0
$a_{11} = \text{brown}$	0	0	0	0	0	0	0	0	0	0	1

full rank property is not important because, anyway, we neither have a single (local) minimum in the objective function, nor do we want to calculate the MLE of the network parameter. Typically, in FN network regression modeling one uses one-hot encoding for the categorical variables that encodes every level by a unit vector. Assume the raw feature component  $\tilde{x}_j$  is a categorical variable taking  $K$  different levels  $\{a_1, \dots, a_K\}$ . One-hot encoding is obtained by the embedding map

$$\tilde{x}_j \mapsto \mathbf{x}_j = (\mathbb{1}_{\{\tilde{x}_j=a_1\}}, \dots, \mathbb{1}_{\{\tilde{x}_j=a_K\}})^\top \in \{0, 1\}^K. \quad (7.28)$$

An explicit example is given in Table 7.2 which should be compared to Table 5.1.

## Continuous Feature Components

The continuous feature components do not need any pre-processing but they can directly enter the FN network which will take care of representation learning. However, an efficient use of gradient descent methods typically requires that all feature components live on a similar scale and that they are roughly uniformly spread across their domains. This makes gradient descent steps more efficient in exploiting the relevant directions.

One possibility is to use the MinMaxScaler. Let  $x_j^-$  and  $x_j^+$  be the minimal and maximal possible feature values of the continuous feature component  $x_j$ , i.e.,  $x_j \in [x_j^-, x_j^+]$ . We transform this continuous feature component to unit scale for all data  $1 \leq i \leq n$  by

$$x_{i,j} \mapsto x_{i,j}^{\text{MM}} = 2 \frac{x_{i,j} - x_j^-}{x_j^+ - x_j^-} - 1 \in [-1, 1]. \quad (7.29)$$

The resulting feature values  $(x_{i,j}^{\text{MM}})_{1 \leq i \leq n}$  should roughly be uniformly spread across the interval  $[-1, 1]$ . If this is not the case, for instance, because we have outliers in the feature values, we may first transform them non-linearly to get



more uniformly spread values. For example, we consider the Density of the car frequency example on the log scale.

An alternative to the MinMaxScaler is to consider normalization with the empirical mean  $\bar{x}_j$  and the empirical standard deviation  $\hat{\sigma}_j$  over all data  $x_{i,j}$ . That is,

$$x_{i,j} \mapsto x_{i,j}^{\text{sd}} = \frac{x_{i,j} - \bar{x}_j}{\hat{\sigma}_j}. \quad (7.30)$$

It depends on the application whether the MinMaxScaler or normalization with the empirical mean and standard deviation works better. Important in applications is that we use exactly the same values for the normalization of training data  $\mathcal{U}$ , validation data  $\mathcal{V}$  and test data  $\mathcal{T}$ , to make the same network applicable to all these data sets. For notational convenience we will drop the upper index in  $x_{i,j}^{\text{MM}}$  or  $x_{i,j}^{\text{sd}}$ , respectively, and we throughout assume that all feature components are appropriately pre-processed.

### 7.3.2 Lab: Poisson FN Network for Car Insurance Frequencies

We present a first FN network example applied to the French MTPL claim frequency data studied in Sect. 5.2.4. We assume that the claim counts  $N_i$  are independent and Poisson distributed with claim count density (5.26), where we replace the GLM regression function  $\mathbf{x} \mapsto \exp\langle \boldsymbol{\beta}, \mathbf{x} \rangle$  by a FN network regression function

$$\mathbf{x} \in \mathcal{X} \mapsto \mu(\mathbf{x}) = \exp\langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle.$$

We use a FN network of depth  $d = 3$  having number of neurons  $(q_1, q_2, q_3) = (20, 15, 10)$  and using the hyperbolic tangent activation function. We pre-process the categorical variables VehBrand and Region by one-hot encoding providing input dimensions 11 and 22, respectively. The binary variable VehGas is encoded as 0–1. Because of scarcity of data we right-censor the continuous variables VehAge at 20, DrivAge at 90 and BonusMalus at 150, and we transform Density to the log scale. We then apply to each of these (modified) continuous variables Area, VehPower, VehAge, DrivAge, BonusMalus and  $\log(\text{Density})$  a MinMaxScaler. This provides us with an input dimension  $q_0 = 11 + 22 + 1 + 6 = 40$ . The resulting FN network is illustrated in Fig. 7.2, with the one-hot encoded variables VehBrand in orange color and Region in magenta color. It has a network parameter  $\boldsymbol{\vartheta} \in \mathbb{R}^r$  of dimension  $r = 1'306$ .

This network is implemented in R using the library keras [77]. The code is provided in Listing 7.1 and the resulting network architecture is summarized in Listing 7.2. This network is now fitted to the data. We use a batch size of 10'000, we use the nadam version of SGD, we use 10% of the learning data  $\mathcal{L}$  as validation data  $\mathcal{V}$  and the remaining 90% as training data  $\mathcal{U}$ . We then run the corresponding

**Listing 7.1** FN network of depth  $d = 3$  using the R library keras [77]

---

```

1 library(keras)
2 #
3 Design = layer_input(shape = c(40), dtype = 'float32', name = 'Design')
4 Vol     = layer_input(shape = c(1), dtype = 'float32', name = 'Vol')
5 #
6 Network = Design %>%
7   layer_dense(units=20, activation='tanh', name='FNNLayer1') %>%
8   layer_dense(units=15, activation='tanh', name='FNNLayer2') %>%
9   layer_dense(units=10, activation='tanh', name='FNNLayer3') %>%
10  layer_dense(units=1, activation='exponential', name='Network',
11    weights=list(array(0, dim=c(10,1)), array(log(lambda0), dim=c(1))))
12 #
13 Response = list(Network, Vol) %>% layer_multiply(name='Multiply')
14 #
15 model = keras_model(inputs = c(Design, Vol), outputs = c(Response))
16 #
17 summary(model)

```

---

**Listing 7.2** FN network illustrated in Fig. 7.2

---

Layer (type)	Output Shape	Param #	Connected to
Design (InputLayer)	(None, 40)	0	
FNNLayer1 (Dense)	(None, 20)	820	Design[0][0]
FNNLayer2 (Dense)	(None, 15)	315	FNNLayer1[0][0]
FNNLayer3 (Dense)	(None, 10)	160	FNNLayer2[0][0]
Network (Dense)	(None, 1)	11	FNNLayer3[0][0]
Vol (InputLayer)	(None, 1)	0	
Multiply (Multiply)	(None, 1)	0	Network[0][0] Vol[0][0]

---

```

18 Total params: 1,306
19 Trainable params: 1,306
20 Non-trainable params: 0

```

---

**Listing 7.3** Fitting a FN network using the R library keras [77]

---

```

1 path0 <- "path_for_callback"
2 CBs    <- callback_model_checkpoint(path0, monitor = "val_loss", verbose = 0,
3   save_best_only = TRUE, save_weights_only = TRUE)
4 #
5 model %>% compile(loss = 'poisson', optimizer = 'nadam')
6 fit <- model %>% fit(list(Xlearn, Vlearn), Ylearn, validation_split=0.1,
7   batch_size=10000, epochs=1000, verbose=0, callbacks=CBs)
8 #
9 load_model_weights_hdf5(model, path0)

```

---

**Table 7.3** Run times, number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model, model Poisson GLM3 of Table 5.5 and the FN network model (with one-hot encoding of the categorical variables)

	Run time	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	25.213	25.445	7.36%
Poisson GLM3	15 s	50	24.084	24.102	7.36%
One-hot FN $(q_1, q_2, q_3) = (20, 15, 10)$	51 s	1'306	23.757	23.885	6.96%

SGD algorithm and we retrieve the network with the lowest validation loss using a `callback`. This is illustrated in Listing 7.3. The fitting performance on the training and validation data is illustrated in Fig. 7.8, and we retrieve the network calibration after the 52th epoch because it has the lowest validation loss. The results are presented in Table 7.3.

From the results of Table 7.3 we conclude that the FN network outperforms model Poisson GLM3 (out-of-sample) since it has a (clearly) lower out-of-sample deviance loss on the test data  $\mathcal{T}$ . This may indicate that there is an interaction between the feature components that has not been captured in the GLM. The run time of 51s corresponds to the run time until the minimal validation loss is reached, of course, in practice we need to continue beyond this minimal validation loss to ensure that we have really found the minimum. Finally, and importantly, we observe that this early stopped FN network calibration does not meet the balance property because the resulting average frequency of this fitted model of 6.96% is below the empirical frequency of 7.36%. This is a major deficiency of this FN network fitting approach, and this is going to be discussed further in Sect. 7.4.2, below.

We can perform a detailed analysis about different batch sizes, variants of SGD methods, run times, etc. We briefly summarize our findings; this summary is also based on the findings in Ferrario et al. [127]. We have fitted this model on batches of sizes 2'000, 5'000, 10'000 and 20'000, and it seems that a batch size around 5'000 has the best performance, both concerning out-of-sample performance and run time to reach the minimal validation loss. Comparing the different optimizers `rmsprop`, `adam` and `nadam`, a clear preference can be given to `nadam`: the resulting prediction accuracy is similar in all three optimizers (they all reach the green area in Fig. 7.5), but `nadam` reaches this optimal point in half of the time compared to `rmsprop` and `adam`.

We conclude by highlighting that different initial points  $\vartheta^{(0)}$  of the SGD algorithm will give different network calibrations, and differences can be considerable. This is discussed in Sect. 7.4.4, below. Moreover, we could explore different network architectures, more simple ones, more complex ones, different activation functions, etc. The results of these different architectures will not be essentially different from our results, as long as the networks are above a minimal complexity bound. This closes our first example on FN networks and this example is the benchmark for refined versions that are presented in the subsequent sections.

## 7.4 Special Features in Networks

### 7.4.1 Special Purpose Layers

So far, our networks consist of stacked FN layers, and information is passed in a directed acyclic feed-forward path from one to the next FN layer. In this section we discuss special purpose layers that perform a specific task in a FN network. These include *embedding layers*, *drop-out layers* and *normalization layers*. These modules should be seen as add-ons to the FN layers. Besides these add-ons, there are also *recurrent layers* and *convolutional layers*. These two types of layers are going to be discussed in own chapters, below, because their importance goes beyond just being add-ons to the FN layers.

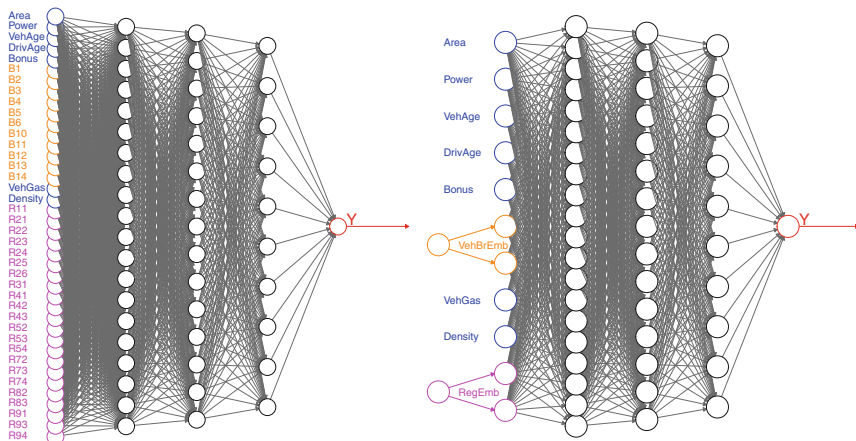
#### Embedding Layers for Categorical Feature Components

The categorical feature components have been treated either by dummy coding or by one-hot encoding, and this has resulted in numerous network parameters in the first FN layer, see Fig. 7.2. Natural language processing (NLP) treats categorical feature components differently, namely, it *embeds* categorical feature components (or words in NLP) into a Euclidean space  $\mathbb{R}^b$  of a *small* dimension  $b$ . This small dimension  $b$  is a hyper-parameter that has to be selected by the modeler, and which, typically, is selected much smaller than the total number of levels of the categorical feature. This embedding technique is quite common in NLP, see Bengio et al. [27–29], but it goes beyond NLP applications, see Guo–Berkhahn [176], and it has been introduced to the actuarial community by Richman [312, 313] and the tutorial of Schellldorfer–Wüthrich [329].

We assume the same set-up as in dummy coding (5.21) and in one-hot encoding (7.28), namely, that we have a raw categorical feature component  $\tilde{x}_j$  taking  $K$  different levels  $\{a_1, \dots, a_K\}$ . In one-hot encoding these  $K$  levels are mapped to the  $K$  unit vectors of the Euclidean space  $\mathbb{R}^K$ , and consequently all levels have the same mutual Euclidean distance. This does not seem to be the best way of comparing the different levels because in our regression analysis we would like to identify the levels that are more similar w.r.t. the regression task and, thus, these should cluster. For an *embedding layer* one chooses a Euclidean space  $\mathbb{R}^b$  of a dimension  $b < K$ , typically being (much) smaller than  $K$ . One then considers the *embedding map*

$$\mathbf{e} : \{a_1, \dots, a_K\} \rightarrow \mathbb{R}^b, \quad a_k \mapsto \mathbf{e}(a_k) \stackrel{\text{def.}}{=} \mathbf{e}^{(k)}. \quad (7.31)$$

That is, every level  $a_k$  receives a vector representation  $\mathbf{e}^{(k)} \in \mathbb{R}^b$  which is lower dimensional than its one-hot encoding counterpart in  $\mathbb{R}^K$ . Proximity of the representations  $\mathbf{e}^{(k)}$  and  $\mathbf{e}^{(k')}$  in  $\mathbb{R}^b$ , i.e., of two levels  $a_k$  and  $a_{k'}$ , should be related to similarity w.r.t. the regression task at hand. Such an embedding involves  $K$



**Fig. 7.9** (lhs) One-hot encoding with  $q_0 = 40$ , and (rhs) embedding layers for VehBrand and Region with embedding dimension  $b = 2$  and  $q_0 = 11$ ; the remaining network architecture is identical with  $(q_1, q_2, q_3) = (20, 15, 10)$  for depth  $d = 3$

vectors  $\mathbf{e}^{(k)} \in \mathbb{R}^b$  of dimension  $b$ , thus, it involves  $Kb$  parameters, called *embedding weights*.

In network modeling, these embedding weights  $\mathbf{e}^{(1)}, \dots, \mathbf{e}^{(K)}$  can also be learned during gradient descent training. Basically, it just means that for the categorical variables we add an additional embedding layer before the first FN layer  $\mathbf{z}^{(1)}$ , i.e., we increase the depth of the network by 1 for the categorical feature components (by a layer that is not fully connected). This is illustrated in Fig. 7.9 (rhs) for the French MTPL insurance example of Sect. 7.3.2. The graph on the left-hand side shows the network if we apply one-hot encoding to the categorical variables VehBrand and Region; this results in a network parameter of dimension  $r = 1'306$ . The graph on the right-hand side first embeds VehBrand and Region into two 2-dimensional spaces, illustrated by the orange and magenta circles. These embeddings are concatenated with the remaining feature components, which then provides a new dimension  $q_0 = 7 + 2 + 2 = 11$  in that example. This results in a network parameter of dimension  $r = 726 + 22 + 44 = 792$ , where  $22 + 44 = 66$  stands for the 2-dimensional embedding weights of the 11 VehBrands and the 22 French Regions, see Listing 7.5.

**Example 7.10 (Embedding Layers for Categorical Features)** We revisit the example of Sect. 7.3.2, but we replace one-hot encoding of the categorical variables by embedding layers of dimension  $b = 2$ . The corresponding R code is given in Listing 7.4 and the resulting model is illustrated in Listing 7.5 and Fig. 7.9 (rhs).

Apart from replacing one-hot encoding by embedding layers, we use exactly the same FN network architecture as in Sect. 7.3.2 and we apply the same fitting strategy in terms of batch sizes, optimizer and early stopping strategy. The results are presented in Table 7.4.

**Listing 7.4** FN network of depth  $d = 3$  using embedding layers

---

```

1 Design = layer_input(shape = c(7), dtype = 'float32', name = 'Design')
2 VehBrand = layer_input(shape = c(1), dtype = 'int32', name = 'VehBrand')
3 Region = layer_input(shape = c(1), dtype = 'int32', name = 'Region')
4 Vol = layer_input(shape = c(1), dtype = 'float32', name = 'Vol')
5 #
6 BrandEmb = VehBrand %>%
7   layer_embedding(input_dim=11,output_dim=2,input_length=1,name='BrandEmb') %>%
8   layer_flatten(name='Brand_flat')
9 RegionEmb = Region %>%
10   layer_embedding(input_dim=22,output_dim=2,input_length=1,name='RegionEmb') %>%
11   layer_flatten(name='Region_flat')
12 #
13 Network = list(Design,BrandEmb,RegionEmb) %>% layer_concatenate(name='concat') %>%
14   layer_dense(units=20, activation='tanh', name='FNLayer1') %>%
15   layer_dense(units=15, activation='tanh', name='FNLayer2') %>%
16   layer_dense(units=10, activation='tanh', name='FNLayer3') %>%
17   layer_dense(units=1, activation='exponential', name='Network',
18     weights=list(array(0, dim=c(10,1)), array(log(lambda0), dim=c(1))))
19 #
20 Response = list(Network, Vol) %>% layer_multiply(name='Multiply')
21 #
22 model = keras_model(inputs = c(Design, VehBrand, Region, Vol),
23   outputs = c(Response))

```

---

**Table 7.4** Run times, number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model, model Poisson GLM3 of Table 5.5 and the FN network models (with one-hot encoding and embedding layers of dimension  $b = 2$ , respectively)

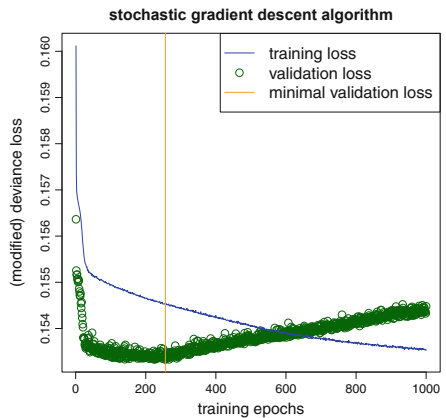
	Run time	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	25.213	25.445	7.36%
Poisson GLM3	15 s	50	24.084	24.102	7.36%
One-hot FN $(q_1, q_2, q_3) = (20, 15, 10)$	51 s	1'306	23.757	23.885	6.96%
Embed FN $(q_1, q_2, q_3) = (20, 15, 10)$	120 s	792	23.694	23.820	7.24%

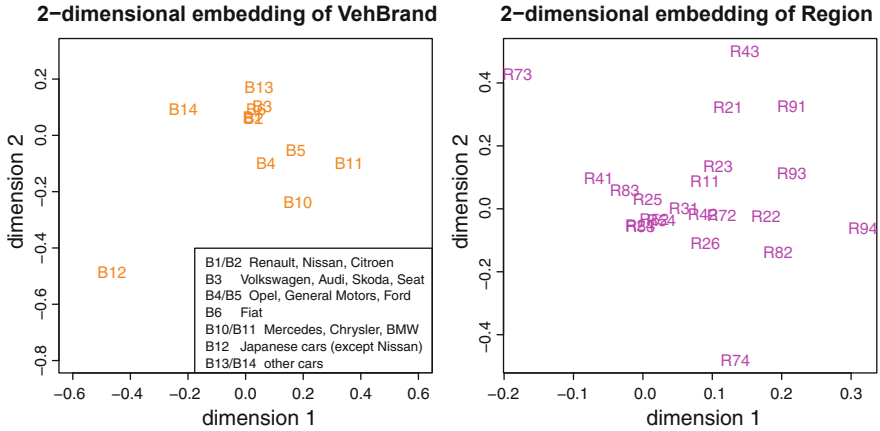
A first remark is that the model calibration takes longer using embedding layers compared to one-hot encoding. The main reason for this is that having an embedding layer increases the depth of the network by one layer, as can be seen from Fig. 7.9. Therefore, the back-propagation takes more time, and the convergence is slower requiring more gradient descent steps. We have less over-fitting as can be seen from Fig. 7.10. The final fitted model has a slightly better out-of-sample performance compared to the one-hot encoding one. However, this slight improvement in the performance should not be overstated because, as explained in Remarks 7.9, there are a couple of elements of randomness involved in SGD fitting, and choosing a different seed may change the results. We remark that the balance property is not fulfilled because the average frequency of the fitted model does not meet the empirical frequency, see the last column of Table 7.4; we come back to this in Sect. 7.4.2, below.

**Listing 7.5** Summary of FN network of Fig. 7.9 (rhs) using embedding layers of dimension  $b = 2$

Layer (type)	Output Shape	Param #	Connected to
VehBrand (InputLayer)	(None, 1)	0	
Region (InputLayer)	(None, 1)	0	
BrandEmb (Embedding)	(None, 1, 2)	22	VehBrand[0][0]
RegionEmb (Embedding)	(None, 1, 2)	44	Region[0][0]
Design (InputLayer)	(None, 7)	0	
Brand_flat (Flatten)	(None, 2)	0	BrandEmb[0][0]
Region_flat (Flatten)	(None, 2)	0	RegionEmb[0][0]
concat (Concatenate)	(None, 11)	0	Design[0][0] Brand_flat[0][0] Region_flat[0][0]
FNLayer1 (Dense)	(None, 20)	240	concat[0][0]
FNLayer2 (Dense)	(None, 15)	315	FNLayer1[0][0]
FNLayer3 (Dense)	(None, 10)	160	FNLayer2[0][0]
Network (Dense)	(None, 1)	11	FNLayer3[0][0]
Vol (InputLayer)	(None, 1)	0	
Multiply (Multiply)	(None, 1)	0	Network[0][0] Vol[0][0]
Total params: 792			
Trainable params: 792			
Non-trainable params: 0			

**Fig. 7.10** Training loss  $\mathcal{D}(\mathcal{U}, \boldsymbol{\vartheta}^{(t)})$  vs. validation loss  $\mathcal{D}(\mathcal{V}, \boldsymbol{\vartheta}^{(t)})$  over different iterations  $t \geq 0$  of the SGD algorithm in the deep FN network with embedding layers for categorical variables





**Fig. 7.11** Embedding weights  $e^{\text{VehBrand}} \in \mathbb{R}^2$  and  $e^{\text{Region}} \in \mathbb{R}^2$  of the categorical variables VehBrand and Region for embedding dimension  $b = 2$

A major advantage of using embedding layers for the categorical variables is that we receive a continuous representation of nominal variables, where proximity can be interpreted as similarity for the regression task at hand. This is nicely illustrated in Fig. 7.11 which shows the resulting 2-dimensional embeddings  $e^{\text{VehBrand}} \in \mathbb{R}^2$  and  $e^{\text{Region}} \in \mathbb{R}^2$  of the categorical variables VehBrand and Region. The Region embedding  $e^{\text{Region}} \in \mathbb{R}^2$  shows surprising similarities with the French map, for instance, Paris region R11 is adjacent to R23, R22, R21, R26, R24 (which is also the case in the French map), the Isle of Corsica R94 and the South of France R93, R91 and R73 are well separated from other regions, etc. Similar observations can be made for the embedding of VehBrand, Japanese cars B12 are far apart from the other cars, cars B1, B2, B3 and B6 (Renault, Nissan, Citroen, Volkswagen, Audi, Skoda, Seat and Fiat) cluster, etc. ■

## Drop-Out Layers and Regularization

Above, over-fitting to the learning data has been taken care of by early stopping. In view of Sect. 6.2 one could also use regularization. This can easily be obtained by replacing (7.14), for instance, by the following  $L^p$ -regularized counterpart

$$\boldsymbol{\vartheta} \mapsto \frac{2}{n} \sum_{i=1}^n \frac{v_i}{\varphi} \left( Y_i h(Y_i) - \kappa(h(Y_i)) - Y_i h(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i)) + \kappa(h(\mu_{\boldsymbol{\vartheta}}(\mathbf{x}_i))) \right) + \lambda \|\boldsymbol{\vartheta}\|_p^p,$$

for some  $p \geq 1$ , regularization parameter  $\lambda > 0$  and where the reduced network parameter  $\boldsymbol{\vartheta}_- \in \mathbb{R}^{r-1}$  excludes the intercept parameter  $\beta_0$  of the output layer, we also refer to (6.4) in the context of GLMs. For grouped penalty terms we



refer to (6.21). The difficulty with this approach is the tuning of the regularization parameter(s)  $\lambda$ : run time is one issue, suitable grouping is another issue, and non-uniqueness of the optimal network a further one that can substantially distort the selection of reasonable regularization parameters.

A more popular method to prevent from over-fitting individual neurons in a FN layer to a certain task are so-called *drop-out layers*. A drop-out layer is an additional layer between FN layers that removes at random during gradient descent training neurons from the network, i.e., in each gradient descent step, any of the earmarked neurons is offset independently from the others with a fixed probability  $\delta \in (0, 1)$ . This random removal will imply that the composite of the remaining neurons needs to be sufficiently well balanced to take over the role of the dropped-out neurons. Therefore, a single neuron cannot be over-trained to a certain task because it needs to be able play several different roles. Drop-out has been introduced by Srivastava et al. [345] and Wager et al. [373].

**Listing 7.6** FN network of depth  $d = 3$  using a drop-out layer, ridge regularization and a normalization layer

---

```

1 Network = list(Design,BrandEmb,RegionEmb) %>%
2   layer_concatenate(name='concat') %>%
3   layer_dense(units=20, activation='tanh', name='FNLayr1') %>%
4   layer_dropout(rate = 0.01) %>%
5   layer_dense(units=15, kernel_regularizer=regularizer_l2(0.0001),
6     activation='tanh', name='FNLayr2') %>%
7   layer_batch_normalization() %>%
8   layer_dense(units=10, activation='tanh', name='FNLayr3') %>%
9   layer_dense(units=1, activation='exponential', name='Network',
10     weights=list(array(0, dim=c(10,1)), array(log(lambda0), dim=c(1))))

```

---

Listing 7.6 gives an example, where we add a drop-out layer with a drop-out probability of  $\delta = 0.01$  after the first FN layer, and in the second FN layer we apply ridge regularization to the weights  $(w_{1,1}^{(2)}, \dots, w_{q_1,q_2}^{(2)})$ , i.e., excluding the intercepts  $w_{0,j}^{(2)}$ ,  $1 \leq j \leq q_2$ . Both the drop-out layer and regularization are only used during the gradient descent fitting, and these network features are disabled during the prediction.

Drop-out is closely related to ridge regularization as the following linear Gaussian regression example shows; this consideration is taken from Section 18.6 of Efron–Hastie [117]. Assume we have a linear regression problem with square loss function

$$\mathfrak{D}(Y, \beta) = \frac{1}{2} \sum_{i=1}^n (Y_i - \langle \beta, x_i \rangle)^2.$$

We assume in this Gaussian case that the observations and the features are standardized, see Sect. 6.2.4. This means that  $\sum_{i=1}^n Y_i = 0$ ,  $\sum_{i=1}^n x_{i,j} = 0$  and

$n^{-1} \sum_{i=1}^n x_{i,j}^2 = 1$ , for all  $1 \leq j \leq q$ . This standardization implies that we can omit the intercept parameter  $\beta_0$  because its MLE is equal to 0.

We introduce i.i.d. drop-out random variables  $I_{i,j}$  for  $1 \leq i \leq n$  and  $1 \leq j \leq q$  with  $(1 - \delta)I_{i,j}$  being Bernoulli distributed with probability  $1 - \delta \in (0, 1)$ . This scaling implies  $\mathbb{E}[I_{i,j}] = 1$ . Using these Bernoulli random variables we modify the above square loss function to

$$\mathfrak{D}_I(\mathbf{Y}, \boldsymbol{\beta}) = \frac{1}{2} \sum_{i=1}^n \left( Y_i - \sum_{j=1}^q \beta_j I_{i,j} x_{i,j} \right)^2,$$

i.e., every individual component  $x_{i,j}$  can drop out independently of the others. Gaussian MLE requires to set the gradient of  $\mathfrak{D}_I(\mathbf{Y}, \boldsymbol{\beta})$  w.r.t.  $\boldsymbol{\beta} \in \mathbb{R}^q$  equal to zero. The average score equation is given by (we average over the drop-out random variables  $I_{i,j}$ )

$$\begin{aligned} \mathbb{E}_\delta [\nabla_{\boldsymbol{\beta}} \mathfrak{D}_I(\mathbf{Y}, \boldsymbol{\beta}) | \mathbf{Y}] &= -\mathfrak{X}^\top \mathbf{Y} + \mathfrak{X}^\top \boldsymbol{\beta} + \frac{\delta}{1 - \delta} \text{diag} \left( \sum_{i=1}^n x_{i,1}^2, \dots, \sum_{i=1}^n x_{i,q}^2 \right) \boldsymbol{\beta} \\ &= -\mathfrak{X}^\top \mathbf{Y} + \mathfrak{X}^\top \boldsymbol{\beta} + \frac{\delta n}{1 - \delta} \boldsymbol{\beta} \stackrel{!}{=} 0, \end{aligned}$$

where we have used the normalization of the columns of the design matrix  $\mathfrak{X} \in \mathbb{R}^{n \times q}$  (we drop the intercept column). This is ridge regression in the linear Gaussian case with a regularization parameter  $\lambda = \delta/(2(1 - \delta)) > 0$  for  $\delta \in (0, 1)$ , see (6.9).

## Normalization Layers

In (7.29) and (7.30) we have discussed that the continuous feature components should be pre-processed so that all components live on the same scale, otherwise the gradient descent fitting may not be efficient. A similar phenomenon may occur with the learned representations  $\mathbf{z}^{(m:1)}(\mathbf{x}_i)$  in the FN layers  $1 \leq m \leq d$ . In particular, this is the case if we choose an unbounded activation function  $\phi$ . For this reason, it can be advantageous to rescale the components  $z_j^{(m:1)}(\mathbf{x}_i)$ ,  $1 \leq j \leq q_m$ , in a given FN layer back to the same scale. To achieve this, a normalization step (7.30) is applied to every neuron  $z_j^{(m:1)}(\mathbf{x}_i)$  over the given cases  $i$  in the considered (mini-)batch. This involves two more parameters (for the empirical mean and the empirical standard deviation) in each neuron of the corresponding FN layer. Note, however, that all these operations are of a linear nature. Therefore, they do not affect the predictive model (i.e., these operations cancel in the scalar products in (7.6)), but they may improve the performance of the gradient descent algorithm.

The code in Listing 7.6 uses a normalization layer on line 6. In our applications, it has not been necessary to use these normalization layers, as it has not led to better

run times in SGD algorithms; note that our networks are not very deep and they use the symmetric and bounded hyperbolic tangent activation function.

### 7.4.2 The Balance Property in Neural Networks

We have seen in Table 7.4 that our FN network outperforms the GLM for claim frequency prediction in terms of a lower out-of-sample loss. We interpret this as follows. Feature engineering has not been done in the most optimal way for the GLM because the FN network finds modeling structure that is not present in the selected GLM. As a consequence, the FN network provides a better generalization to unseen data, i.e., we can better predict new data on a granular level with the FN network. However, having a more precise model on an individual policy level does not necessarily imply that the model also performs better on a global portfolio level. In our example we see that we may have smaller errors on an individual policy level, but these smaller errors do not aggregate to a more precise model in the average portfolio frequency. In our case, we have a misspecification of the average portfolio frequency, see the last column of Table 7.4. This is a major deficiency in insurance pricing because it may result in a misspecification of the overall price level, and this requires a correction. We call this correction *bias regularization*.

#### Simple Bias Regularization

The straightforward correction is to adjust the intercept parameter  $\beta_0 \in \mathbb{R}$  accordingly. That is, compare the empirical mean

$$\bar{\mu} = \frac{\sum_{i=1}^n v_i Y_i}{\sum_{i=1}^n v_i},$$

to the model average of the fitted FN network

$$\hat{\mu} = \frac{\sum_{i=1}^n v_i \mu_{\hat{\boldsymbol{\theta}}}(x_i)}{\sum_{i=1}^n v_i},$$

where  $\hat{\boldsymbol{\theta}} = (\hat{\mathbf{w}}_1^{(1)}, \dots, \hat{\mathbf{w}}_{q_d}^{(d)}, \hat{\boldsymbol{\beta}})^T \in \mathbb{R}^r$  is the learned network parameter from the (early stopped) SGD algorithm. The output of this fitted model reads as

$$x_i \mapsto \mu_{\hat{\boldsymbol{\theta}}}(x_i) = g^{-1} \left\langle \hat{\boldsymbol{\beta}}, \hat{\mathbf{z}}^{(d:1)}(x_i) \right\rangle = g^{-1} \left( \hat{\beta}_0 + \sum_{j=1}^{q_d} \hat{\beta}_j \hat{z}_j^{(d:1)}(x_i) \right),$$

where the hat in  $\widehat{\mathbf{z}}^{(d:1)}$  indicates that we use the estimated weights  $\widehat{\mathbf{w}}_l^{(m)}$ ,  $1 \leq l \leq q_m$ ,  $1 \leq m \leq d$ , in the FN layers. The balance property can be rectified by replacing  $\widehat{\beta}_0$  by the solution  $\widehat{\widehat{\beta}}_0$  of the following identity

$$\sum_{i=1}^n v_i Y_i \stackrel{!}{=} \sum_{i=1}^n v_i g^{-1} \left( \widehat{\widehat{\beta}}_0 + \sum_{j=1}^{q_d} \widehat{\beta}_j \widehat{\mathbf{z}}_j^{(d:1)}(\mathbf{x}_i) \right).$$

Since  $g^{-1}$  is continuous and strictly monotone, there is a unique solution to this requirement supposed that the range of  $g^{-1}$  covers the support of the  $Y_i$ 's. If we work with the log-link  $g(\cdot) = \log(\cdot)$ , this can easily be solved and we obtain

$$\widehat{\widehat{\beta}}_0 = \widehat{\beta}_0 + \log \left( \frac{\bar{\mu}}{\widehat{\mu}} \right).$$

### Sophisticated Bias Regularization Under the Canonical Link Choice

If we work with the canonical link  $g = h = (\kappa')^{-1}$ , we can do better because the MLE of such a GLM automatically provides the balance property, see Corollary 5.7. Choose the SGD learned network parameter  $\widehat{\boldsymbol{\theta}} = (\widehat{\mathbf{w}}_1^{(1)}, \dots, \widehat{\mathbf{w}}_{q_d}^{(d)}, \widehat{\boldsymbol{\beta}})^\top \in \mathbb{R}^r$ . Denote by  $\widehat{\mathbf{z}}^{(d:1)}$  the fitted network architecture that is based on the estimated weights  $\widehat{\mathbf{w}}_1^{(1)}, \dots, \widehat{\mathbf{w}}_{q_d}^{(d)}$ . This allows us to study the learned representations of the raw features  $\mathbf{x}_1, \dots, \mathbf{x}_n$  in the last FN layer. We denote these learned representations by

$$\widehat{\mathbf{z}}_1 = \widehat{\mathbf{z}}^{(d:1)}(\mathbf{x}_1), \dots, \widehat{\mathbf{z}}_n = \widehat{\mathbf{z}}^{(d:1)}(\mathbf{x}_n) \in \{1\} \times \mathbb{R}^{q_d}. \quad (7.32)$$

These learned representations can be used as new features to explain the response  $Y$ . We define the feature engineered design matrix by

$$\widehat{\mathbf{X}} = (\widehat{\mathbf{z}}_1, \dots, \widehat{\mathbf{z}}_n)^\top \in \mathbb{R}^{n \times (q_d+1)}.$$

Based on this new design matrix  $\widehat{\mathbf{X}}$  we can run a classical GLM receiving a unique MLE  $\widehat{\boldsymbol{\beta}}^{\text{MLE}} \in \mathbb{R}^{q_d+1}$  supposed that this design matrix has a full rank  $q_d + 1 \leq n$ , see Proposition 5.1. Since we work with the canonical link, this re-calibrated FN network will automatically satisfy the balance property, and the resulting regression function reads as

$$\mathbf{x} \mapsto \widehat{\mu}(\mathbf{x}) = h^{-1} \left\langle \widehat{\boldsymbol{\beta}}^{\text{MLE}}, \widehat{\mathbf{z}}^{(d:1)}(\mathbf{x}) \right\rangle. \quad (7.33)$$

This is the proposal of Wüthrich [390]. We give some remarks.

*Remarks 7.11*

- This additional MLE step for the output parameter  $\beta \in \mathbb{R}^{qd+1}$  may lead to over-fitting. In that case one might choose a lower dimensional last FN layer. Alternatively, one might explore a more early stopping rule in SGD.
- Wüthrich [390] also explores other bias correction methods like regularization using shrinkage. In combination with regression trees one can achieve averages on pre-defined sub-portfolios. We will not further explore these other approaches because they are less robust and more difficult in the applications.

*Example 7.12 (Balance Property in Networks)* We apply this additional MLE step to the two FN networks of Table 7.4. Note that in these two examples we consider a Poisson model using the canonical link for  $g$ , thus, the resulting adjusted network (7.33) will automatically satisfy the balance property, see Corollary 5.7.

**Listing 7.7** Balance property adjustment (7.33)

---

```

1 glm.formula <- function(nn){
2   string <- "yy ~ X1"
3   if (nn>1){for (ll in 2:nn){ string <- paste(string, "+X",ll, sep="")}}
4   string
5   }
6   #
7   zz <- keras_model(inputs=model$input,
8                     outputs=get_layer(model, 'FNLayer3')$output)
9   xx.learn <- data.frame(zz %>% predict(list(Xlearn, Vlearn)))
10  q3 <- ncol(xx.learn)
11  xx.learn$yy <- Ylearn
12  xx.learn$Exposure <- learn$Exposure
13  #
14  glm1 <- glm(as.formula(glm.formula(q3)),
15             data=xx.learn, offset=log(Exposure), family=poisson())
16
17  #
18  w1 <- get_weights(model)
19  w1[[7]] <- array(glm1$coefficients[2:(q3+1)], dim=c(q3,1))
20  w1[[8]] <- array(glm1$coefficients[1], dim=c(1))
21  set_weights(model, w1)

```

---

In Listing 7.7 we illustrate the necessary code that has to be added to Listings 7.1–7.3. On lines 7–8 of Listing 7.7 we retrieve the learned representations (7.32) which are used as the new features in the Poisson GLM on lines 13–14. The resulting MLE  $\hat{\beta}^{\text{MLE}} \in \mathbb{R}^{qd+1}$  is imputed to the network parameter  $\hat{\theta}$  on lines 17–20. Table 7.5 shows the performance of the resulting bias regularized FN networks.

Firstly, we observe from the last column of Table 7.5 that, indeed, the bias regularization step (7.33) provides the balance property. In general, in-sample losses (have to) decrease because  $\hat{\beta}^{\text{MLE}}$  is (in-sample) more optimal than the early stopped SGD solution  $\hat{\beta}$ . Out-of-sample this leads to a small improvement in the one-

**Table 7.5** Run times, number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model, model Poisson GLM3 of Table 5.5 and the FN network models (with one-hot encoding and embedding layers of dimension  $b = 2$ , respectively), and their bias regularized counterparts

	Run time	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	25.213	25.445	7.36%
Poisson GLM3	15 s	50	24.084	24.102	7.36%
One-hot FN $(q_1, q_2, q_3) = (20, 15, 10)$	51 s	1'306	23.757	23.885	6.96%
Embed FN $(q_1, q_2, q_3) = (20, 15, 10)$	120 s	792	23.694	23.820	7.24%
One-hot FN bias regularized	+4 s	1'306	23.742	23.878	7.36%
Embed FN bias regularized	+4 s	792	23.690	23.824	7.36%

hot encoded variant and a small worsening in the embedding variant, i.e., the latter slightly over-fits in this additional MLE step. However, these differences are comparably small so that we do not further worry about the over-fitting, here. This closes this example. ■

### Auto-Calibration for Bias Regularization

We present another approach of correcting for the potential failure of the balance property. This method does not depend on a particular type of regression model, i.e., it can be applied to any regression model. This proposal goes back to Denuit et al. [97], and it is based on the notion of *auto-calibration* introduced by Patton [297] and Krüger–Ziegel [227]. We first describe auto-calibration and its implications.

**Definition 7.13** The random variable  $Z$  is an auto-calibrated forecast of random variable  $Y$  if  $\mathbb{E}[Y|Z] = Z$ , a.s.

If the response  $Y$  is described by the features  $X = \mathbf{x}$ , we consider the conditional mean of  $Y$ , given  $X$ ,

$$\mu(X) = \mathbb{E}[Y|X].$$

This conditional mean  $\mu(X)$  is an auto-calibrated forecast for the response  $Y$ . Use the tower property and note that  $\sigma(\mu(X)) \subset \sigma(X)$  to receive, a.s.,

$$\mathbb{E}[Y|\mu(X)] = \mathbb{E}[\mathbb{E}[Y|X]|\mu(X)] = \mathbb{E}[\mu(X)|\mu(X)] = \mu(X).$$

For the further understanding of auto-calibration and forecast dominance, we introduce the concept of *convex order*; forecast dominance has been introduced in Definition 4.20.

**Definition 7.14 (Convex Order)** A random variable  $Z_1$  is bigger in convex order than a random variable  $Z_2$ , write  $Z_1 \succeq_{\text{cx}} Z_2$ , if  $\mathbb{E}[\Psi(Z_1)] \geq \mathbb{E}[\Psi(Z_2)]$ , for all convex functions  $\Psi$  for which the expectations exist.

By Strassen's theorem [346],  $Z_1 \succeq_{\text{cx}} Z_2$  if and only if there exist random variables  $Z'_1$  and  $Z'_2$  with  $Z_1 \stackrel{(d)}{=} Z'_1$  and  $Z_2 \stackrel{(d)}{=} Z'_2$  and  $\mathbb{E}[Z'_1|Z'_2] = Z'_2$ , a.s. In particular, the convex order  $Z_1 \succeq_{\text{cx}} Z_2$  implies that  $\text{Var}(Z_1) \geq \text{Var}(Z_2)$  and  $\mathbb{E}[Z_1] = \mathbb{E}[Z_2]$ . The latter follows from Strassen's theorem and the tower property, and the former follows from the latter and the convex order by using the explicit choice  $\Psi(x) = x^2$ . Thus, the random variable  $Z_1$  is more volatile than  $Z_2$ , both having the same mean. The following theorem shows that this additional volatility is a favorable property in terms of forecast dominance under auto-calibration.

**Theorem 7.15 (Krüger–Ziegel [227, Theorem 3.1], Without Proof)** Assume that  $\hat{\mu}_1$  and  $\hat{\mu}_2$  are auto-calibrated forecasts for the random variable  $Y$ . Predictor  $\hat{\mu}_1$  forecast dominates  $\hat{\mu}_2$  if and only if  $\hat{\mu}_1 \succeq_{\text{cx}} \hat{\mu}_2$ .

Recall that forecast dominance of  $\hat{\mu}_1$  over  $\hat{\mu}_2$  was defined as follows, see Definition 4.20,

$$\mathbb{E}[D_\psi(Y, \hat{\mu}_1)] \leq \mathbb{E}[D_\psi(Y, \hat{\mu}_2)],$$

for all Bregman divergences  $D_\psi$ . Strassen's theorem tells us that  $\hat{\mu}_1$  is more volatile than  $\hat{\mu}_2$  (both being auto-calibrated and unbiased for  $\mathbb{E}[Y]$ ) and this additional volatility implies that the former auto-calibrated predictor can better follow  $Y$ . This provides the superior forecast dominance of  $\hat{\mu}_1$  over  $\hat{\mu}_2$ . This relation is most easily understood by the following example. Consider  $(Y, X)$  as above. Assume that the feature  $\tilde{X}$  is a sub-variable of the feature  $X$  by dropping some of the components of  $X$ . Naturally, we have  $\sigma(\tilde{X}) \subset \sigma(X)$ , and both sets of information provide auto-calibrated forecasts

$$\mu(X) = \mathbb{E}[Y|X] \quad \text{and} \quad \mu(\tilde{X}) = \mathbb{E}[Y|\tilde{X}].$$

The tower property and Jensen's inequality give for any convex function  $\Psi$  (subject to existence)

$$\begin{aligned} \mathbb{E}[\Psi(\mu(X))] &= \mathbb{E}[\Psi(\mathbb{E}[Y|X])] = \mathbb{E}[\mathbb{E}[\Psi(\mathbb{E}[Y|X])|\tilde{X}]] \\ &\geq \mathbb{E}[\Psi(\mathbb{E}[\mathbb{E}[Y|X]|\tilde{X}])] = \mathbb{E}[\Psi(\mathbb{E}[Y|\tilde{X}])] = \mathbb{E}[\Psi(\mu(\tilde{X}))]. \end{aligned}$$

Thus, we have  $\mu(X) \succeq_{\text{cx}} \mu(\tilde{X})$  which implies forecast dominance of  $\mu(X)$  over  $\mu(\tilde{X})$ . This makes perfect sense in view of  $\sigma(\tilde{X}) \subset \sigma(X)$ . Basically, this describes the construction of a  $\mathbb{F}$ -martingale using an integrable random variable  $Y$  and a filtration  $\mathbb{F}$  on the underlying probability space  $(\Omega, \mathcal{A}, \mathbb{P})$ . This martingale sequence provides forecast dominance with increasing information sets described by the filtration  $\mathbb{F}$ .

We now turn our attention to the balance property and the unbiasedness of predictors, this follows Denuit et al. [97]. Assume we have any predictor  $\widehat{\mu}(\mathbf{x})$  of  $Y$ , for instance, this can be any FN network predictor  $\mu_{\widehat{\theta}}(\mathbf{x})$  coming from an early stopped SGD algorithm. We define its *balance-corrected* version by

$$\widehat{\mu}_{BC}(\mathbf{x}) = \mathbb{E}[Y | \widehat{\mu}(\mathbf{x})] . \quad (7.34)$$

**Proposition 7.16 (Wüthrich [391, Proposition 4.6], Without Proof)** *The balance-corrected predictor  $\widehat{\mu}_{BC}(X)$  is an auto-calibrated forecast for  $Y$ .*

*Remarks 7.17 (Expected Deviance Generalization Loss)* We return to the decomposition of the expected deviance GL given in Theorem 4.7, but we add the features  $X = \mathbf{x}$ , now. The expected deviance GL of a predictor  $\widehat{\mu}(X)$  under the unit deviance  $\mathfrak{d}$  then reads as

$$\begin{aligned} \mathbb{E}_{\theta} [\mathfrak{d}(Y, \widehat{\mu}(X))] &= \mathbb{E}_{\theta} [\mathfrak{d}(Y, \mu)] \\ &\quad + 2 \left( \mu h(\mu) - \kappa(h(\mu)) - \mathbb{E}_{\theta} [Y h(\widehat{\mu}(X))] + \mathbb{E}_{\theta} [\kappa(h(\widehat{\mu}(X)))] \right), \end{aligned}$$

where  $\mu = \mathbb{E}_{\theta}[Y]$  is the unconditional mean of  $Y$  (averaging also over the feature distribution of  $X$ ). Note that this formula differs from (4.13) because  $Y$  and  $h(\widehat{\mu}(X))$  are no longer independent if we include the features  $X$ . The term  $\mathbb{E}_{\theta}[\mathfrak{d}(Y, \mu)]$  is called the *entropy* which is driven by the stochastic nature of the random variable  $Y$ . This is the irreducible risk if no feature information is available.

In statistical modeling one considers different decompositions of the expected deviance GL, we refer to Fissler et al. [129]. Namely, introducing the features  $X$  we can reduce the expected deviance GL compared to the unconditional mean  $\mu$  in terms of forecast dominance. This allows us to decouple as follows for the prediction  $\mu(X) = \mathbb{E}_{\theta}[Y|X]$

$$\begin{aligned} \mathbb{E}_{\theta} [\mathfrak{d}(Y, \widehat{\mu}(X))] &= \mathbb{E}_{\theta} [\mathfrak{d}(Y, \mu)] - \left( \mathbb{E}_{\theta} [\mathfrak{d}(Y, \mu)] - \mathbb{E}_{\theta} [\mathfrak{d}(Y, \mu(X))] \right) \\ &\quad + \left( \mathbb{E}_{\theta} [\mathfrak{d}(Y, \widehat{\mu}(X))] - \mathbb{E}_{\theta} [\mathfrak{d}(Y, \mu(X))] \right). \end{aligned}$$

This expresses the expected deviance GL of the predictor  $\widehat{\mu}(X)$  as the entropy (first term), the *conditional resolution* (second term) and the *conditional calibration* (third term). The conditional resolution describes the information gain in terms of forecast dominance knowing the feature  $X$ , and the conditional calibration describes how



well we estimate  $\mu(X)$ . The conditional resolution is positive because  $\mu(X) \succeq_{\text{cx}} \mu$  and the unit deviance  $\mathfrak{d}(Y, \cdot)$  is a convex function, see Lemma 2.22. The conditional calibration is also positive, this can be seen by considering the deviance GL, conditional on  $X$ .

We can reformulate this expected deviance GL in terms of the auto-calibration property

$$\begin{aligned} \mathbb{E}_\theta [\mathfrak{d}(Y, \hat{\mu}(X))] &= \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)] - \left( \mathbb{E}_\theta [\mathfrak{d}(Y, \mu)] - \mathbb{E}_\theta [\mathfrak{d}(Y, \hat{\mu}_{\text{BC}}(X))] \right) \\ &\quad + \left( \mathbb{E}_\theta [\mathfrak{d}(Y, \hat{\mu}(X))] - \mathbb{E}_\theta [\mathfrak{d}(Y, \hat{\mu}_{\text{BC}}(X))] \right). \end{aligned}$$

The first term is the entropy, the second term is called the *auto-resolution* and the third term describes the *auto-calibration*. If we have an auto-calibrated forecast  $\hat{\mu}(X)$  then the last term vanishes because it is equal to its balance-corrected version  $\hat{\mu}_{\text{BC}}(X)$ . Again these two latter terms are positive, for the auto-calibration this can be seen by considering the deviance GL, conditioned on  $\hat{\mu}(X)$ .

To rectify the balance property we directly focus on (7.34), and we *estimate* this conditional expectation. That is, the balance correction can be achieved by an additional regression step directly estimating the balance-corrected version  $\hat{\mu}_{\text{BC}}(x)$  in (7.34). This additional regression step differs from (7.33) as it does not use the learned representations  $\hat{z}^{(d:1)}(x)$  in the last FN layer (7.32), but it uses the learned representations in the output layer. That is, consider the learned features

$$\hat{z}_1^\star = (1, \mu_{\hat{\theta}}(x_1))^\top, \dots, \hat{z}_n^\star = (1, \mu_{\hat{\theta}}(x_n))^\top \in \{1\} \times \mathbb{R},$$

and perform an additional linear regression step for the response  $Y$  using the design matrix

$$\hat{\mathbf{x}} = (\hat{z}_1^\star, \dots, \hat{z}_n^\star)^\top \in \mathbb{R}^{n \times 2}.$$

This additional linear regression step gives us an estimate

$$\hat{\beta} = \left( \hat{\mathbf{x}}^\top V \hat{\mathbf{x}} \right)^{-1} \hat{\mathbf{x}}^\top V Y \in \mathbb{R}^2, \quad (7.35)$$

with diagonal weight matrix  $V = \text{diag}(v_i)_{1 \leq i \leq n}$ . The balance property is then restored by estimating the balance-corrected means  $\hat{\mu}_{\text{BC}}(x_i)$  by

$$\hat{\mu}_{\text{BC}}(x_i) = \hat{\beta}_0 + \hat{\beta}_1 \mu_{\hat{\theta}}(x_i), \quad (7.36)$$

for  $1 \leq i \leq n$ . Note that this can be done for any regression model since we do not rely on the network architecture in this step.

*Remarks 7.18*

- Balance correction (7.36) may lead to some conflict in range if the dual (mean) parameter space  $\mathcal{M}$  is (one-sided) bounded. Moreover, it does not consider the deviance loss of the response  $Y$ , but it rather underlies a Gaussian model by using the weighted square loss function for finding (the Gaussian MLE)  $\hat{\beta} \in \mathbb{R}^2$ . Alternatively, we could consider the canonical link  $h$  that belongs to the chosen EDF. This then allows us to study the regression problem on the canonical scale by setting for the learned representations

$$\hat{z}_1^\theta = (1, h(\mu_{\hat{\theta}}(\mathbf{x}_1)))^\top, \dots, \hat{z}_n^\theta = (1, h(\mu_{\hat{\theta}}(\mathbf{x}_n)))^\top \in \{1\} \times \Theta. \quad (7.37)$$

The latter motivates the consideration of a GLM under the chosen EDF

$$\mathbf{x}_i \mapsto h(\hat{\mu}_{\text{BC}}(\mathbf{x}_i)) = \langle \beta, \hat{z}_i^\theta \rangle = \beta_0 + \beta_1 h(\mu_{\hat{\theta}}(\mathbf{x}_i)), \quad (7.38)$$

for regression parameter  $\beta \in \mathbb{R}^2$ . The choice of the canonical link and the inclusion of an intercept will provide the balance property when estimating  $\beta$  with MLE, see Corollary 5.7. If the mean estimates  $\mu_{\hat{\theta}}(\mathbf{x}_i)$  involve the canonical link  $h$ , (7.38) reads as

$$\mathbf{x}_i \mapsto h(\hat{\mu}_{\text{BC}}(\mathbf{x}_i)) = \langle \beta, \hat{z}_i^\theta \rangle = \beta_0 + \beta_1 \left\langle \hat{\beta}, \hat{z}^{(d:1)}(\mathbf{x}_i) \right\rangle,$$

the latter scalar product is the output activation received from the FN network. From this we see that the estimated balance-corrected calibration on the canonical scale will give us a non-optimal (in-sample) estimation step compared to (7.33), if we work with the canonical link  $h$ .

- Denuit et al. [97] give a proposal to break down the global balance to a local version using a suitable kernel function, this will be further discussed in the next Example 7.19.

*Example 7.19 (Auto-calibration in Networks)* We apply this additional auto-calibration step (7.34) to the FN network with embedding layers that does not satisfy the balance property, i.e., having an average frequency of 7.24% < 7.36%, see Tables 7.4 and 7.5. We start by analyzing the auto-calibration property (7.34) of this network predictor  $v\mu_{\hat{\theta}}(\mathbf{x})$  by studying an empirical version of

$$z \mapsto v\hat{\mu}_{\text{BC}}(\mathbf{x}) = \mathbb{E} [vY \mid v\mu_{\hat{\theta}}(\mathbf{x}) = z]. \quad (7.39)$$

This empirical version is obtained from the R library `locfit` [254] that allows us to consider a local polynomial regression fit of degree `deg=2`, and we use a nearest neighbor fraction of `alpha=0.05`, the code is provided in Listing 7.8. We use the exposure  $v$  scaled version in (7.39) since the balance property should hold on that scale, see Corollary 5.7. The claim counts are given by  $N = vY$ , and the exposure

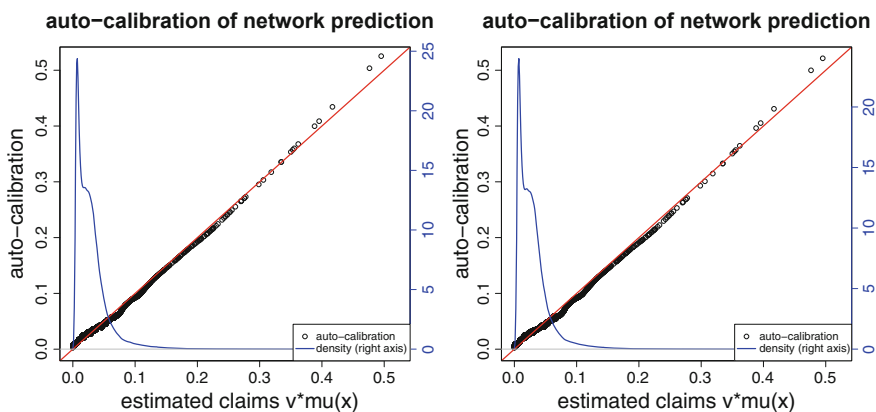
$v$  is integrated as an offset into the FN network regression function, see line 20 of Listing 7.4.

**Listing 7.8** Empirical auto-calibration using the R library `locfit` [254]

```
1 z <- learn$pred
2 mu.BC <- predict(locfit(learn$N ~ learn$pred, alpha=0.05, deg=2), newdata=z)
```

Figure 7.12 (lhs) shows the empirical auto-calibration of (7.39) using the R code of Listing 7.8. If the auto-calibration would hold exactly, then the black dots should lie on the red diagonal line. We observe a very good match, which indicates that the auto-calibration property holds quite accurately for our network predictor  $(v, \mathbf{x}) \mapsto v\mu_{\hat{\theta}}(\mathbf{x})$ . For very small expectations  $\mathbb{E}_{\theta(\mathbf{x})}[N]$  we slightly underestimate, and for bigger expectations we slightly overestimate. The blue line shows the empirical density of the predictors  $v_i\mu_{\hat{\theta}}(\mathbf{x}_i)$ ,  $1 \leq i \leq n$ , highlighting heavy-tailedness and that the underestimation in the right tail will not substantially contribute to the balance property as these are only very few insurance policies.

We explore the Gaussian balance correction (7.35) considering a linear regression model with weighted square loss function. We receive the estimate  $\hat{\beta} = (9 \cdot 10^{-4}, 1.005)^\top$ , thus,  $\mu_{\hat{\theta}}(\mathbf{x})$  only gets very gently distorted, see (7.36). The results of this balance-corrected version  $\hat{\mu}_{BC}(\mathbf{x})$  are given on line ‘embed FN Gauss balance-corrected’ in Table 7.6. We observe that this approach is rather competitive leading to a slightly better model (out-of-sample). Figure 7.12 (rhs) shows the resulting (empirical) auto-calibration plot which is still not fully in line with Proposition 7.16; this empirical plot may be distorted by the exposures, by the fact that it is an



**Fig. 7.12** (lhs) Empirical auto-calibration (7.39), the blue line shows the empirical density of the predictors  $v_i\mu_{\hat{\theta}}(\mathbf{x}_i)$ ,  $1 \leq i \leq n$ ; (rhs) balance-corrected version using the weighted Gaussian correction (7.35)

**Table 7.6** Run times, number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model, model Poisson GLM3 of Table 5.5, the FN network model (with embedding layers of dimension  $b = 2$ ), and their bias regularized and balance-corrected counterparts, the local correction uses a GAM with 2.6 degrees of freedom in the cubic spline part

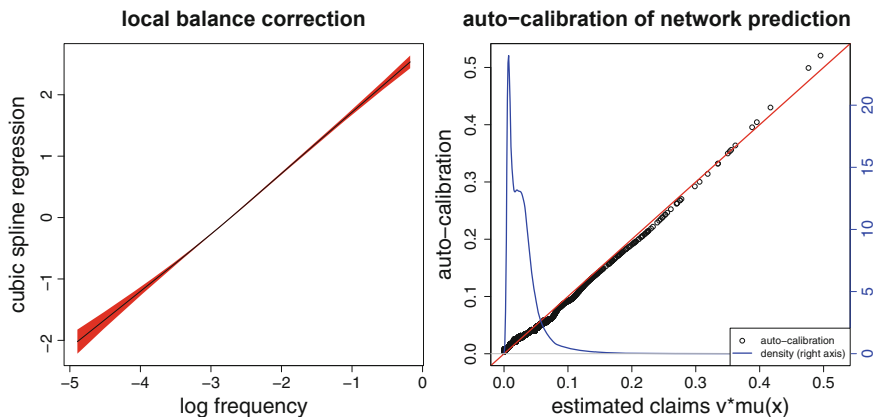
	Run time	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	25.213	25.445	7.36%
Poisson GLM3	15 s	50	24.084	24.102	7.36%
Embed FN $(q_1, q_2, q_3) = (20, 15, 10)$	120 s	792	23.694	23.820	7.24%
Embed FN bias regularized	+4 s	792	23.690	23.824	7.36%
Embed FN Gauss balance-corrected	–	792 + 2	23.692	23.819	7.36%
Embed FN locally balance-corrected	–	792 + 3.6	23.692	23.818	7.36%

empirical plot fitted with `locfit`, and by fact that a linear Gaussian correction estimate may not be fully suitable.

Denuit et al. [97] propose a local balance correction that is very much in the spirit of the local polynomial regression fit with `locfit`. However, when using `locfit` we did not pay any attention to the balance property. Therefore, we proceed slightly differently, here. In formula (7.37) we give the network predictors on the canonical scale. This equips us with the data  $(Y_i, v_i, \hat{\mathbf{z}}_i^\theta)_{1 \leq i \leq n}$ . To perform a local balance correction we fit a generalized additive model (GAM) to this data, using the canonical link, the Poisson deviance loss function, the observations  $Y_i$ , the exposures  $v_i$  and the feature information  $\hat{\mathbf{z}}_i^\theta$ ; for GAMs we refer to Hastie–Tibshirani [181, 182], Wood [384] and Chapter 3 in Wüthrich–Buser [392], in particular, we proceed as in Example 3.4 of the latter reference.

The GAM regression fit on the canonical scale is illustrated in Fig. 7.13 (lhs). We essentially receive a straight line which says that the auto-calibration property is already well satisfied by the FN network predictor  $\mu_{\hat{\mathbf{z}}}$ . In fact, it is not completely a straight line, but GCV provides an optimal model with 2.6 effective degrees of freedom in the natural cubic spline part. This local (GAM) balance correction leads to another small model improvement (out-of-sample), see last line of Table 7.6.

**Conclusion** The balance property adjustment and the bias regularization are crucial in ensuring that the predictive model is on the right (price) level. We have presented three sophisticated methods of balance property adjustments: the additional GLM step under the canonical link choice (7.33), the model-free global Gaussian correction (7.35)–(7.36), and the local balance correction using a GAM under the canonical link choice. In our example, the results of the three different approaches are rather similar. In the sequel, we use the additional GLM step solution (7.33), the reason being that under this approach we can rely on one single regression model that directly predicts the claims. The other two approaches need two steps to get the predictions, which requires the storage of two models. ■



**Fig. 7.13** (lhs) GAM fit on the canonical scale having 2.6 effective degrees of freedom (red shows the estimated confidence bounds); (rhs) balance-corrected version using the local GAM correction

### 7.4.3 Boosting Regression Models with Network Features

From Table 7.5 we conclude that the FN networks find systematic structure in the data that is not present in model Poisson GLM3, thus, the feature engineering for the GLM can be improved. Unfortunately, FN networks neither directly build on GLMs nor do they highlight the weaknesses of GLMs. In this section we discuss a proposal presented in Wüthrich–Merz [394] and Schelldorfer–Wüthrich [329] of combining two regression approaches. We are going to boost a GLM with FN network features. Typically, boosting is applied within the framework of regression trees. It goes back to the work of Valiant [362], Kearns–Valiant [209, 210], Schapire [328], Freund [139] and Freund–Schapire [140]. The idea behind boosting is to analyze the residuals of a given regression model with a second regression model to see whether this second regression model can still find systematic effects in the residuals which have not been discovered by the first one.

We start from the GLM studied in Chap. 5, and we boost this GLM with a FN network. Assume that both regression models act on the same feature space  $\mathcal{X} \subset \{1\} \times \mathbb{R}^{q_0}$ . The GLM provides a regression function for link function  $g$  and GLM parameter  $\beta^{\text{GLM}} \in \mathbb{R}^{q_0+1}$

$$x \mapsto \mu^{\text{GLM}}(x) = g^{-1} \left( \beta^{\text{GLM}}, x \right).$$

Recall that this GLM can be interpreted as a FN network of depth 0, see Remarks 7.2. Next, we choose a FN network of depth  $d \geq 1$  with the same link

function  $g$  as the GLM

$$\mathbf{x} \mapsto \mu^{\text{FN}}(\mathbf{x}) = g^{-1} \left\langle \boldsymbol{\beta}^{\text{FN}}, \mathbf{z}^{(d:1)}(\mathbf{x}) \right\rangle,$$

having a network parameter  $\boldsymbol{\vartheta} = (\mathbf{w}_1^{(1)}, \dots, \mathbf{w}_{q_d}^{(d)}, \boldsymbol{\beta}^{\text{FN}})^\top \in \mathbb{R}^r$ . In particular, we have the FN output parameter  $\boldsymbol{\beta}^{\text{FN}} \in \mathbb{R}^{q_d+1}$ , we refer to Fig. 7.2.

We blend these two regression models by combining their regression functions

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = g^{-1} \left\{ \left\langle \boldsymbol{\beta}^{\text{GLM}}, \mathbf{x} \right\rangle + \left\langle \boldsymbol{\beta}^{\text{FN}}, \mathbf{z}^{(d:1)}(\mathbf{x}) \right\rangle \right\}, \quad (7.40)$$

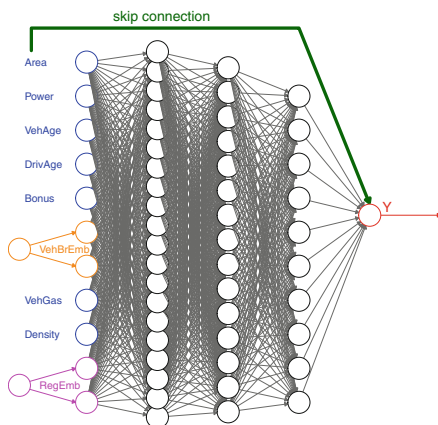
with parameter  $\Phi = (\boldsymbol{\beta}^{\text{GLM}}, \boldsymbol{\vartheta})^\top = (\boldsymbol{\beta}^{\text{GLM}}, \mathbf{w}_1^{(1)}, \dots, \mathbf{w}_{q_d}^{(d)}, \boldsymbol{\beta}^{\text{FN}})^\top \in \mathbb{R}^{q_0+1+r}$ .

An example is provided in Fig. 7.14. It shows the FN network using embedding layers for the categorical variables, see also Fig. 7.9 (rhs), and we add a GLM (in green color) that directly links the input  $\mathbf{x}$  to the response variable. In machine learning this green connection is called a *skip connection* because it skips the FN layers.

#### Remarks 7.20

- Skip connections are a popular tool in network modeling, and they can be applied to any FN layers, i.e., a skip connection can, for instance, be added to skip the first FN layer. There are two benefits from skip connections. Firstly, they allow for more modeling flexibility, in (7.40) we directly combine a linear function

**Fig. 7.14** Illustration of the combined regression function (7.40) using a GLM (in a skip connection) and a FN network



(coming from the GLM) with a non-linear one (coming from the FN network). This has the flavor of a Taylor expansion to combine terms of different orders. Secondly, skip connections can also be beneficial for gradient descent fitting because the inputs have a more direct link to the outputs, and the network only builds the functional form around the function in the skip connection.

- There are numerous variants of (7.40). A straightforward one is to choose a weight  $\alpha \in (0, 1)$  and consider the regression function

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = g^{-1} \left\{ \alpha \langle \boldsymbol{\beta}^{\text{GLM}}, \mathbf{x} \rangle + (1 - \alpha) \langle \boldsymbol{\beta}^{\text{FN}}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle \right\}. \quad (7.41)$$

The weight  $\alpha$  can be interpreted as the credibility assigned to the GLM.

- Regression function (7.40) considers two intercepts  $\beta_0^{\text{GLM}}$  and  $\beta_0^{\text{FN}}$ . If we do not consider the credibility version (7.41), one of the two intercepts is redundant.
- This approach also allows us to learn systematic effects across different insurance portfolios. If we have three insurance portfolios living on the same feature space and if  $\chi \in \{1, 2, 3\}$  indicates which insurance portfolio we consider, we can modify the regression function (7.40) to

$$(\mathbf{x}, \chi) \mapsto \mu(\mathbf{x}, \chi) = g^{-1} \left\{ \sum_{j=1}^3 \langle \boldsymbol{\beta}_j^{\text{GLM}}, \mathbf{x} \rangle \mathbb{1}_{\{\chi=j\}} + \langle \boldsymbol{\beta}^{\text{FN}}, \mathbf{z}^{(d:1)}(\mathbf{x}, \chi) \rangle \right\}.$$

The indicator  $\mathbb{1}_{\{\chi=j\}}$  chooses the GLM that belongs to the corresponding insurance portfolio  $\chi \in \{1, 2, 3\}$  with the (individual) GLM parameter  $\boldsymbol{\beta}_\chi^{\text{GLM}}$ . The FN network term makes them related, i.e., the GLMs of the different insurance portfolios interact (jointly learn) via the FN network module. This is the approach used in Gabrielli et al. [149] to improve the chain-ladder reserving method by learning across different claims reserving triangles.

The regression function (7.40) gives the structural form of the combined regression model, but there is a second important ingredient proposed by Wüthrich–Merz [394]. Namely, the gradient descent algorithm (7.15) for model fitting can be started in an initial network parameter  $\Phi^{(0)} \in \mathbb{R}^{q_0+1+r}$  that corresponds to the MLE of the GLM. Denote by  $\hat{\boldsymbol{\beta}}^{\text{GLM}}$  the MLE of the GLM part, only.

Choose the initial value of the gradient descent algorithm for the fitting of the combined regression model (7.40)

$$\Phi^{(0)} = \left( \hat{\boldsymbol{\beta}}^{\text{GLM}}, \mathbf{w}_1^{(1)}, \dots, \mathbf{w}_{q_d}^{(d)}, \boldsymbol{\beta}^{\text{FN}} \equiv \mathbf{0} \right)^\top \in \mathbb{R}^{q_0+1+r}, \quad (7.42)$$

that is, initially, no signals traverse the FN network part because we set  $\boldsymbol{\beta}^{\text{FN}} \equiv \mathbf{0}$ .

*Remarks 7.21*

- Using the initialization (7.42), the gradient descent algorithm starts exactly in the optimal GLM. The algorithm then tries to improve this GLM w.r.t. the given loss function using the additional FN network features. If the loss substantially reduces during the gradient descent training, the GLM misses systematic structure and it can be improved, otherwise the GLM is already good (enough).
- We can declare the MLE  $\hat{\beta}^{\text{GLM}}$  to be *non-trainable*. In that case the original GLM always remains in the combined regression model and it acts as an offset. If we declare the MLE  $\hat{\beta}^{\text{GLM}}$  to be non-trainable, we could choose a trainable credibility weight  $\alpha \in (0, 1)$ , see (7.41), which gradually reduces the influence of the GLM (if necessary).

Implementation of the general combined regression model (7.40) can be a bit cumbersome, see Listing 4 in Gabrielli et al. [149], but things can substantially be simplified by declaring the GLM part in (7.40) as being non-trainable, i.e., estimating  $\beta^{\text{GLM}}$  by  $\hat{\beta}^{\text{GLM}}$  in the GLM, and then freeze this parameter. In view of (7.40) this simply means that we add an offset  $o_i = \langle \hat{\beta}^{\text{GLM}}, \mathbf{x}_i \rangle$  to the FN network that is treated as a prior difference between the different data points, we refer to Sect. 5.2.3.

*Example 7.22 (Combined GLM and FN Network)* We revisit the French MTPL claim frequency GLM of Sect. 5.3.4, and we boost model Poisson GLM3 with FN network features. For the FN architecture we use the structure depicted in Fig. 7.14, i.e., a FN network of depth  $d = 3$  having  $(q_1, q_2, q_3) = (20, 15, 10)$  neurons, and using embedding layers of dimension  $b = 2$  for the categorical feature components. Moreover, we declare the GLM part to be non-trainable which allows us to use the GLM as an offset in the FN network. Moreover, we apply bias regularization (7.33) to receive the balance property.

The results are presented in Table 7.7. A first observation is that using model Poisson GLM3 as an offset reduces the run time of gradient descent fitting because we start the algorithm already in a reasonable model. Secondly, as expected, the

**Table 7.7** Run times, number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model, model Poisson GLM3 of Table 5.5, the FN network model (with embedding layers of dimension  $b = 2$ ), and the combined regression model GLM3+FN, see (7.40)

	Run time	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	25.213	25.445	7.36%
Poisson GLM3	15 s	50	24.084	24.102	7.36%
Embed FN $(q_1, q_2, q_3) = (20, 15, 10)$	120 s	792	23.694	23.820	7.24%
Embed FN bias regularized	+4 s	792	23.690	23.824	7.36%
Combined GLM+FN (20, 15, 10)	+53 s	50 + 792	23.772	23.834	7.24%
Combined GLM+FN bias regularized	+4 s	50 + 792	23.765	23.830	7.36%



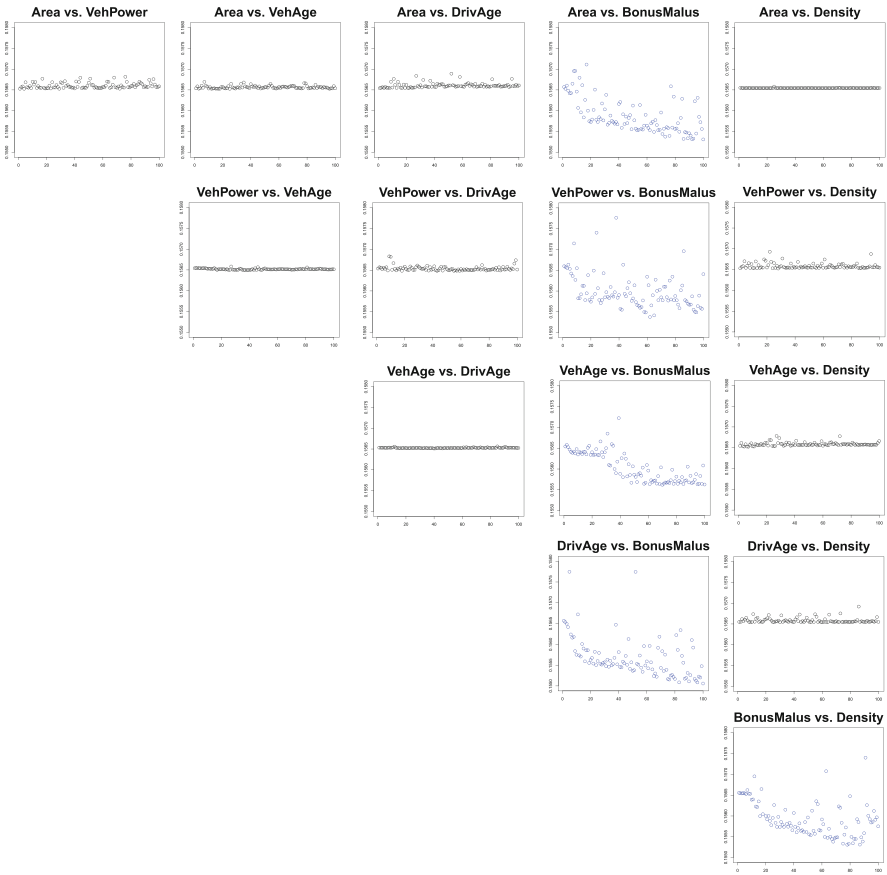
FN features decrease the loss of model Poisson GLM3, this indicates that there are systematic effects that are not captured by the GLM. The final combined and regularized model has roughly the same out-of-sample loss as the corresponding FN network, showing that this approach can be beneficial in run times, and the predictive power is similar to a pure FN network. ■

*Example 7.23 (Improving Model Poisson GLM3)* In this example we would like to explore the deficiencies of model Poisson GLM3 by boosting it with FN network features. We do this in a systematic way by only considering two (continuous) features components at a time in the FN network. That is, we consider the combined approach (7.40) with initialization (7.42), but as feature information for the network part, we only consider two components at a time. For instance, we start with the features  $(1, \text{Area}, \text{VehPower}) \in \{1\} \times \mathbb{R}^2$  for the network part, and the remaining feature information is ignored in this step. This way we can test whether the marginal modeling of Area and VehPower is suitable in model Poisson GLM3, and whether a pairwise interaction in these two components is missing. We train this FN network starting from model Poisson GLM3 (and keeping this GLM part frozen). The decrease in the out-of-sample loss during the gradient descent training is shown in Fig. 7.15 (top-left). We observe that the loss remains rather constant over 100 training epochs. This tells us that the pair (Area, VehPower) is appropriately considered in model Poisson GLM3.

Figure 7.15 gives all pairwise plots of the continuous feature components Area, VehPower, VehAge, DrivAge, BonusMalus, Density, the scale on the y-axis is identical in all plots. We observe that only the plots including the variable BonusMalus provide a bigger decrease in loss (in blue color in the colored version). This indicates that mainly this feature component is not modeled optimally in model Poisson GLM3, because boosting with a FN network finds systematic structure here that improves the loss of model Poisson GLM3. In model Poisson GLM3, the variable BonusMalus has been modeled log-linearly with an interaction term with DrivAge and  $(\text{DrivAge})^2$ , see (5.35). Table 7.8 shows the result if we add a FN network feature (7.40) for the pair (DrivAge, BonusMalus) to model Poisson GLM3. Indeed, we see that the resulting combined GLM-FN network model has the same GL as the full FN network approach. Thus, we conclude that model Poisson GLM3 performs fairly well and only the modeling of the pair (DrivAge, BonusMalus) should be improved. ■

#### 7.4.4 Network Ensemble Learning

Ensemble learning is a popular way of expressing that one takes an average over different predictors. There are many established methods that belong to the family of ensemble learning, e.g., there is **bootstrap aggregating** (called *bagging*) introduced by Breiman [51], there are random forests, and there is boosting. Random forests



**Fig. 7.15** Exploring all pairwise interactions: out-of-sample losses over 100 gradient descent epochs for all pairs of the continuous feature components Area, VehPower, VehAge, DrivAge, BonusMalus, Density (the scale on the y-axis is identical in all plots)

and boosting are mainly based on classification and regression trees (CARTs) and they belong to the most powerful machine learning methods for tabular data. These methods combine a family of predictors to a more powerful predictor. The present section is inspired by the bagging method of Breiman [51], and we perform **network aggregating** (called *nagging*).

### Stochastic Gradient Descent Fitting of Networks

We have described that network calibration involves several elements of randomness. This in combination with early stopping leads to the non-uniqueness of reasonably good networks for prediction and pricing. We have discussed this based

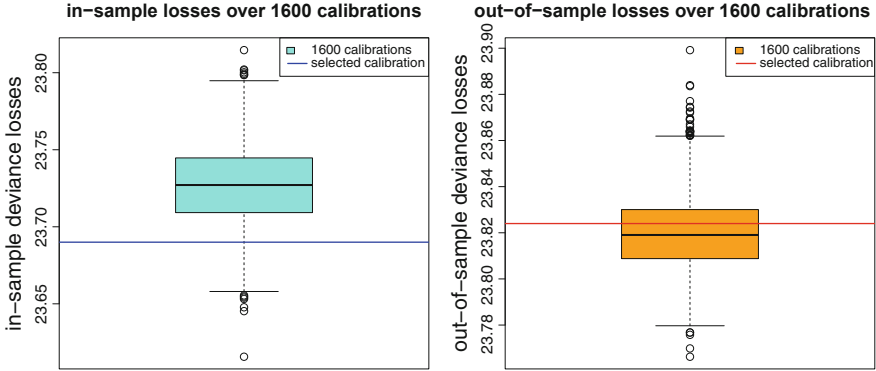
**Table 7.8** Run times, number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model, model Poisson GLM3 of Table 5.5, model Poisson GLM3 with additional FN features for (DrivAge, BonusMalus), the FN network model (with embedding layers of dimension  $b = 2$ ), and the combined regression model GLM3+FN, see (7.40)

	Run time	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	25.213	25.445	7.36%
Poisson GLM3	15 s	50	24.084	24.102	7.36%
GLM3 +FN(DrivAge, BonusMalus)	–	50 + 792	23.804	23.805	7.36%
Embed FN bias regularized	124 s	792	23.690	23.824	7.36%
Combined GLM+FN bias regularized	72 s	50 + 792	23.765	23.830	7.36%

on Fig. 7.5, namely, for a given network architecture we have a continuum of comparably good models (w.r.t. the chosen objective function) that lie in the green area of Fig. 7.5. One SGD calibration picks one specific model from this green area, we also refer to Remarks 7.9. Of course, this is very unsatisfactory in insurance pricing because it implies that the selection of a price for an insurance policy has a substantial element of subjectivity (that cannot be explained to the customer). Naturally, we would like to combine models in the green area of Fig. 7.5, for instance, by performing some sort of integration over the models in the green area. Intuitively, this should lead to a very powerful predictive model because it diversifies the weaknesses of each individual model. This is exactly what we discuss in this section. Before doing so, we would first like to understand the different single calibrations of a given network architecture.

We consider the MTPL data of Example 7.12. We model this data with a Poisson FN network using embedding layers for the categorical features and using bias regularization (7.33) to guarantee the balance property to hold. For the FN network architecture we choose depth  $d = 3$  with  $(q_1, q_2, q_3) = (20, 15, 10)$  FN neurons; this setup gives us the results on the last line of Table 7.5. We now repeat this procedure  $M = 1'600$  times, using exactly the same FN network architecture, the same early stopping strategy, the same SGD method and the same batch size. We only change the seeds of the starting point  $\boldsymbol{\theta}^{(0)} \in \mathbb{R}^r$  of the SGD algorithm, the partitioning of the learning data  $\mathcal{L}$  into training data  $\mathcal{U}$  and validation data  $\mathcal{V}$ , see Fig. 7.7, and the partitioning of the training data into the (mini-)batches.

The resulting 1'600 in-sample and out-of-sample deviance losses are presented in Fig. 7.16. We observe a considerable variation in these figures. The in-sample losses vary between 23.616 and 23.815 (mean 23.728), and the corresponding out-of-sample loss between 23.766 and 23.899 (mean 23.819), units are in  $10^{-2}$ ; note that all network calibrations are bias regularized. The in-sample loss is an average over  $n = 610'206$  (individual) unit deviance losses, and the out-of-sample an average over  $T = 67'801$  unit deviance losses, see also Definition 4.24. Therefore, we expect an even much bigger variation on individual insurance policies. We are going to analyze this in more detail in this section.



**Fig. 7.16** Boxplots over 1'600 network calibrations only differing in the seeds for the SGD algorithm and the partitioning of the learning data: (lhs) in-sample losses on  $\mathcal{L}$  and (rhs) out-of-sample losses on  $\mathcal{T}$ , the horizontal lines show the calibration chosen in Table 7.5; units are in  $10^{-2}$

Before doing so, we would like to understand whether there is some dependence between the in-sample and the out-of-sample losses over the  $M = 1'600$  runs of the SGD algorithm with different seeds. In Fig. 7.17 we provide a scatter plot of the out-of-sample losses vs. the in-sample losses. This plot is complemented by a cubic spline regression (in orange color). From this plot we conclude that the models with very small in-sample losses tend to over-fit, and the models with large in-sample losses tend to under-fit (always using the same early stopping rule). In view of these results we conclude that the chosen early stopping rule is sensible because on average it tends to provide the model with the smallest out-of-sample loss on  $\mathcal{T}$ . Recall that we do not use  $\mathcal{T}$  during the SGD fitting, but only the learning data  $\mathcal{L}$  that is split into the training data  $\mathcal{U}$  and the validation data  $\mathcal{V}$  for exercising the early stopping, see Fig. 7.7.

**Fig. 7.17** Scatter plot of out-of-sample losses vs. in-sample losses for different seeds, the orange line gives a fitted cubic spline, and the cyan lines show the empirical means; units are in  $10^{-2}$



Next, we study the estimated prices on the test data (out-of-sample)

$$\mathcal{T} = \left\{ (Y_t^\dagger = N_t^\dagger / v_t^\dagger, \mathbf{x}_t^\dagger, v_t^\dagger) : t = 1, \dots, T = 67'801 \right\}.$$

For each run of the SGD algorithm we receive a different (early stopped) network parameter estimate  $\hat{\boldsymbol{\theta}}^m \in \mathbb{R}^r$ ,  $1 \leq m \leq M = 1'600$ . Using these parameter estimates we receive the estimated network regression functions, for  $1 \leq m \leq M$ ,

$$\mathbf{x} \mapsto \hat{\mu}^m(\mathbf{x}) = \mu_{\hat{\boldsymbol{\theta}}^m}(\mathbf{x}),$$

using the FN network of Listing 7.4 with network parameter  $\hat{\boldsymbol{\theta}}^m$ . Thus, for the out-of-sample policies  $1 \leq t \leq T$  we receive the expected frequencies

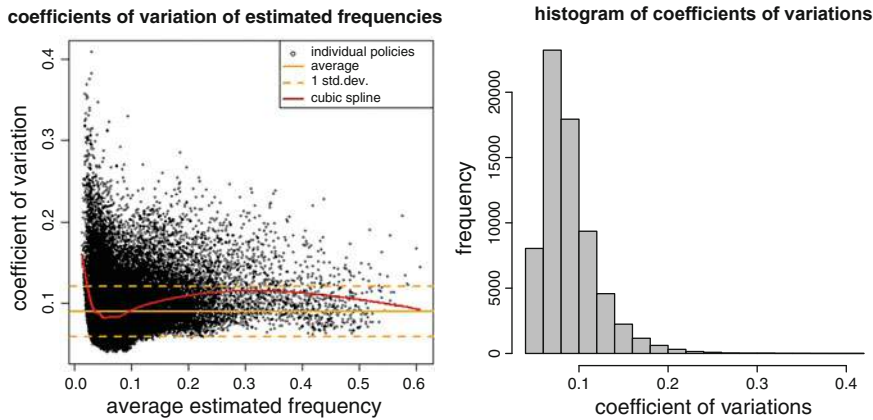
$$\mathbf{x}_t^\dagger \mapsto \hat{\mu}_t^m = \hat{\mu}^m(\mathbf{x}_t^\dagger) = \mu_{\hat{\boldsymbol{\theta}}^m}(\mathbf{x}_t^\dagger).$$

Since we choose the seeds of the SGD runs *at random* we may (and will) assume that we have independence between the prices  $(\hat{\mu}_t^m)_{t \in \mathcal{T}}$  of the different runs  $1 \leq m \leq M$  of the SGD algorithm. This allows us to estimate the average price and the coefficient of variation of these prices of a fixed insurance policy  $t$  over the different SGD runs

$$\bar{\mu}_t^{(1:M)} = \frac{1}{M} \sum_{m=1}^M \hat{\mu}_t^m \quad \text{and} \quad \text{Vco}_t = \frac{1}{\bar{\mu}_t^{(1:M)}} \sqrt{\frac{1}{M-1} \sum_{m=1}^M \left( \hat{\mu}_t^m - \bar{\mu}_t^{(1:M)} \right)^2}. \quad (7.43)$$

These (out-of-sample) coefficients of variation are illustrated in Fig. 7.18. We observe a considerable variation on some policies. The average coefficient of variation is roughly 10% (orange horizontal line, lhs). The maximal coefficient of variation is about 40%, thus, for this policy the individual prices  $\hat{\mu}_t^m$  of the different SGD runs  $1 \leq m \leq M$  fluctuate considerably around  $\bar{\mu}_t^{(1:M)}$ . This now explains why we choose  $M = 1'600$  SGD runs, namely, the averaging in (7.43) reduces the coefficient of variation on this policy to  $40\% / \sqrt{M} = 40\% / 40 = 1\%$ , note that we have independence between the different SGD runs. Thus, by averaging we receive an acceptable influence of the variation of the individual SGD fittings.

Listing 7.9 shows the 10 policies (out-of-sample) with the largest coefficients of variations  $\text{Vco}_t$ . These policies have in common that they belong to the lowest BonusMalus level, the drivers are very young, the cars are comparably old and they have a bigger vehicle power. From a practical point of view we should doubt these policies, since the information provided may not be correct. New drivers (at the age of 18) typically enter a bonus-malus scheme at level 100, and only after several accident-free years these drivers can reach a bonus-malus level of 50. Thus, policies as in Listing 7.9 should not exist, and our pricing framework has difficulties to (correctly) handle them. In practice, this needs further investigation because, obviously, there is a data issue, here.



**Fig. 7.18** Out-of-sample coefficients of variations  $Vco_t$  on an individual policy level  $1 \leq t \leq T$  over the 1'600 calibrations (lhs) scatter plot against the average estimated frequencies  $\bar{\mu}_t^{(1:M)}$  and (rhs) resulting histogram

**Listing 7.9** The 10 policies (out-of-sample) with the largest coefficients of variation

	Area	VehPower	VehAge	DrivAge	BonusMalus	VehBrand	VehGas	Region	vco
1									
2	D	8	16	18	50	B11 Regular	R53	0.4089006	
3	D	9	17	20	50	B11 Regular	R24	0.3827665	
4	C	8	11	18	50	B5 Regular	R24	0.3762306	
5	C	9	18	18	50	B5 Regular	R24	0.3697370	
6	C	7	17	18	50	B1 Regular	R24	0.3579979	
7	C	9	19	19	50	B5 Regular	R24	0.3554879	
8	C	6	15	20	50	B1 Regular	R93	0.3528679	
9	C	7	14	19	50	B1 Regular	R53	0.3518279	
10	A	11	20	50	50	B13 Regular	R74	0.3442184	
11	D	5	14	18	50	B3 Diesel	R24	0.3403783	

Nagging Predictor

The previously observed variations of the prices motivate to average over the different models (network calibrations). This brings us to bagging introduced by Breiman [51]. Bagging is based on averaging/aggregating over several ‘independent’ predictions; this is done in three steps. In a first step, a model is fitted to the data  $\mathcal{L}$ . In a second step, independent bootstrap samples  $\mathcal{L}^{*(m)}$  are generated from this fitted model; the independence has to be understood in a conditional sense, namely, the different bootstrap samples  $\mathcal{L}^{*(m)}$  are independent in  $m$ , given the data  $\mathcal{L}$ . In the third step, for every bootstrap sample  $\mathcal{L}^{*(m)}$  one estimates a model  $\hat{\mu}^m$ , and averaging (7.43) provides the bagging predictor. Bagging is mainly a *variance reduction* technique. Note that if the fitted model of the first step has a bias, then likely the bootstrap samples  $\mathcal{L}^{*(m)}$  are biased, and so is the bagging predictor. Therefore, bagging does not help to reduce a potential bias. All these results have to

be understood conditionally on the data  $\mathcal{L}$ . If this data is atypical for the problem, so will the bootstrap samples be.

We can perform a similar analysis for the fitted networks, but we do not need to bootstrap, here, because the various elements of randomness in SGD fitting allow us to generate independent predictors  $\hat{\mu}^m$ , conditional on the data  $\mathcal{L}$ . Averaging (7.43) over these predictors then provides us with the **network aggregating** (nagging) predictor  $\bar{\mu}^{(1:M)}$ ; we also refer to Dietterich [105] and Richman–Wüthrich [315] for this aggregation. Thus, we replace the bootstrap step by the different runs of the SGD algorithm. Both options provide independent predictors  $\hat{\mu}^m$ , conditional on the data  $\mathcal{L}$ . However, there is a fundamental difference between bagging and nagging. Bagging generates new (bootstrap) samples  $\mathcal{L}^{*(m)}$  and, thus, bagging also involves randomness coming from sampling the new observations. Nagging always acts on the same sample  $\mathcal{L}$ , and it only refits the model multiple times. Therefore, the latter will typically introduce less variation. Of course, bagging and nagging can be combined, and then the full expected GL can be estimated, we come back to this in Sect. 11.4, below. We do not sample new observations, here, because we would like to understand the variations implied by the SGD algorithm with early stopping on the given (fixed) data.

In Fig. 7.18 we have seen that we need nagging over 1'600 network calibrations so that the maximal coefficient of variation on an individual policy level is below 1% in our MTPL example. In this section we would like to understand the minimal out-of-sample loss that can be achieved by nagging on the (entire) test data set, and we would like to analyze its rate of convergence.

For this we define the sequence of nagging predictors

$$\bar{\mu}^{(1:M)}(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^M \hat{\mu}^m(\mathbf{x}) \quad \text{for } M \geq 1. \quad (7.44)$$

This allows us to study the out-of-sample losses on  $\mathcal{T}$  in the Poisson model for  $M \geq 1$

$$\mathfrak{D}(\mathcal{T}, \bar{\mu}^{(1:M)}) = \frac{2}{T} \sum_{t=1}^T v_t^\dagger \left( \bar{\mu}^{(1:M)}(\mathbf{x}_t^\dagger) - Y_t^\dagger - Y_t^\dagger \log \left( \frac{\bar{\mu}^{(1:M)}(\mathbf{x}_t^\dagger)}{Y_t^\dagger} \right) \right).$$

*Remark 7.24* From Remarks 7.17 we know that the expected deviance GL of the estimated model is lower bounded by the expected deviance GL of the true data generating model; the difference is the conditional calibration. Within the family of Tweedie's CP models Richman–Wüthrich [315] proved that, indeed, aggregating decreases monotonically the expected deviance GL of the estimated model (Proposition 2 of [315]), convergence is established (Proposition 3 of [315]),

and the speed of convergence is provided using asymptotic normality (Proposition 4 of [315]). For the Gaussian square loss results we refer to Breiman [51] and Bühlmann–Yu [60].

We revisit Proposition 2 of Richman–Wüthrich [315] which has also been proved in Proposition 3.1 of Denuit–Trufin [103]. We only consider a single case in the next proposition and we drop the feature information  $\mathbf{x}$  (because we can condition on  $X = \mathbf{x}$ ).

**Proposition 7.25** *Choose a response  $Y \sim f(\cdot; \theta, v/\varphi)$  belonging to Tweedie’s CP model having a power variance cumulant function  $\kappa = \kappa_p$  with power variance parameter  $p \in [1, 2]$ , see (2.17). Assume  $\hat{\mu}$  is an estimator for the mean parameter  $\mu = \kappa'_p(\theta) > 0$  satisfying  $\epsilon < \hat{\mu} \leq p/(p-1)\mu$ , a.s., for some  $\epsilon \in (0, p/(p-1)\mu)$ . Choose i.i.d. copies  $\hat{\mu}^m$ ,  $m \geq 1$ , of  $\hat{\mu}$  being all independent of  $Y$ . We have for all  $M \geq 1$*

$$\mathbb{E}_\theta \left[ \mathfrak{d} \left( Y, \hat{\mu}^1 \right) \right] \geq \mathbb{E}_\theta \left[ \mathfrak{d} \left( Y, \bar{\mu}^{(1:M)} \right) \right] \geq \mathbb{E}_\theta \left[ \mathfrak{d} \left( Y, \bar{\mu}^{(1:M+1)} \right) \right] \geq \mathbb{E}_\theta \left[ \mathfrak{d}(Y, \mu) \right].$$

**Proof of Proposition 7.25** The lower bound on the right-hand side immediately follows from Theorem 4.19. For an estimate  $\hat{\mu} > 0$  we define the function, we also refer to (4.18) and we set for the canonical link  $h_p = (\kappa'_p)^{-1}$ ,

$$\hat{\mu} \mapsto \psi_p(\hat{\mu}) = \mu h_p(\hat{\mu}) - \kappa_p(h_p(\hat{\mu})) = \begin{cases} \mu \log(\hat{\mu}) - \hat{\mu} & \text{for } p = 1, \\ \mu \frac{\hat{\mu}^{1-p}}{1-p} - \frac{\hat{\mu}^{2-p}}{2-p} & \text{for } p \in (1, 2), \\ -\mu/\hat{\mu} - \log(\hat{\mu}) & \text{for } p = 2. \end{cases}$$

This is the part of the log-likelihood (and deviance loss) that depends on the canonical parameter  $\hat{\theta} = h_p(\hat{\mu})$ , and replacing the observation  $Y$  by  $\mu$ . Calculating the second derivative w.r.t.  $\hat{\mu}$  provides for  $p \in [1, 2]$

$$\frac{\partial^2}{\partial \hat{\mu}^2} \psi_p(\hat{\mu}) = -p\mu \hat{\mu}^{-p-1} - (1-p)\hat{\mu}^{-p} = \hat{\mu}^{-(1+p)} [-p\mu - (1-p)\hat{\mu}] \leq 0,$$

the last inequality uses that the square bracket is non-positive, a.s., under our assumptions on  $\hat{\mu}$ . Thus,  $\psi_p$  is concave on the interval  $(0, p/(p-1)\mu)$ . We now focus on the inequalities for  $M \geq 1$ . Consider the decomposition of the nagging predictor for  $M+1$

$$\bar{\mu}^{(1:M+1)} = \frac{1}{M+1} \sum_{j=1}^{M+1} \bar{\mu}^{(-j)}, \quad \text{where} \quad \bar{\mu}^{(-j)} = \frac{1}{M} \sum_{m=1}^{M+1} \hat{\mu}^m \mathbb{1}_{\{m \neq j\}}.$$



The predictors  $\bar{\mu}^{(-j)}$ ,  $j \geq 1$ , are copies of  $\bar{\mu}^{(1:M)}$ , though not independent ones. Using the function  $\psi_p$ , the second term on the right-hand side has the same structure as the estimation risk function (4.14),

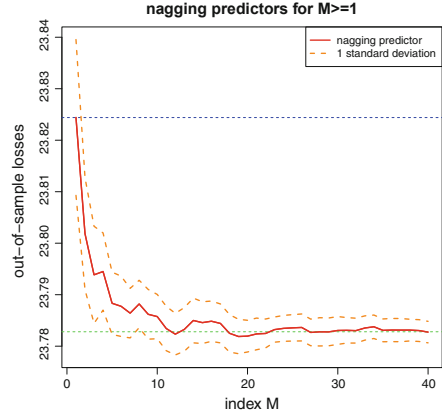
$$\begin{aligned}
& \mathbb{E}_\theta \left[ \mathfrak{d}(Y, \bar{\mu}^{(1:M)}) \right] \\
&= \mathbb{E}_\theta \left[ \mathfrak{d}(Y, \bar{\mu}^{(1:M+1)}) \right] + 2 \mathbb{E}_\theta \left[ Y h_p \left( \bar{\mu}^{(1:M+1)} \right) - \kappa_p \left( h_p \left( \bar{\mu}^{(1:M+1)} \right) \right) \right] \\
&\quad - 2 \mathbb{E}_\theta \left[ Y h_p \left( \bar{\mu}^{(1:M)} \right) - \kappa_p \left( h_p \left( \bar{\mu}^{(1:M)} \right) \right) \right] \\
&= \mathbb{E}_\theta \left[ \mathfrak{d}(Y, \bar{\mu}^{(1:M+1)}) \right] + 2 \left( \mathbb{E} \left[ \psi_p \left( \bar{\mu}^{(1:M+1)} \right) \right] - \mathbb{E} \left[ \psi_p \left( \bar{\mu}^{(1:M)} \right) \right] \right) \\
&= \mathbb{E}_\theta \left[ \mathfrak{d}(Y, \bar{\mu}^{(1:M+1)}) \right] + 2 \left( \mathbb{E} \left[ \psi_p \left( \frac{1}{M+1} \sum_{j=1}^{M+1} \bar{\mu}^{(-j)} \right) \right] - \mathbb{E} \left[ \psi_p \left( \bar{\mu}^{(1:M)} \right) \right] \right) \\
&\geq \mathbb{E}_\theta \left[ \mathfrak{d}(Y, \bar{\mu}^{(1:M+1)}) \right] + 2 \left( \mathbb{E} \left[ \frac{1}{M+1} \sum_{j=1}^{M+1} \psi_p \left( \bar{\mu}^{(-j)} \right) \right] - \mathbb{E} \left[ \psi_p \left( \bar{\mu}^{(1:M)} \right) \right] \right) \\
&= \mathbb{E}_\theta \left[ \mathfrak{d}(Y, \bar{\mu}^{(1:M+1)}) \right],
\end{aligned}$$

the second last step applies Jensen's inequality to the concave function  $\psi_p$ , and the last step follows from the fact that  $\bar{\mu}^{(-j)}$ ,  $j \geq 1$ , are copies of  $\bar{\mu}^{(1:M)}$ .  $\square$

#### Remarks 7.26

- Proposition 7.25 says that aggregation works, i.e., aggregating i.i.d. predictors leads to monotonically decreasing expected deviance GLs. In fact, if  $\hat{\mu} \leq 2\mu$ , a.s., we receive Tweedie's forecast dominance by aggregating, restricted to the power variance parameters  $p \in [1, 2]$ , see Definition 4.22.
- The i.i.d. assumption can be relaxed, indeed, it is sufficient that every  $\bar{\mu}^{(-j)}$  in the above proof has the same distribution as  $\bar{\mu}^{(1:M)}$ . This does not require independence between the predictors  $\hat{\mu}^m$ ,  $m \geq 1$ , but exchangeability is sufficient.
- We need the condition  $\epsilon < \hat{\mu} \leq p/(p-1)\mu$ , a.s., to ensure the monotonicity within Tweedie's CP models. For the Poisson model  $p = 1$  we can drop the upper bound, and we only need the lower bound to ensure the existence of the expected deviance GL. For  $p \in (1, 2]$  the upper bound is increasingly binding, in the gamma case  $p = 2$  requiring  $\hat{\mu} \leq 2\mu$ , a.s.
- Note that we do not require unbiasedness of  $\hat{\mu}$  for  $\mu$  in Proposition 7.25. Thus, at this stage, aggregating is a variance reduction technique.

**Fig. 7.19** Out-of-sample losses  $\mathfrak{D}(\mathcal{T}, \bar{\mu}^{(1:M)})$  of the nagging predictors  $(\bar{\mu}^{(1:M)}(\mathbf{x}_t^\dagger))_{1 \leq t \leq T}$  for  $1 \leq M \leq 40$ ; losses are in  $10^{-2}$



- If additionally we have unbiasedness of  $\hat{\mu}$  for  $\mu$  and a uniformly integrable upper bound on  $\bar{\mu}^{(1:M)}$ , we can use Lebesgue's dominated convergence theorem and the law of large numbers to prove

$$\lim_{M \rightarrow \infty} \mathbb{E}_\theta \left[ \mathfrak{D} \left( Y, \bar{\mu}^{(1:M)} \right) \right] = \mathbb{E}_\theta \left[ \lim_{M \rightarrow \infty} \mathfrak{D} \left( Y, \bar{\mu}^{(1:M)} \right) \right] = \mathbb{E}_\theta [\mathfrak{D}(Y, \mu)]. \quad (7.45)$$

The uniformly integrable upper bound is only needed in the Poisson case  $p = 1$ , because the other cases are covered by  $\epsilon < \hat{\mu} \leq p/(p-1)\mu$ , a.s. Moreover, asymptotic normality can be established, we refer to Proposition 4 in Richman–Wüthrich [315].

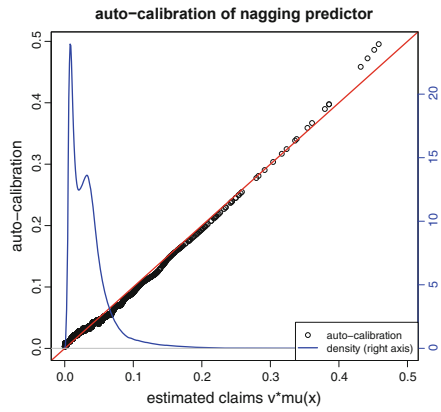
We come back to our MTPL Poisson claim frequency example and its 1/600 network calibrations illustrated in Fig. 7.17. Figure 7.19 provides the out-of-sample portfolio losses  $\mathfrak{D}(\mathcal{T}, \bar{\mu}^{(1:M)})$  of the resulting nagging predictors  $(\bar{\mu}^{(1:M)}(\mathbf{x}_t^\dagger))_{1 \leq t \leq T}$  for  $1 \leq M \leq 40$  in red color, and the corresponding 1 standard deviation confidence bounds in orange color. The blue horizontal dotted line shows the case  $M = 1$  which exactly refers to the (first) bias regularized FN network  $\hat{\mu}^{m=1}$  with embedding layers given in Table 7.5. Indeed, averaging over multiple networks improves the predictive model and the out-of-sample loss decreases over the first  $2 \leq M \leq 10$  nagging steps. After the first 10 steps the picture starts to stabilize which indicates that for this size of portfolio (and this type of problem) we need to average over roughly 10–20 FN networks to receive optimal predictive models on the portfolio level. For  $M \rightarrow \infty$  the out-of-sample loss converges to the green horizontal dotted line in Fig. 7.19 of  $23.783 \cdot 10^{-2}$ . These numbers are also reported on the last line of Table 7.9.

Figure 7.20 provides the empirical auto-calibration property (7.39) of the nagging predictor  $\bar{\mu}^{(1:1600)}$ ; this is obtained completely analogously to Fig. 7.12.

**Table 7.9** Run times, number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model, model Poisson GLM3 of Table 5.5, the FN network models (with embedding layers of dimension  $b = 2$ ), and the nagging predictor for  $M = 1'600$

	Run time	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	25.213	25.445	7.36%
Poisson GLM3	15 s	50	24.084	24.102	7.36%
Embed FN bias regularized $\widehat{\mu}^{m=1}$	+4 s	792	23.690	23.824	7.36%
Average over 1'600 SGD's (Fig. 7.16)	–	792	23.728	23.819	7.36%
Nagging FN $\bar{\mu}^{(1:M)}$ , $M = 1'600$	$\infty$	'792'	23.691	23.783	7.36%

**Fig. 7.20** Empirical auto-calibration (7.39) of the Poisson nagging predictor, the blue line shows the empirical density of  $v_i \bar{\mu}^{(1:1600)}(x_i)$ ,  $1 \leq i \leq n$



The nagging predictors are (already) bias regularized, and Fig. 7.20 supports that the auto-calibration property holds rather accurately.

At this stage, we have fully arrived at Breiman's [53] two modeling cultures dilemma, see also Sect. 1.1. We have started from a parametric data model, and in order to boost its predictive performance we have combined such models in an algorithmic way. Working with many blended networks is not really practical, therefore, in such situations, a meta model can be fitted to the resulting nagging predictor.

## Meta Model

Since working with  $M = 1'600$  different FN networks is not practical, we fit a meta model to the nagging predictors  $\bar{\mu}^{(1:M)}(\cdot)$ . This can easily be done by just selecting an additional FN network and fit this additional network to the working data

$$\mathcal{D}^* = \left\{ \left( \bar{\mu}^{(1:M)}(x_i), x_i, v_i \right) : i = 1, \dots, n \right\} \cup \left\{ \left( \bar{\mu}^{(1:M)}(x_t^\dagger), x_t^\dagger, v_t^\dagger \right) : t = 1, \dots, T \right\}.$$

**Table 7.10** Run times, number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ) and in-sample average frequency of the Poisson null model, model Poisson GLM3 of Table 5.5, the FN network model (with embedding layers of dimension  $b = 2$ ), the nagging predictor, and the meta network model

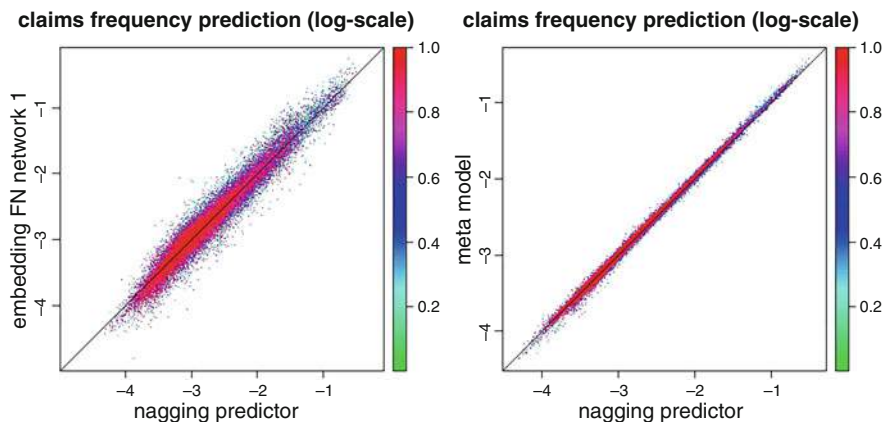
	Run time	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.
Poisson null	–	1	25.213	25.445	7.36%
Poisson GLM3	15 s	50	24.084	24.102	7.36%
Embed FN bias regularized $\hat{\mu}^{m=1}$	+4 s	792	23.690	23.824	7.36%
Nagging FN $\bar{\mu}^{(1:M)}$	$\infty$	‘792’	23.691	23.783	7.36%
Meta FN network $\hat{\mu}^{\text{meta}}$	–	792	23.714	23.777	7.36%

For this calibration step we can consider all data, since we would like to fit a regression model as accurately as possible to the entire regression surface formed by all nagging predictors from the learning and the test data sets  $\mathcal{L}$  and  $\mathcal{T}$ . Moreover, this step should not over-fit since this regression surface of nagging predictors does not include any noise, but it is on the level of expected values. As network architecture we choose again the same FN network of depth  $d = 3$ . The only change to the fitting procedure above is replacing the Poisson deviance loss by the square loss function, since we do not work with the Poisson responses  $N_i$  but rather with their mean estimates  $\bar{\mu}^{(1:M)}(\mathbf{x}_i)$  and  $\bar{\mu}^{(1:M)}(\mathbf{x}_i^\dagger)$  in this fitting step. Since the resulting meta network model may still have a bias we apply the bias regularization step of Listing 7.7 to the Poisson observations with the Poisson deviance loss on the learning data  $\mathcal{L}$  (only). The results are presented in Table 7.10.

From these results we observe that in our case the meta network performs similarly well to the nagging predictor, and it seems to be a very reasonable choice.

Finally, in Fig. 7.21 (lhs) we analyze the resulting frequencies on an individual policy level on the test data set  $\mathcal{T}$ . We plot the estimated frequencies  $\hat{\mu}^{m=1}(\mathbf{x}_i^\dagger)$  of the first FN network (this corresponds to ‘embed FN bias regularized’ in Table 7.10 with an out-of-sample loss of 23.824) against the nagging predictor  $\bar{\mu}^{(1:M)}(\mathbf{x}_i^\dagger)$  which averages over  $M = 1'600$  networks. From Fig. 7.21 (lhs) we conclude that there are quite some differences between these two predictors, this exactly reflects the variations obtained in Fig. 7.18 (lhs). The nagging predictor removes this variation by averaging. Figure 7.21 (rhs) compares the nagging predictor  $\bar{\mu}^{(1:M)}(\mathbf{x}_i^\dagger)$  to the one of the meta model  $\hat{\mu}^{\text{meta}}(\mathbf{x}_i^\dagger)$ . This scatter plot shows that the predictors lie almost perfectly on the diagonal line which suggests that the meta model can be used as a substitute for the nagging predictor. This completes this claim frequency modeling example.

*Remark 7.27* The meta model concept can also be useful in other situations. For instance, we can fit a gradient boosting regression model to the observations. Typically, this is much faster than calculating a nagging predictor (because it directly focuses on the weaknesses of the existing model). If the gradient boosting model is based on regression trees, it has the disadvantage that the resulting regression



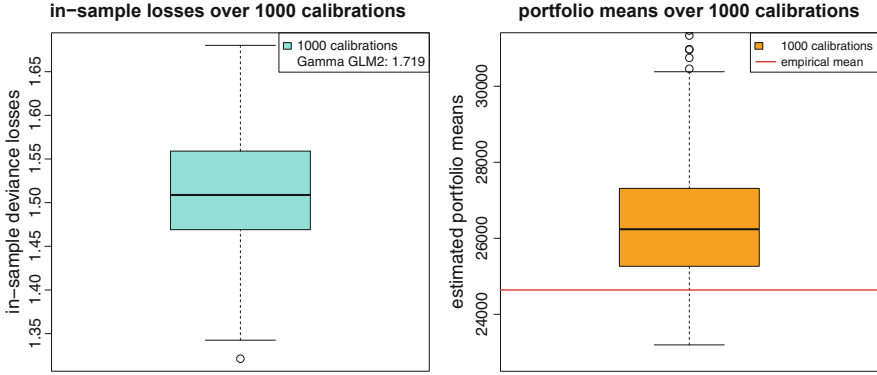
**Fig. 7.21** Scatter plot of the out-of-sample predictions  $\hat{\mu}^{m=1}(x_t^\dagger)$ ,  $\hat{\mu}^{(1:M)}(x_t^\dagger)$  and  $\hat{\mu}^{\text{meta}}(x_t^\dagger)$  over all policies  $1 \leq t \leq T$  on the test data set  $\mathcal{T}$ : (lhs)  $\hat{\mu}^{m=1}(x_t^\dagger)$  vs.  $\hat{\mu}^{(1:M)}(x_t^\dagger)$  and (rhs)  $\hat{\mu}^{\text{meta}}(x_t^\dagger)$  vs.  $\hat{\mu}^{(1:M)}(x_t^\dagger)$ ; the color scale shows the exposures  $v_t^\dagger \in (0, 1]$

function is not continuous, and a non-constant extrapolation might be an issue. In a second step we can fit a meta FN network model to the former regression model, lifting the boosting model to a smooth network that allows for a non-constant extrapolation.

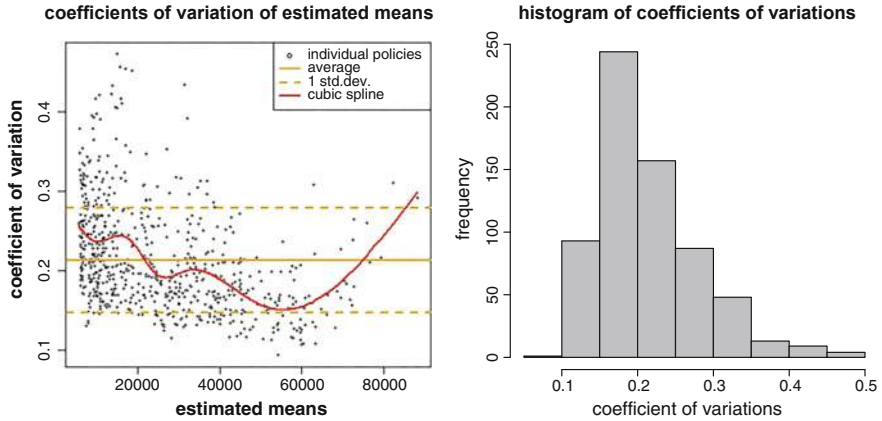
*Example 7.28 (Gamma Claim Size Modeling)* We revisit the gamma claim size example of Sect. 5.3.7. The data comprises Swedish motorcycle claim amounts. We have seen that this claim size data is not heavy-tailed, thus, a gamma distribution may be a reasonable choice for this data. For the modeling of this data we use the same normalization as in (5.45), this parametrization does not require the explicit knowledge of the (constant) shape parameter of the gamma distribution for mean estimation.

The difficulty with this data is that only 656 insurance policies suffer a claim, and likely a single FN network will not lead to stable results in this example. As FN network architecture we again choose a network of depth  $d = 3$  and with  $(q_1, q_2, q_3) = (20, 15, 10)$  neurons. Since the input layer has dimension  $q_0 = 1 + 6 = 7$  we receive a network parameter of dimension  $r = 626$ . As loss function we choose the gamma deviance loss, see Table 4.1. Moreover, we choose the nadam optimizer, a batch size of 300, a training-validation split of 8:2, and we retrieve the network calibration with the lowest validation loss with a callback.

Figure 7.22 shows the results of 1'000 different SGD runs (only differing in the initial seeds and the splits of the training-validation sets as well as the batches). We see a considerable variation between the different SGD runs, both in in-sample deviance losses but also in the average estimated claims. Note that we did not bias-regularize the resulting networks (we work with the log-link here which is not the canonical one). This is why we receive fluctuating portfolio averages in Fig. 7.22



**Fig. 7.22** Boxplots over 1'000 network calibrations only differing in the seeds for the SGD algorithm and the partitioning of the learning-validation data: (lhs) in-sample losses on the (entire) data  $\mathcal{L}$  and (rhs) average estimated claims

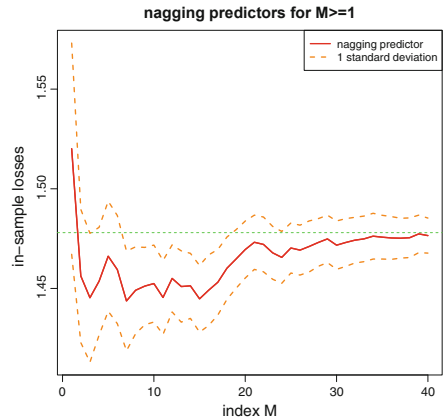


**Fig. 7.23** Coefficients of variations  $V_{co_i}$  on an individual claim level  $1 \leq i \leq n$  over the 1'000 calibrations (lhs) scatter plot against the nagging predictor  $\tilde{\mu}^{(1:M)}(\mathbf{x}_i)$  and (rhs) histogram

(rhs), the red line illustrates the empirical mean. Obviously, these FN networks are (on average) positively biased, and they will need a bias correction for the final prediction.

Figure 7.23 analyzes the variations on an individual claim level by studying the in-sample version of the coefficient of variation given in (7.43). We see that these coefficients of variation are bigger than in the claim frequency example, see Fig. 7.18. Thus, to receive stable results the nagging predictors  $\tilde{\mu}^{(1:M)}(\mathbf{x}_i)$  have to be calculated over many networks. Figure 7.24 confirms that aggregating reduces (in-sample) losses also in this case. From this figure we also see that the convergence is slower compared to the MTPL frequency example of Fig. 7.19, of course, because we have a much smaller claims portfolio.

**Fig. 7.24** In-sample losses  $\mathfrak{D}(\mathcal{L}, \bar{\mu}^{(1:M)})$  of the nagging predictors  $(\bar{\mu}^{(1:M)}(\mathbf{x}_i))_{1 \leq i \leq n}$  for  $1 \leq M \leq 40$  on the motorcycle claim size data

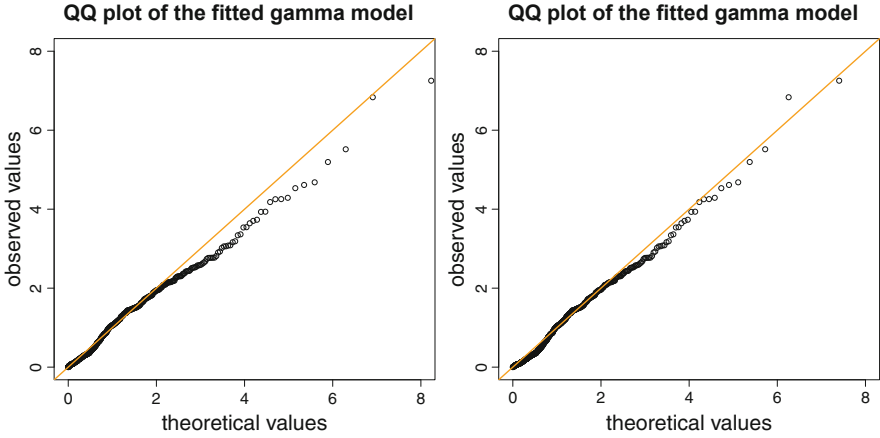


**Table 7.11** Number of parameters, Pearson’s dispersion estimate, MLE dispersion estimate, in-sample losses and in-sample average claim amounts of the null model (gamma intercept model), the gamma GLMs and the network nagging predictor; for the GLMs we refer to Table 5.13

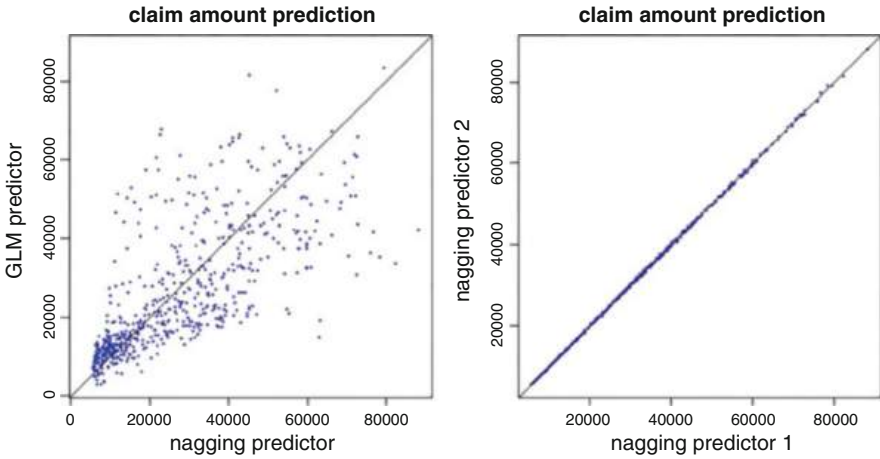
	# param.	Dispersion		In-sample loss on $\mathcal{L}$	Average amount
		$\hat{\varphi}^P$	$\hat{\varphi}^{MLE}$		
Gamma null	1 + 1	2.057	1.690	2.085	24’641
Gamma GLM1	9 + 1	1.537	1.426	1.717	25’105
Gamma GLM2	7 + 1	1.544	1.427	1.719	25’130
Gamma FN network nagging	626 + 1	–	–	1.478	26’387
Gamma FN network nagging (bias reg)	626 + 1	<i>1.050</i>	1.240	1.465	24’641

Table 7.11 presents the results if we take the nagging predictor over 1’000 different networks. The first observation is that we receive a much smaller in-sample loss compared to the GLMs, thus, there seems to be much room for improvements in the GLMs. Secondly, the nagging predictor has a substantial bias. For this reason we shift the intercept parameter in the output layer so that the portfolio average of the nagging predictor is equal to the empirical mean, see the last column of Table 7.11.

A main difficulty in this model is the estimation of the dispersion parameter  $\varphi > 0$  and the shape parameter  $\alpha = 1/\varphi$  of the gamma distribution, respectively. Pearson’s dispersion estimate does not work because we do not know the degrees of freedom of the nagging predictor, see also (5.49). In Table 7.11 we calculate Pearson’s dispersion estimate by simply dividing by the number of observations; this should be understood as a lower bound; this number is highlighted in italic. Alternatively, we can calculate the MLE, however, this may be rather different from Pearson’s estimate, as indicated in Table 7.11. Figure 7.25 (lhs) shows the resulting QQ plot of the nagging predictor if we use the MLE  $\hat{\varphi}^{MLE} = 1.240$ , and the right-hand side shows the same plot for  $\hat{\varphi} = 1.050$ . From these plots it seems that we should rather go for a smaller dispersion parameter, the MLE being probably too much dominated by the small claims. This observation should also be understood as a red flag, as it tells us that the chosen gamma model is not fully suitable. This may



**Fig. 7.25** QQ plots of the nagging predictors against the gamma density with (lhs)  $\hat{\varphi}^{\text{MLE}} = 1.240$  and (rhs)  $\hat{\varphi} = 1.050$



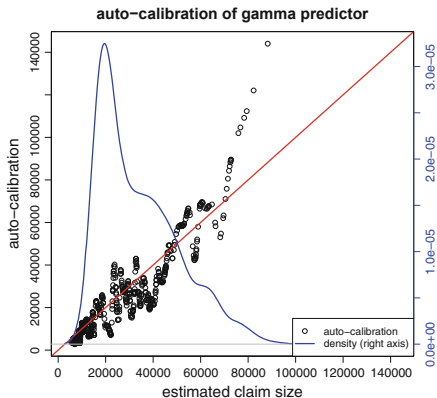
**Fig. 7.26** (lhs) Scatter plot of model Gamma GLM2 predictors against the nagging predictors  $\bar{\mu}^{(1:M)}(\mathbf{x}_i)$  over all instances  $1 \leq i \leq n$ , (rhs) scatter plot of two (independent) nagging predictors

be for various reasons: (1) the dispersion is not constant and should be modeled policy dependent, (2) the features are not sufficient to explain the observations, or (3) the gamma distribution is not suitable and should be replaced by another distribution.

In Fig. 7.26 (lhs) we compare the predictions received from model Gamma GLM2 against the nagging predictors  $\bar{\mu}^{(1:M)}(\mathbf{x}_i)$  over all instances  $1 \leq i \leq n$ . The scatter plot spreads quite wildly around the diagonal which seriously questions at least one of the two models. To ensure that this variability between the two models is not caused by the (complex) FN network architecture, we verify the nagging



**Fig. 7.27** Empirical auto-calibration (7.39) of the Gamma FN network nagging predictor of Table 7.11, the blue line shows the empirical density of  $\tilde{\mu}^{(1:M)}(x_i)$ ,  $1 \leq i \leq n$



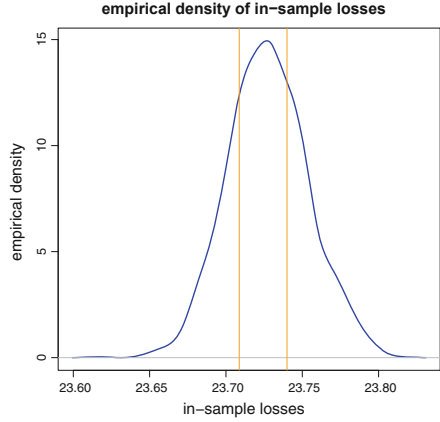
predictor  $\tilde{\mu}^{(1:M)}$ ,  $M = 1'000$ , by computing a second independent one. Indeed, Fig. 7.26 shows that these two independent nagging predictors come to the same conclusion on the individual instance level. Thus, the network finds/uses systematic effects that are not present in model Gamma GLM2. If we perform a pairwise interaction analysis for boosting the GLM as in Example 7.23, we find that we should add interactions to the GLM between (VehAge, RiskClass), (VehAge, BonusClass), (OwnerAge, Area), and (OwnerAge, VehAge); recall that model Gamma GLM2 neither includes BonusClass nor Gender as supported by a drop1 backward elimination analysis from model Gamma GLM1. However, it turns out, here, that we should have BonusClass in the model by letting it interact with VehAge.

Finally, Fig. 7.27 shows the empirical auto-calibration behavior (7.39) of the Gamma FN network nagging predictor of Table 7.11. The resulting black dots are rather volatile which shows that we do not (fully) have the auto-calibration property, here, but it also expresses that we fit a model on only 656 claims. The prediction of these claims is highlighted by the blue empirical density given by  $\tilde{\mu}^{(1:M)}(x_i)$ ,  $1 \leq i \leq n$ . On the positive side, the auto-calibration plot shows that we neither systematically under- nor over-estimate because the black dots fluctuate around the diagonal red line, only the upper tail seems to under-estimate the true claim size. ■

## Ensembling over Selected Networks vs. All Networks

Zhou et al. [406] ask the question whether ensembling over ‘selected’ networks is better than ensembling over all networks. In their proposal they introduce a weighted averaging scheme over the different network predictors  $\hat{\mu}^m$ ,  $1 \leq m \leq M$ . We perform a slightly different analysis here. We are re-using the  $M = 1'600$  SGD calibrations of the Poisson FN network illustrated in Fig. 7.17. We order these SGD calibrations w.r.t. their in-sample losses  $\mathfrak{D}(\mathcal{L}, \hat{\mu}^m)$ ,  $1 \leq m \leq M$ , and partition this ordered sample into three equally sized sets: the first one containing the smallest

**Fig. 7.28** Empirical density of the in-sample losses  $\mathfrak{D}(\mathcal{L}, \hat{\mu}^m)$ ,  $1 \leq m \leq M$ , of Fig. 7.17



in-sample losses, the second one the middle sized in-sample losses, and the third one the largest in-sample losses. Figure 7.28 shows the empirical density of these in-sample losses, and the vertical lines give the partition into the three sets, we call the resulting (disjoint) index sets  $\mathcal{I}^{\text{small}}, \mathcal{I}^{\text{middle}}, \mathcal{I}^{\text{large}} \subset \{1, \dots, M\}$ . Remark that this partition is done fully *in-sample*, based on the learning data  $\mathcal{L}$ , only.

We then consider the nagging predictors on each of these index sets separately, i.e.,

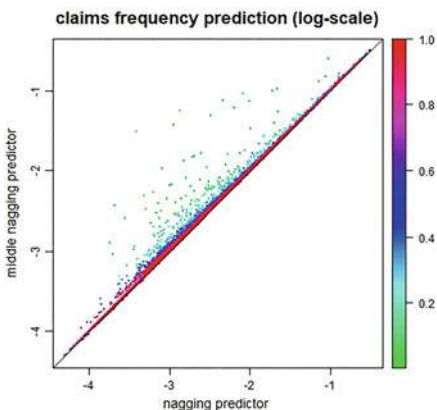
$$\begin{aligned}\bar{\mu}^{\text{small}}(\mathbf{x}) &= \frac{1}{|\mathcal{I}^{\text{small}}|} \sum_{m \in \mathcal{I}^{\text{small}}} \hat{\mu}^m(\mathbf{x}), \\ \bar{\mu}^{\text{middle}}(\mathbf{x}) &= \frac{1}{|\mathcal{I}^{\text{middle}}|} \sum_{m \in \mathcal{I}^{\text{middle}}} \hat{\mu}^m(\mathbf{x}), \\ \bar{\mu}^{\text{large}}(\mathbf{x}) &= \frac{1}{|\mathcal{I}^{\text{large}}|} \sum_{m \in \mathcal{I}^{\text{large}}} \hat{\mu}^m(\mathbf{x}).\end{aligned}\tag{7.46}$$

If we believe into the orange cubic spline in Fig. 7.17, the middle nagging predictor  $\bar{\mu}^{\text{middle}}$  should out-perform the other two nagging predictors. Indeed, this is the case, here. We receive the out-of-sample losses (in  $10^{-2}$ ) on the three subsets

$$\mathfrak{D}(\mathcal{T}, \bar{\mu}^{\text{small}}) = 23.784, \quad \mathfrak{D}(\mathcal{T}, \bar{\mu}^{\text{middle}}) = 23.272, \quad \mathfrak{D}(\mathcal{T}, \bar{\mu}^{\text{large}}) = 23.782.\tag{7.47}$$

This approach boosts by far any other approach considered, see Table 7.10; note that this analysis relies on a fully proper in-sample and out-of-sample testing strategy. Moreover, this also supports our early stopping strategy because, obviously, the optimal networks are centered around our early stopping rule. How does this result match Proposition 7.25 saying that the nagging predictor has a monotonically

**Fig. 7.29** Scatter plot of the nagging predictors  $\bar{\mu}^{\text{middle}}(\mathbf{x}_i^\dagger)$  and  $\bar{\mu}^{(1:M)}(\mathbf{x}_i^\dagger)$  over all out-of-sample policies  $1 \leq i \leq T$ ; the color scale shows the sizes of the exposures  $v_i^\dagger \in (0, 1]$



decreasing deviance loss. For the convergence (7.45) we need unbiasedness, and (7.47) indicates that averaging over all  $M$  network calibrations results in biases on an *individual* policy level; on the aggregate portfolio level, we have applied the bias regularization step (7.33), but this does not act on an individual policy level. The latter would require a local balance correction similar to the GAM approach presented in Example 7.19.

Figure 7.29 is truly striking! It compares the nagging predictors  $\bar{\mu}^{(1:M)}(\mathbf{x}_i^\dagger)$  to the ones  $\bar{\mu}^{\text{middle}}(\mathbf{x}_i^\dagger)$  only using the calibrations  $m \in \mathcal{I}^{\text{middle}}$ , i.e., only using the calibrations with middle sized in-sample losses. The different colors show the exposures  $v_i^\dagger \in (0, 1]$ . We observe that only portfolios with short exposures do not lie on the diagonal line. Thus, there seems to be an issue with insurance policies with short exposures. Recall that we model the Poisson claim counts  $N_i$  using the assumption, see (5.27),

$$N_i \sim \text{Poi}(v_i \mu(\mathbf{x}_i)). \quad (7.48)$$

That is, the expected claim count  $\mathbb{E}_{\theta_i}[N_i] = v_i \mu(\mathbf{x}_i)$  is assumed to scale proportionally in the exposure  $v_i > 0$ . Figure 7.29 raises some doubts whether this is really the case, or at least SGD fitting has some difficulties to assess the expected frequencies  $\mu(\mathbf{x}_i)$  on the policies  $i$  with short exposures  $v_i > 0$ . We discuss this further in the next subsection. Table 7.12 gives a summary of our results.

### Analysis of Over-dispersion

With all the excitement of Fig. 7.29, the above models do not fit the observations since the over-dispersion is too large, see the last column of Table 7.12. This has motivated the study of the negative binomial model in Sect. 5.3.5, the ZIP model in Sect. 5.3.6, and the hurdle Poisson model in Example 6.19. These models have led to an improvement in terms of AIC, see Table 6.6. We could go down the same

**Table 7.12** Number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ), in-sample average frequency and (over-)dispersion of the Poisson null model, model Poisson GLM3 of Table 5.5, the FN network model (with embedding layers of dimension  $b = 2$ ), the nagging predictor, the meta network model, and the middle nagging predictor

	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.	Disp. $\widehat{\varphi}^P$
Poisson null	1	25.213	25.445	7.36%	1.7160
Poisson GLM3	50	24.084	24.102	7.36%	1.6644
Embed FN bias regularized $\widehat{\mu}^{m=1}$	792	23.690	23.824	7.36%	1.6812
Nagging FN $\widehat{\mu}^{(1:M)}$	‘792’	23.691	23.783	7.36%	1.6592
Meta FN network $\widehat{\mu}^{\text{meta}}$	792	23.714	23.777	7.36%	1.6737
Middle nagging FN $\widehat{\mu}^{\text{middle}}$	‘792’	23.698	23.272	7.36%	1.6618

route here by substituting the Poisson model. We refrain from doing so, as we want to further analyze the Poisson model. Suppose we calculate an AIC value for the Poisson FN network using 792 as the number of parameters involved. In that case, we receive a value of 191’790, thus, clearly lower than the one of the negative binomial GLM, and also slightly lower than the one of the hurdle Poisson model, see Table 6.6. Remark that AIC values within FN networks are not supported by any theory as we neither use the MLE nor do we have a reasonable evaluation of the number of parameters involved in networks. Thus, such a value may serve at best as a rough rule of thumb.

This lower AIC value suggests that we should try to improve the modeling of the systematic effects by better regression functions. In particular, there may be more explanatory variables involved that have predictive power. If these explanatory variables are latent, we can rely on the negative binomial model, as it can be interpreted as a mixture model averaging over latent variables. In view of Fig. 7.29, the exposures  $v_i$  seem to have a predictive power different from proportional scaling, see (7.48); we also mention some peculiarities of the exposures on page 556. This motivates to change the FN network regression model such that the exposures are considered non-proportionally. We choose a FN network that directly models the mean of the claim counts

$$(\mathbf{x}, v) \in \mathcal{X} \times (0, 1] \mapsto \mu(\mathbf{x}, v) = \exp\left\langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}, v) \right\rangle > 0, \quad (7.49)$$

modeling the mean  $\mathbb{E}_{\boldsymbol{\theta}}[N] = \mu(\mathbf{x}, v)$  of the Poisson datum  $(N, \mathbf{x}, v)$ . The expected frequency is then given by  $\mathbb{E}_{\boldsymbol{\theta}}[Y] = \mathbb{E}_{\boldsymbol{\theta}}[N/v] = \mu(\mathbf{x}, v)/v$ .

*Remark 7.29* At this stage we clearly have to distinguish between statistical modeling and actuarial modeling. In statistical modeling it makes perfect sense to choose the regression function (7.49), since including the exposure in a non-proportional way may increase the predictive power of the model, at least this is what our data suggests.

From an actuarial point of view this approach should clearly be doubted. The typical exposure of car insurance policies is one calendar year, i.e.,  $v = 1$ , if the renewals of insurance policies are accounted correctly. Shorter exposures may have a specific (non-predictable) reason, for example, the policyholder or the insurance company may terminate an insurance contract after a claim. Thus, if this is possible, the exposure is a random variable, too, and it clearly has a predictive power for claims prediction; in that case we lose the properties of the Poisson count process (having independent and stationary increments).

As a consequence, we should include the exposure proportionally from an actuarial modeling point of view. Nevertheless we do the modeling exercise based on the regression function (7.49), here. This will indicate the predictive power of the exposure, which may be thought of a proxy for another (non-available) explanatory variable. Moreover, if (7.49) allows for a good Poisson regression model, we have a simple way of bootstrapping from our data (conditionally on given exposures  $v$ ).

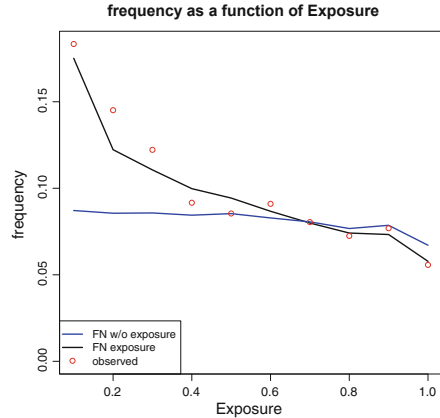
We would also like to emphasize that if one feature component dominates all others in terms of the predictive power, then likely there is a leakage of information through this component, and this needs a more careful analysis.

We implement the FN network regression model (7.49) using again a network architecture of depth  $d = 3$  with  $(q_1, q_2, q_3) = (20, 15, 10)$  neurons. We use embedding layers for the two categorical variables VehBrand and Region, and we have 8 continuous/binary feature components. This is one more compared to Fig. 7.9 (rhs) because we also model the exposure  $v_i$  as a continuous input to the network. As a result, the dimension  $r$  of the network parameter  $\vartheta \in \mathbb{R}^r$  increases from 792 to 812 (because we have  $q_1 = 20$  neurons in the first FN layer). We calculate the nagging predictor  $\bar{\mu}^{(1:M)}$  of this network averaging over  $M = 500$  individual (early stopped) FN network calibrations, the results are presented in Table 7.13.

**Table 7.13** Number of parameters, in-sample and out-of-sample deviance losses (units are in  $10^{-2}$ ), in-sample average frequency and (over-)dispersion of the Poisson null model, model Poisson GLM3 of Table 5.5, the FN network models (with embedding layers of dimension  $b = 2$ ), the nagging predictors, and the middle nagging predictors excluding and including exposures  $v_i$  as continuous network inputs

	# param.	In-sample loss on $\mathcal{L}$	Out-of-sample loss on $\mathcal{T}$	Aver. freq.	Disp. $\hat{\varphi}^P$
Poisson null	1	25.213	25.445	7.36%	1.7160
Poisson GLM3	50	24.084	24.102	7.36%	1.6644
Embed FN $\hat{\mu}^{m=1}$	792	23.690	23.824	7.36%	1.6812
Nagging FN $\bar{\mu}^{(1:M)}$	‘792’	23.691	23.783	7.36%	1.6592
Middle nagging FN $\bar{\mu}^{\text{middle}}$	‘792’	23.698	23.272	7.36%	1.6618
Exposure $v$ : FN $\hat{\mu}^{m=1}$	812	23.358	23.496	7.36%	1.0650
Exposure $v$ : nagging FN $\bar{\mu}^{(1:M)}$	‘812’	23.299	23.382	7.36%	1.0416
Exposure $v$ : middle nagging FN $\bar{\mu}^{\text{middle}}$	‘812’	23.303	23.299	7.36%	1.0427

**Fig. 7.30** Average frequency as a function of the exposure  $v \in (0, 1]$ : nagging predictors considering the exposures proportionally (blue), the model including exposures non-proportionally through the FN network (black) and observed (red)



We observe a major improvement when including the exposure  $v$  as an input to the network, i.e., by including the exposure non-proportionally into the mean estimate. This is true in-sample (we use early stopping here), and in terms of Pearson's dispersion estimate; we set  $r = 812$  for the number of parameters in Pearson's dispersion estimate (5.30) which may be too big because we do not perform proper MLE, here. In particular, we receive a dispersion estimate close to one which, now, is in support of modeling the claim counts by Poisson random variables (using this regression function). That is, this regression function explains the systematic effects so that we no longer observe much over-dispersion in the data relative to the chosen model. However, we would like to remind of Remark 7.29 which needs a careful consideration for the use of this regression model in insurance practice.

This is also supported by Fig. 7.30 which studies the average frequency as a function of the exposure  $v \in (0, 1]$ . The red observed average frequency has a clear decreasing slope which can be modeled by running the exposure  $v$  through the FN network (black), but not by including it proportionally (blue). From an actuarial modeling point of view this plot clearly questions the quality of the data, because there seem to be effects in the exposures that certainly require more investigation. Unfortunately, we cannot do this here because we do not have additional insight into this data set. This closes the example.

#### 7.4.5 Identifiability in Feed-Forward Neural Networks

In the previous section we have studied ensembles of FN networks. One may also aim at directly comparing these networks to each other in terms of the fitted network parameters  $\hat{\boldsymbol{\theta}}^j$  over the different calibrations  $1 \leq j \leq M$  (of the same FN network architecture). Such a comparison may, e.g., be useful if one wants to choose a

prior parameter distribution  $\pi$  for  $\vartheta$  in a Bayesian setting. Comparing the different network calibrations  $\hat{\vartheta}^j$ ,  $1 \leq j \leq M$ , of an architecture needs some care because networks have many symmetries that make the parameters non-identifiable. We can, for instance, permute the neurons in a FN layer  $\mathbf{z}^{(m)}$ , with the corresponding permutation of the weights that connect this layer to the previous layer  $\mathbf{z}^{(m-1)}$  and to the succeeding layer  $\mathbf{z}^{(m+1)}$ . The resulting predictive model under this permutation is the same as the original one. For this reason we need to introduce some order in a FN network to make the parameters identifiable.

Rüger–Ossen [323] have introduced the notion of a fundamental domain for the network parameter  $\vartheta$ , and we briefly review this idea. We start with an explicit example. Assume that the activation function fulfills the anti-symmetry property  $-\phi(x) = \phi(-x)$  for all  $x \in \mathbb{R}$ , this is the case for the hyperbolic tangent. This implies several symmetries in the FN network parametrization. E.g., if we consider the output of a shallow FN network  $d = 1$  with link function  $g$ , we can do a sign switch in a fixed neuron  $1 \leq k \leq q_1$

$$\begin{aligned} g(\mu(\mathbf{x})) &= \beta_0 + \sum_{j=1}^{q_1} \beta_j z_j^{(1:1)}(\mathbf{x}) = \beta_0 + \sum_{j=1}^{q_1} \beta_j \phi(\mathbf{w}_j^{(1)}, \mathbf{x}) \\ &= \beta_0 + \sum_{j \neq k} \beta_j \phi(\mathbf{w}_j^{(1)}, \mathbf{x}) + (-\beta_k) \phi(-\mathbf{w}_k^{(1)}, \mathbf{x}). \end{aligned} \quad (7.50)$$

From this we see that the following two network parameters (we switch signs in all the parameters that belong to index  $k$ )

$$\begin{aligned} \vartheta &= (\mathbf{w}_1^{(1)}, \dots, \mathbf{w}_k^{(1)}, \dots, \mathbf{w}_{q_1}^{(1)}, \beta_0, \dots, \beta_k, \dots, \beta_{q_1})^\top \quad \text{and} \\ \tilde{\vartheta} &= (\mathbf{w}_1^{(1)}, \dots, -\mathbf{w}_k^{(1)}, \dots, \mathbf{w}_{q_1}^{(1)}, \beta_0, \dots, -\beta_k, \dots, \beta_{q_1})^\top \end{aligned}$$

give the same FN network predictions. Beside these sign switches, we can also permute the enumeration of the neurons in a given FN layer, giving the same predictions. We discuss Theorem 2 of Rüger–Ossen [323] to solve this identifiability issue. First, we consider the network weights from the input  $\mathbf{x}$  to the first FN layer  $\mathbf{z}^{(1)}(\mathbf{x})$ . Apply the sign switch operation (7.50) to the neurons in the first FN layer so that all the resulting intercepts  $w_{0,1}^{(1)}, \dots, w_{0,q_1}^{(1)}$  are positive while not changing the regression function  $\mathbf{x} \mapsto g(\mu(\mathbf{x}))$ . Next, apply a permutation to the indices  $1 \leq j \leq q_1$  so that we receive ordered intercepts

$$w_{0,1}^{(1)} > \dots > w_{0,q_1}^{(1)} > 0,$$

with an unchanged regression function  $\mathbf{x} \mapsto g(\mu(\mathbf{x}))$ . To make these transformations well-defined we need to assume that all intercepts are non-zero and mutually different (which we assume for the time-being).

Then, we move recursively through the FN layers  $2 \leq m \leq d$  applying the sign switch operations and the permutations so that the regression function  $\mathbf{x} \mapsto g(\mu(\mathbf{x}))$  remains unchanged and such that for all  $1 \leq m \leq d$

$$w_{0,1}^{(m)} > \dots > w_{0,q_m}^{(m)} > 0.$$

This provides us with a unique representation of every network parameter  $\boldsymbol{\vartheta} \in \mathbb{R}^r$  in the *fundamental domain*

$$\left\{ \boldsymbol{\vartheta} \in \mathbb{R}^r; w_{0,1}^{(m)} > \dots > w_{0,q_m}^{(m)} > 0 \text{ for all } 1 \leq m \leq d \right\} \subset \mathbb{R}^r, \quad (7.51)$$

supposed that all intercepts are different from zero and mutually different in the same FN layers. As stated in Section 2.2 of Rüger–Ossen [323], there may still exist different parameters in this fundamental domain that provide the same predictive model, but these are of zero Lebesgue measure. The same applies to the intercepts  $w_{0,j}^{(m)}$  being zero or having equal intercepts for different neurons. Basically, this means that we are fine if we work with absolutely continuous prior distributions on the fundamental domain when we want to work within a Bayesian setup.

## 7.5 Auto-encoders

Auto-encoders are tools that aim at reducing the dimension of high-dimensional data such that the reconstruction error of the original data is small, i.e., such that the loss of information by the dimension reduction is minimized. The most popular auto-encoder is the principal components analysis (PCA) which we are going to present here. The PCA is a linear dimension reduction technique. Bottleneck neural (BN) networks can be viewed as a non-linear extension of the PCA. This is going to be discussed in Sect. 7.5.5, below. Dimension reduction techniques belong to the family of unsupervised learning methods because they do not consider a response variable, but they aim at finding common structure in the features. Unsupervised learning methods can roughly be categorized into three classes: dimension reduction techniques (studied in this section), clustering methods and visualization methods. For a discussion of clustering and visualization methods we refer to the tutorial of Rentzmann–Wüthrich [310].



### 7.5.1 Standardization of the Data Matrix

Assume we have  $q$ -dimensional data points  $\mathbf{y}_i \in \mathbb{R}^q$ ,  $1 \leq i \leq n$ . This provides us with a data matrix

$$\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)^\top = \begin{pmatrix} y_{1,1} & \cdots & y_{1,q} \\ \vdots & \ddots & \vdots \\ y_{n,1} & \cdots & y_{n,q} \end{pmatrix} \in \mathbb{R}^{n \times q}.$$

We assume that each of the  $q$  columns of  $\mathbf{Y}$  measures a quantity in a given unit. The first column may, for instance, describe the age of a car driver in years, the second column his body weight in kilograms, etc. That is, each column  $1 \leq j \leq q$  of  $\mathbf{Y}$  describes a specific quantity, and each row  $\mathbf{y}_i^\top$  of  $\mathbf{Y}$  describes these quantities for a given instance  $1 \leq i \leq n$ . Since often the analysis should not depend on the units of the columns of  $\mathbf{Y}$ , one centers the columns with the empirical means  $\bar{y}_j = \sum_{i=1}^n y_{i,j} / n$ , and one normalizes them with the empirical standard deviations  $\hat{\sigma}_j = (\sum_{i=1}^n (y_{i,j} - \bar{y}_j)^2 / n)^{1/2}$ ,  $1 \leq j \leq q$ . This gives the normalized data matrix

$$\begin{pmatrix} \frac{y_{1,1} - \bar{y}_1}{\hat{\sigma}_1} & \cdots & \frac{y_{1,q} - \bar{y}_q}{\hat{\sigma}_q} \\ \vdots & \ddots & \vdots \\ \frac{y_{n,1} - \bar{y}_1}{\hat{\sigma}_1} & \cdots & \frac{y_{n,q} - \bar{y}_q}{\hat{\sigma}_q} \end{pmatrix} \in \mathbb{R}^{n \times q}. \quad (7.52)$$

We typically center the data matrix  $\mathbf{Y}$ , providing  $\sum_{i=1}^n y_{i,j} = 0$  for all  $1 \leq j \leq q$ , normalization w.r.t. the standard deviation can be done, but is not always necessary. Centering implies that we can interpret  $\mathbf{Y}$  as a  $q$ -dimensional empirical distribution with each component (column) being centered. The covariance matrix of this (centered) empirical distribution is calculated as

$$\hat{\Sigma} = \frac{1}{n} \left( \sum_{i=1}^n y_{i,j} y_{i,k} \right)_{1 \leq j, k \leq q} = \frac{1}{n} \mathbf{Y}^\top \mathbf{Y} \in \mathbb{R}^{q \times q}. \quad (7.53)$$

This is a covariance matrix, and if the columns of  $\mathbf{Y}$  are normalized with the empirical standard deviations  $\hat{\sigma}_j$ ,  $1 \leq j \leq q$ , this is a correlation matrix.

### 7.5.2 Introduction to Auto-encoders

An auto-encoder encodes a high-dimensional vector  $\mathbf{y} \in \mathbb{R}^q$  to a low-dimensional representation so that the dimension reduction leads to a minimal loss of information. A function  $L(\cdot, \cdot) : \mathbb{R}^q \times \mathbb{R}^q \rightarrow \mathbb{R}_+$  is called *dissimilarity function* if  $L(\mathbf{y}, \mathbf{y}') = 0$  if and only if  $\mathbf{y} = \mathbf{y}'$ .

An auto-encoder is a pair  $(\Phi, \Psi)$  of mappings, for given dimensions  $p < q$ ,

$$\Phi : \mathbb{R}^q \rightarrow \mathbb{R}^p \quad \text{and} \quad \Psi : \mathbb{R}^p \rightarrow \mathbb{R}^q, \quad (7.54)$$

such that their composition  $\Psi \circ \Phi$  has a small reconstruction error w.r.t. the chosen dissimilarity function  $L(\cdot, \cdot)$ , that is,

$$\mathbf{y} \mapsto L(\mathbf{y}, \Psi \circ \Phi(\mathbf{y})) \text{ is small for all cases } \mathbf{y} \text{ of interest.} \quad (7.55)$$

Note that we want (7.55) for selected cases  $\mathbf{y}$ , and if they are within a  $p$ -dimensional manifold the auto-encoding will be successful. The first mapping  $\Phi : \mathbb{R}^q \rightarrow \mathbb{R}^p$  is called encoder, and the second mapping  $\Psi : \mathbb{R}^p \rightarrow \mathbb{R}^q$  is called decoder. The object  $\Phi(\mathbf{y}) \in \mathbb{R}^p$  is a  $p$ -dimensional encoding (representation) of  $\mathbf{y} \in \mathbb{R}^q$  which contains maximal information of  $\mathbf{y}$  up to the reconstruction error (7.55).

### 7.5.3 Principal Components Analysis

PCA gives us a linear auto-encoder (7.54). If the data matrix  $\mathbf{Y} \in \mathbb{R}^{n \times q}$  has rank  $q$ , there exist  $q$  linearly independent rows of  $\mathbf{Y}$  that span  $\mathbb{R}^q$ . PCA determines a different, very specific basis of  $\mathbb{R}^q$ . It looks for an orthonormal basis  $\mathbf{v}_1, \dots, \mathbf{v}_q \in \mathbb{R}^q$  such that  $\mathbf{v}_1$  explains the direction of the biggest variability in  $\mathbf{Y}$ ,  $\mathbf{v}_2$  the direction of the second biggest variability in  $\mathbf{Y}$  orthogonal to  $\mathbf{v}_1$ , and so forth. Variability is understood in the sense of maximal empirical variance under the assumption that the columns of  $\mathbf{Y}$  are centered, see (7.52)–(7.53). Such an orthonormal basis can be found by determining  $q$  linearly independent eigenvectors of the symmetric and positive definite matrix

$$\mathbf{A} = n\widehat{\Sigma} = \mathbf{Y}^\top \mathbf{Y} \in \mathbb{R}^{q \times q}.$$

For this we can solve recursively the following convex Lagrange problems. The first basis vector  $\mathbf{v}_1 \in \mathbb{R}^q$  is determined by the solution of<sup>3</sup>

$$\mathbf{v}_1 = \arg \max_{\|\mathbf{w}\|_2=1} \|\mathbf{Y}\mathbf{w}\|_2^2 = \arg \max_{\mathbf{w}^\top \mathbf{w}=1} \left( \mathbf{w}^\top \mathbf{Y}^\top \mathbf{Y} \mathbf{w} \right), \quad (7.56)$$

and the  $j$ -th basis vector  $\mathbf{v}_j \in \mathbb{R}^q$ ,  $2 \leq j \leq q$ , is received recursively by the solution of

$$\mathbf{v}_j = \arg \max_{\|\mathbf{w}\|_2=1} \|\mathbf{Y}\mathbf{w}\|_2^2 \quad \text{subject to } \langle \mathbf{v}_k, \mathbf{w} \rangle = 0 \text{ for all } 1 \leq k \leq j-1. \quad (7.57)$$

---

<sup>3</sup> If the  $q$  eigenvalues of  $\mathbf{A}$  are distinct, the solution to (7.56) and (7.57) is unique up to the sign, otherwise this requires more care.

Singular value decomposition (SVD) gives an alternative way of computing this orthonormal basis, we refer to Section 14.5.1 in Hastie et al. [183]. The algorithm of Golub–Van Loan [165] gives an efficient way of performing a SVD. There exist orthogonal matrices  $U \in \mathbb{R}^{n \times q}$  and  $V \in \mathbb{R}^{q \times q}$  (with  $U^\top U = V^\top V = \mathbf{1}_q$ ), and a diagonal matrix  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_q) \in \mathbb{R}^{q \times q}$  with singular values  $\lambda_1 \geq \dots \geq \lambda_q > 0$  such that we have the SVD

$$\mathbf{Y} = U \Lambda V^\top. \quad (7.58)$$

The matrix  $U$  is called left-singular matrix of  $\mathbf{Y}$ , and the matrix  $V$  is called right-singular matrix of  $\mathbf{Y}$ . Observe by using the SVD (7.58)

$$V^\top A V = V^\top \mathbf{Y}^\top \mathbf{Y} V = V^\top V \Lambda U^\top U \Lambda V^\top V = \Lambda^2 = \text{diag}(\lambda_1^2, \dots, \lambda_q^2).$$

That is, the squared singular values  $(\lambda_j^2)_{1 \leq j \leq q}$  are the eigenvalues of matrix  $A$ , and the column vectors of the right-singular matrix  $V = (\mathbf{v}_1, \dots, \mathbf{v}_q)$  (eigenvectors of  $A$ ) give an orthonormal basis  $\mathbf{v}_1, \dots, \mathbf{v}_q$ . This motivates to define the  $q$  principal components of  $\mathbf{Y}$  by the column vectors of

$$\begin{aligned} \mathbf{Y} V &= U \Lambda = U \text{diag}(\lambda_1, \dots, \lambda_q) \\ &= (\lambda_1 \mathbf{u}_1, \dots, \lambda_q \mathbf{u}_q) \in \mathbb{R}^{n \times q}. \end{aligned} \quad (7.59)$$

E.g., the first principal component of the instances  $1 \leq i \leq n$  is given by  $\mathbf{Y} \mathbf{v}_1 = \lambda_1 \mathbf{u}_1 \in \mathbb{R}^n$ . Considering the first  $p \leq q$  principal components gives the rank  $p$  matrix

$$\mathbf{Y}_p = U \text{diag}(\lambda_1, \dots, \lambda_p, 0, \dots, 0) V^\top \in \mathbb{R}^{n \times q}. \quad (7.60)$$

The Eckart–Young–Mirsky theorem [114, 279]<sup>4</sup> proves that this rank  $p$  matrix  $\mathbf{Y}_p$  minimizes the Frobenius norm relative to  $\mathbf{Y}$  among all rank  $p$  matrices, that is,

$$\mathbf{Y}_p \in \arg \min_{B \in \mathbb{R}^{n \times q}} \|\mathbf{Y} - B\|_F \quad \text{subject to } \text{rank}(B) \leq p, \quad (7.61)$$

where the Frobenius norm is given by  $\|C\|_F^2 = \sum_{i,j} c_{i,j}^2$  for a matrix  $C = (c_{i,j})_{i,j}$ .

The orthonormal basis  $\mathbf{v}_1, \dots, \mathbf{v}_q \in \mathbb{R}^q$  gives the (linear) encoder (projection)

$$\Phi : \mathbb{R}^q \rightarrow \mathbb{R}^p, \quad \mathbf{y} \mapsto \Phi(\mathbf{y}) = \left( \mathbf{y}^\top \mathbf{v}_1, \dots, \mathbf{y}^\top \mathbf{v}_p \right)^\top = (\mathbf{v}_1, \dots, \mathbf{v}_p)^\top \mathbf{y}.$$

<sup>4</sup> In fact, (7.61) holds for both the Frobenius norm and the spectral norm.

These gives the first  $p$  principal components in (7.59) if we insert the transposed data matrix  $\mathbf{Y}^\top = (\mathbf{y}_1, \dots, \mathbf{y}_n) \in \mathbb{R}^{q \times n}$  for  $\mathbf{y} \in \mathbb{R}^q$ . The (linear) decoder  $\Psi$  is given by

$$\Psi : \mathbb{R}^p \rightarrow \mathbb{R}^q, \quad \mathbf{z} \mapsto \Psi(\mathbf{z}) = (\mathbf{v}_1, \dots, \mathbf{v}_p)\mathbf{z}.$$

The following is understood column-wise for the transposed data matrix  $\mathbf{Y}^\top$ ,

$$\begin{aligned} \Psi \circ \Phi(\mathbf{Y}^\top) &= \Psi \left( (\mathbf{v}_1, \dots, \mathbf{v}_p)^\top \mathbf{Y}^\top \right) \\ &= \left( \mathbf{Y}(\mathbf{v}_1, \dots, \mathbf{v}_p)(\mathbf{v}_1, \dots, \mathbf{v}_p)^\top \right)^\top \\ &= \left( \mathbf{Y}(\mathbf{v}_1, \dots, \mathbf{v}_p, 0, \dots, 0)(\mathbf{v}_1, \dots, \mathbf{v}_p, \mathbf{v}_{p+1}, \dots, \mathbf{v}_q)^\top \right)^\top \\ &= \left( U \text{diag}(\lambda_1, \dots, \lambda_p, 0, \dots, 0) V^\top \right)^\top = \mathbf{Y}_p^\top. \end{aligned}$$

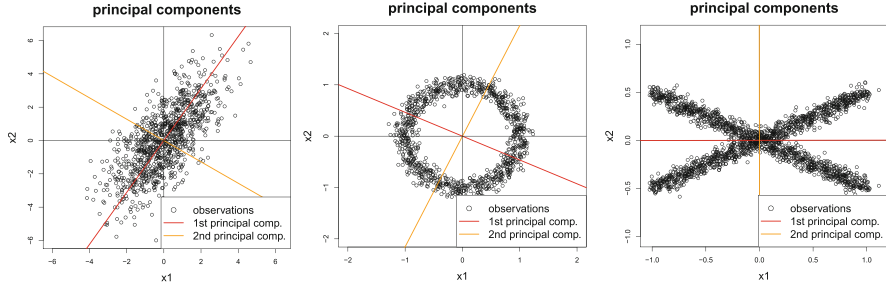
Thus,  $\Psi \circ \Phi(\mathbf{Y}^\top)$  minimizes the Frobenius reconstruction error (7.61) on the data matrix  $\mathbf{Y}^\top$  among all linear maps of rank  $p$ . In view of (7.55) we can express the squared Frobenius reconstruction error as

$$\|\mathbf{Y} - \mathbf{Y}_p\|_F^2 = \sum_{i=1}^n \|\mathbf{y}_i - \Psi \circ \Phi(\mathbf{y}_i)\|_2^2 = \sum_{i=1}^n L(\mathbf{y}_i, \Psi \circ \Phi(\mathbf{y}_i)), \quad (7.62)$$

thus, we choose the squared Euclidean distance as the dissimilarity measure, here, that we minimize simultaneously on all cases  $\mathbf{y}_i$ ,  $1 \leq i \leq n$ .

*Remark 7.30* The PCA gives a linear approximation to the data matrix  $\mathbf{Y}$  by minimizing (7.61) and (7.62) for given rank  $p$ . This may not be appropriate if the non-linear terms are dominant. Figure 7.31 (lhs) gives a situation where the PCA works well; this data has been generated by i.i.d. multivariate Gaussian random vectors  $\mathbf{y}_i \sim \mathcal{N}(\mathbf{0}, \Sigma)$ . Figure 7.31 (middle) gives a non-linear example where the PCA does not work well, the data matrix  $\mathbf{Y} \in \mathbb{R}^{n \times 2}$  is a column-centered matrix that builds a circle around the origin.

Another nice example where the PCA fails is Fig. 7.31 (rhs). This figure is inspired by Shlens [337] and Ruckstuhl [321]. It shows a situation where the level sets are non-convex, and the principal components point into a completely wrong direction to explain the structure of the data.



**Fig. 7.31** Two-dimensional PCAs in different situations of the data matrix  $Y \in \mathbb{R}^{n \times 2}$

### 7.5.4 Lab: Lee–Carter Mortality Model

We use the SVD to fit the most popular stochastic mortality model, the Lee–Carter (LC) model [238], to (raw) mortality data. The raw mortality data considers for each calendar year  $t$  and each age  $x$  the number of people  $D_{x,t}$  who died (in that year  $t$  at age  $x$ ) divided by the corresponding population exposure  $e_{x,t}$ . In practice this requires some care. Due to migration, often, the exposures  $e_{x,t}$  are non-observable figures and need to be estimated. Moreover, also the death counts  $D_{x,t}$  in year  $t$  at age  $x$  can be defined differently, age cohorts are usually defined by the year of birth. We denote the (observed) raw mortality rates by  $M_{x,t} = D_{x,t}/e_{x,t}$ . The subsequent derivations consider the raw log-mortality rates  $\log(M_{x,t})$ , for this reason we assume that  $M_{x,t} > 0$  for all calendar years  $t$  and ages  $x$ . The goal is to model these raw log-mortality rates (for each country, region, risk group and gender separately).

The LC model defines the force of mortality as

$$\log(\mu_{x,t}) = a_x + b_x k_t, \quad (7.63)$$

where  $\log(\mu_{x,t})$  is the (deterministic) log-mortality rate in calendar year  $t$  for a person aged  $x$  (for a fixed country, region and gender). The individual terms in (7.63) have the following meaning:  $a_x$  is the average force of mortality at age  $x$ ,  $b_x$  is the rate of change of the force of mortality broken down to the different ages  $x$ , and  $k_t$  is the time index describing the change of the force of mortality in calendar year  $t$ .

Strictly speaking, we do not have a stochastic model, here, that can explain the observations  $M_{x,t}$ , but we try to fit a deterministic mortality surface  $(\mu_{x,t})_{x,t}$  to these noisy observations  $(M_{x,t})_{x,t}$ . For this we use the PCA and the Frobenius norm as the measure of dissimilarity (on the log-scale).

In a first step, we center the raw log-mortality rates for all ages  $x$ , i.e., over the calendar years  $t \in \mathcal{T}$  under consideration. We define the centered raw log-mortality rates  $y_{x,t}$  and the estimate  $\hat{a}_x$  of the average force of mortality at age  $x$  as follows

$$Y_{x,t} = \log(M_{x,t}) - \hat{a}_x = \log(M_{x,t}) - \frac{1}{|\mathcal{T}|} \sum_{s \in \mathcal{T}} \log(M_{x,s}), \quad (7.64)$$

where the last identity defines the estimate  $\widehat{a}_x$ . Strictly speaking we have a slight difference to the centering in Sect. 7.5.1 because we center the rows and not the columns of the data matrix, here, but the role of rows and columns is exchangeable in the PCA. The optimal (parameter) values  $(\widehat{b}_x)_x$  and  $(\widehat{k}_t)_t$  are determined as follows, see (7.63),

$$\arg \min_{(b_x)_x, (k_t)_t} \sum_{x,t} (Y_{x,t} - b_x k_t)^2,$$

where the sum runs over the years  $t \in \mathcal{T}$  and the ages  $x_0 \leq x \leq x_1$ , with  $x_0$  and  $x_1$  being the lower and upper age boundaries. This can be rewritten as an optimization problem (7.61)–(7.62). Consider the data matrix  $\mathbf{Y} = (Y_{x,t})_{x_0 \leq x \leq x_1; t \in \mathcal{T}} \in \mathbb{R}^{n \times q}$ , and set  $n = x_1 - x_0 + 1$  and  $q = |\mathcal{T}|$ . Assume  $\mathbf{Y}$  has rank  $q$ . This allows us to consider

$$\mathbf{Y}_1 \in \arg \min_{B \in \mathbb{R}^{n \times q}} \|\mathbf{Y} - B\|_F \quad \text{subject to } \text{rank}(B) \leq 1.$$

A solution to this problem is given, see (7.60),

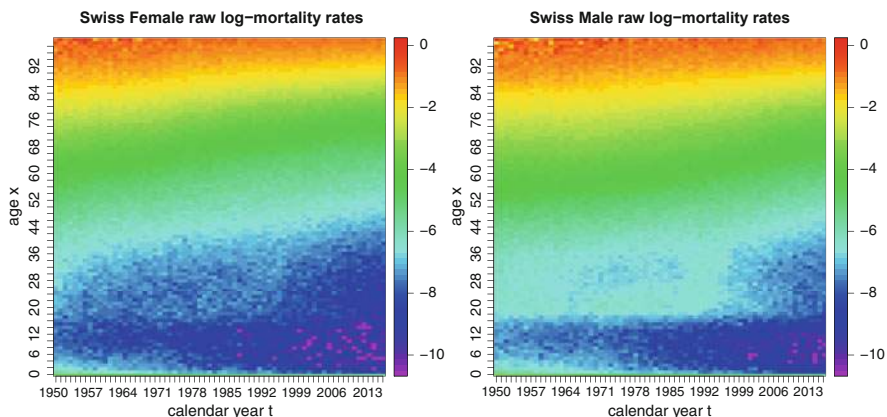
$$\mathbf{Y}_1 = U \text{diag}(\lambda_1, 0, \dots, 0) V^\top = (\lambda_1 \mathbf{u}_1) \mathbf{v}_1^\top = (\mathbf{Y} \mathbf{v}_1) \mathbf{v}_1^\top \in \mathbb{R}^{n \times q},$$

with left-singular matrix  $U = (\mathbf{u}_1, \dots, \mathbf{u}_q) \in \mathbb{R}^{n \times q}$  and right-singular matrix  $V = (\mathbf{v}_1, \dots, \mathbf{v}_q) \in \mathbb{R}^{q \times q}$  of  $\mathbf{Y}$ . This implies that the first principal component  $\lambda_1 \mathbf{u}_1 = \mathbf{Y} \mathbf{v}_1 \in \mathbb{R}^n$  gives an estimate for  $(b_x)_{x_0 \leq x \leq x_1}$ , and the first column vector  $\mathbf{v}_1 \in \mathbb{R}^q$  of  $V$  gives an estimate for the time index  $(k_t)_{t \in \mathcal{T}}$ . For parameter identifiability we normalize

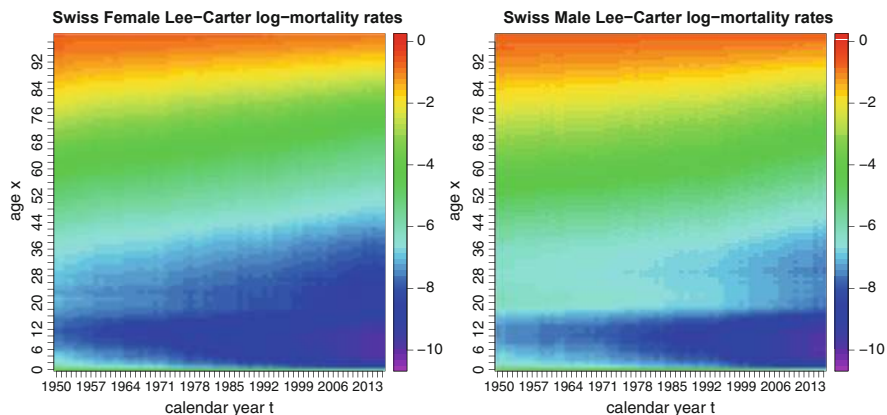
$$\sum_{x=x_0}^{x_1} \widehat{b}_x = 1 \quad \text{and} \quad \sum_{t \in \mathcal{T}} \widehat{k}_t = 0, \quad (7.65)$$

the latter being consistent with the centering of the rows of  $\mathbf{Y}$  with  $\widehat{a}_x$  in (7.64).

We fit the LC model to the Swiss mortality data of females and males separately. The raw log-mortality rates  $\log(M_{x,t})$  for the years  $t \in \mathcal{T} = \{1950, \dots, 2016\}$  and the ages  $0 \leq x \leq 99$  are illustrated in Fig. 7.32; both plots use the same color scale. This mortality data has been obtained from the Human Mortality Database (HMD) [195]. In general, we observe a diagonal structure that indicates mortality improvements over time.



**Fig. 7.32** Raw log-mortality rates  $\log(M_{x,t})$  for the calendar years  $1950 \leq t \leq 2016$  and the ages  $x_0 = 0 \leq x \leq x_1 = 99$  of Swiss females (lhs) and Swiss males (rhs); both plots use the same color scale

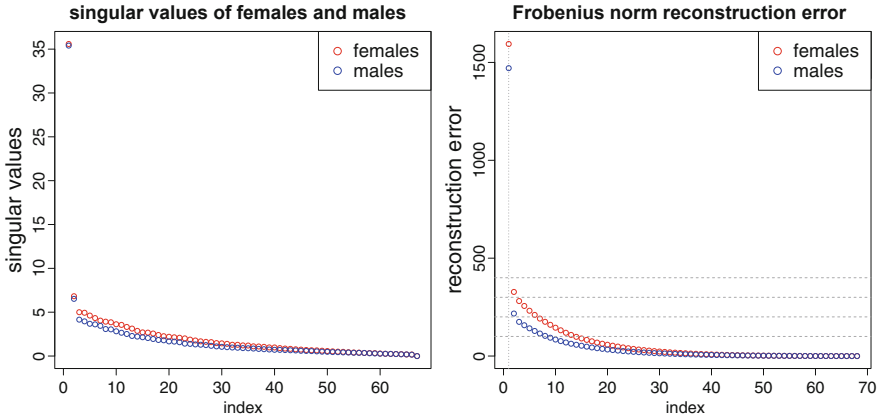


**Fig. 7.33** LC fitted log-mortality rates  $\log(\hat{\mu}_{x,t})$  for the calendar years  $1950 \leq t \leq 2016$  and the ages  $x_0 = 0 \leq x \leq x_1 = 99$  of Swiss females (lhs) and Swiss males (rhs); the plots use the same color scale as Fig. 7.32

Define the fitted log-mortality surface

$$\log(\hat{\mu}_{x,t}) = \hat{a}_x + \hat{b}_x \hat{k}_t \quad \text{for } x_0 \leq x \leq x_1 \text{ and } t \in \mathcal{T}.$$

Figure 7.33 shows the LC fitted log-mortality surface  $(\log(\hat{\mu}_{x,t}))_{0 \leq x \leq 99; t \in \mathcal{T}}$  separately for Swiss females and Swiss males, the color scale is the same as in Fig. 7.32. The plots show a huge similarity between the raw log-mortality data and the LC fitted log-mortality surface which clearly supports the LC model for the Swiss data. In general, the LC surface is a smoothed version of the raw log-mortality surface. The main difference in our LC fit concerns the male population for ages



**Fig. 7.34** (lhs) Singular values  $\lambda_j$ ,  $1 \leq j \leq |\mathcal{T}|$ , of the SVD of the data matrix  $\mathbf{Y} \in \mathbb{R}^{n \times |\mathcal{T}|}$ , and (rhs) the reconstruction errors  $\|\mathbf{Y} - \mathbf{Y}_p\|_F^2$  for  $0 \leq p \leq |\mathcal{T}|$

$20 \leq x \leq 40$  from 1980 to 2000, one explanation of the special pattern in the observed data during that time is the emergence of HIV.

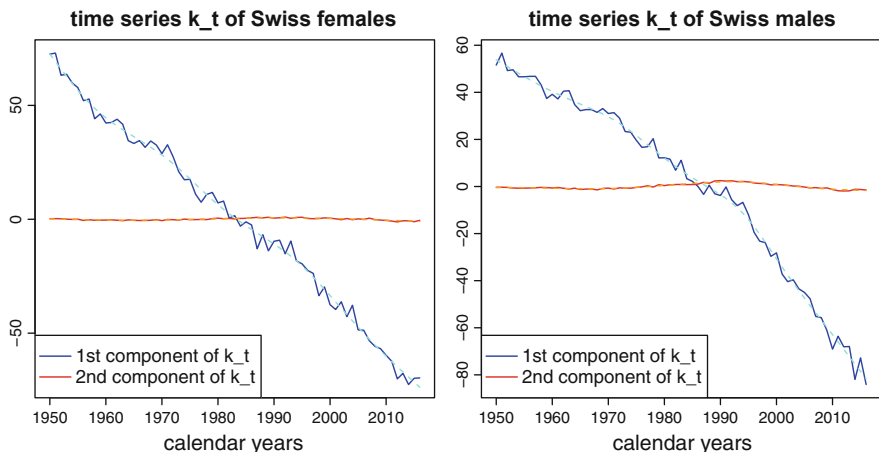
Figure 7.34 (lhs) shows the singular values  $\lambda_1 \geq \dots \geq \lambda_{|\mathcal{T}|} > 0$  for Swiss females and Swiss males. We observe that the first singular value  $\lambda_1$  by far dominates the remaining singular values  $\lambda_j$ ,  $j \geq 2$ . Thus, the first principal component indeed may already be sufficient, and the centered raw log-mortality data  $\mathbf{Y}$  can be described by a matrix  $\mathbf{Y}_1$  of rank  $p = 1$ . Figure 7.34 (rhs) gives the squared Frobenius reconstruction errors of the approximations  $\mathbf{Y}_p$  of ranks  $0 \leq p \leq |\mathcal{T}|$ , where  $\mathbf{Y}_0$  corresponds to the zero matrix where we do not use any approximation, but use just the average observed log-mortality rate. We observe that the first singular value leads by far to the biggest decrease in the reconstruction error, and the subsequent expansions  $\lambda_j$ ,  $j \geq 2$ , improve it only slightly in each step. This supports the use of the LC model using a rank  $p = 1$  approximation to the centered raw log-mortality rates  $\mathbf{Y}$ . The higher rank PCA within mortality modeling has been studied in Renshaw–Haberman (RH) [308], and the  $\text{RH}(p)$  mortality model considers the rank  $p$  approximation  $\mathbf{Y}_p$  to the raw log-mortality rates  $\mathbf{Y}$  given by

$$\log(\mu_{x,t}) = a_x + \langle \mathbf{b}_x, \mathbf{k}_t \rangle,$$

for  $\mathbf{b}_x, \mathbf{k}_t \in \mathbb{R}^p$ .

We have (only) fitted a mortality surface to the raw log-mortality rates on the rectangle  $\{x_0, \dots, x_1\} \times \mathcal{T}$ . This does not allow us to forecast mortality into the future. Forecasting requires a two step procedure, which, after this first estimation step, extrapolates the time index (time-series)  $(\hat{k}_t)_{t \in \mathcal{T}}$  beyond the latest observation point in  $\mathcal{T}$ . The simplest (meaningful) model for this second (extrapolation) step is a random walk with drift for the time index process  $(\hat{k}_t)_{t \geq 0}$ . Figure 7.35 shows the estimated two-dimensional process  $(\hat{\mathbf{k}}_t)_{t \in \mathcal{T}}$ , i.e., for  $p = 2$ , on the rectangle





**Fig. 7.35** Estimated two-dimensional processes  $(\hat{k}_t)_{t \in \mathcal{T}}$  for Swiss females (lhs) and Swiss males (rhs); these are normalized such that they are centered and such that the components of  $\hat{\mathbf{b}}_x$  add up to 1

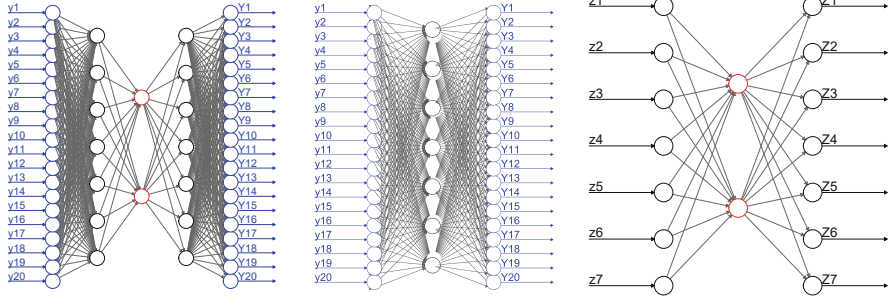
$\{x_0, \dots, x_1\} \times \mathcal{T}$  which needs to be extrapolated to predict within the RH ( $p = 2$ ) mortality model. We refrain from doing this step, but extrapolation will be studied in Sect. 8.4, below.

### 7.5.5 Bottleneck Neural Network

BN networks have become popular in studying non-linear generalizations of PCA, we refer to Kramer [225] and Hinton–Salakhutdinov [186]. The BN network architecture is such that (1) the input dimension  $q_0$  is equal to the output dimension  $q_{d+1}$  of a FN network, and (2) in between there is a FN layer  $1 \leq m \leq d$  that has a very low dimension  $q_m \ll q_0$ , called the bottleneck. Figure 7.36 (lhs) shows such a BN network of depth  $d = 3$  and neurons

$$(q_0, q_1, q_2, q_3, q_4) = (20, 7, 2, 7, 20).$$

The input and output neurons have blue color, and the bottleneck of dimension  $q_2 = 2$  is shown in red color in Fig. 7.36 (lhs).



**Fig. 7.36** (lhs) BN network of depth  $d = 3$  with  $(q_0, q_1, q_2, q_3, q_4) = (20, 7, 2, 7, 20)$ , (middle and rhs) shallow BN networks with a bottleneck of dimensions 7 and 2, respectively

The motivation is as follows. Assume we have a given dissimilarity function  $L(\cdot, \cdot) : \mathbb{R}^q \times \mathbb{R}^q \rightarrow \mathbb{R}_+$  that measures the reconstruction error of an auto-encoder  $\Psi \circ \Phi(\mathbf{y}) \in \mathbb{R}^q$  relative to the original input  $\mathbf{y} \in \mathbb{R}^q$ , see (7.55). We try to find a BN network with input and output dimensions  $q_0 = q_{d+1} = q$  (we drop the intercepts in the entire construction) and a bottleneck in layer  $m$  having a low dimension  $q_m$ , such that the BN network provides a small reconstruction error. Choose a FN network

$$\mathbf{y} \in \mathbb{R}^q \mapsto \Psi \circ \Phi(\mathbf{y}) = \mathbf{z}^{(d+1:1)}(\mathbf{y}) = \left( \mathbf{z}^{(d+1)} \circ \mathbf{z}^{(d)} \circ \dots \circ \mathbf{z}^{(1)} \right)(\mathbf{y}) \in \mathbb{R}^q,$$

with FN layers for  $1 \leq m \leq d$  (excluding intercepts)

$$\mathbf{z}^{(m)} : \mathbb{R}^{q_{m-1}} \rightarrow \mathbb{R}^{q_m}, \quad \mathbf{z} \mapsto \mathbf{z}^{(m)}(\mathbf{z}) = \left( \phi(\langle \mathbf{w}_1^{(m)}, \mathbf{z} \rangle), \dots, \phi(\langle \mathbf{w}_{q_m}^{(m)}, \mathbf{z} \rangle) \right)^\top,$$

and having network weights  $\mathbf{w}_j^{(m)} \in \mathbb{R}^{q_{m-1}}$ ,  $1 \leq j \leq q_m$ . For the output we choose the identity function as activation function

$$\mathbf{z}^{(d+1)} : \mathbb{R}^{q_d} \rightarrow \mathbb{R}^{q_{d+1}}, \quad \mathbf{z} \mapsto \mathbf{z}^{(d+1)}(\mathbf{z}) = \left( \langle \mathbf{w}_1^{(d+1)}, \mathbf{z} \rangle, \dots, \langle \mathbf{w}_{q_{d+1}}^{(d+1)}, \mathbf{z} \rangle \right)^\top,$$

and having network weights  $\mathbf{w}_j^{(d+1)} \in \mathbb{R}^{q_d}$ ,  $1 \leq j \leq q_{d+1}$ . The resulting network parameter  $\boldsymbol{\vartheta}$  is now fitted to the data matrix  $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)^\top \in \mathbb{R}^{n \times q}$  such that the reconstruction error is minimized over all instances

$$\hat{\boldsymbol{\vartheta}} = \arg \min_{\boldsymbol{\vartheta} \in \mathbb{R}^r} \sum_{i=1}^n L(\mathbf{y}_i, \Psi \circ \Phi(\mathbf{y}_i)) = \arg \min_{\boldsymbol{\vartheta} \in \mathbb{R}^r} \sum_{i=1}^n L(\mathbf{y}_i, \mathbf{z}^{(d+1:1)}(\mathbf{y}_i)).$$

We use this fitted network parameter  $\hat{\boldsymbol{\vartheta}}$  and denote the resulting FN layers by  $\hat{\mathbf{z}}^{(m)}$  for  $1 \leq m \leq d + 1$ .

This allows us to define the BN encoder, set  $q = q_0$  and  $p = q_m$ ,

$$\Phi : \mathbb{R}^{q_0} \rightarrow \mathbb{R}^{q_m}, \quad \mathbf{y} \mapsto \Phi(\mathbf{y}) = \hat{\mathbf{z}}^{(m:1)}(\mathbf{y}) = \left( \hat{\mathbf{z}}^{(m)} \circ \dots \circ \hat{\mathbf{z}}^{(1)} \right)(\mathbf{y}), \quad (7.66)$$

and the BN decoder is given by, set  $q_m = p$  and  $q_{d+1} = q$ ,

$$\Psi : \mathbb{R}^{q_m} \rightarrow \mathbb{R}^{q_{d+1}}, \quad \mathbf{z} \mapsto \Psi(\mathbf{z}) = \hat{\mathbf{z}}^{(d+1:m+1)}(\mathbf{z}) = \left( \hat{\mathbf{z}}^{(d+1)} \circ \dots \circ \hat{\mathbf{z}}^{(m+1)} \right)(\mathbf{z}).$$

The BN encoder (7.66) gives us a  $q_m$ -dimensional representation of the data. A linear rank  $p$  representation  $\mathbf{Y}_p$  of  $\mathbf{Y}$ , see (7.61), can be found by a BN network architecture that has a minimal FN layer width of dimension  $p = \min_{1 \leq j \leq d} q_j$ , and with the identity activation function  $\phi(x) = x$ . Such a BN network is a linear map of maximal rank  $p$ . Using the Euclidean square distance as dissimilarity measure provides us an optimal network parameter  $\hat{\boldsymbol{\theta}}$  for this linear map such that we receive  $\mathbf{Y}_p^\top = \hat{\mathbf{z}}^{(d+1:1)}(\mathbf{Y}^\top)$ . There is one point to be considered, here, why the bottleneck activations  $\Phi(\mathbf{y}) = \hat{\mathbf{z}}^{(m:1)}(\mathbf{y}) \in \mathbb{R}^p$  in the linear activation case are not directly comparable to the principal components  $(\mathbf{y}^\top \mathbf{v}_1, \dots, \mathbf{y}^\top \mathbf{v}_p)^\top$  of the PCA. Namely, the PCA uses an orthonormal basis  $\mathbf{v}_1, \dots, \mathbf{v}_p$  whereas the linear BN network case uses any  $p$ -dimensional basis, i.e., to directly bring these two representations in line we still need a coordinate transformation of the bottleneck activations.

Hinton–Salakhutdinov [186] noticed that the gradient descent fitting of a BN network needs some care, otherwise we may find a local minimum of the loss function that has a poor reconstruction performance. In order to implement a more sophisticated way of SGD fitting we require that the depth  $d$  of the network is an odd number and that the network architecture is symmetric around the central FN layer  $(d+1)/2$ . This is the case in Fig. 7.36 (lhs). Fitting of this network of depth  $d = 3$  is now done in three steps:

1. The symmetry around the central FN layer  $m = 2$  allows us to collapse this central layer by merging layers 1 and 3 (because  $q_1 = q_3$ ). Merging these two layers provides us a shallow BN network with neurons  $(q_0, q_1 = q_3, q_{d+1} = q_0) = (20, 7, 20)$ . This shallow BN network is shown in Fig. 7.36 (middle). In a first step we fit this simpler network to the data  $\mathbf{Y}$ . This gives us the preliminary estimates for the network weights  $\mathbf{w}_1^{(1)}, \dots, \mathbf{w}_{q_1}^{(1)}$  and  $\mathbf{w}_1^{(4)}, \dots, \mathbf{w}_{q_4}^{(4)}$  of the full BN network. From this fitted shallow BN network we receive the learned representations  $\mathbf{z}_i = \mathbf{z}^{(1)}(\mathbf{y}_i) \in \mathbb{R}^{q_1}$ ,  $1 \leq i \leq n$ , in the central layer using the preliminary estimates of the network weights.
2. In the second step we use the learned representations  $\mathbf{z}_i \in \mathbb{R}^{q_1}$ ,  $1 \leq i \leq n$ , to fit the inner part of the original network (using a suitable dissimilarity function). This inner part is a shallow network with neurons  $(q_1, q_2, q_3 = q_1) = (7, 2, 7)$ ,

see Fig. 7.36 (rhs). This second step gives us the preliminary estimates for the network weights  $\mathbf{w}_1^{(2)}, \dots, \mathbf{w}_{q_2}^{(2)}$  and  $\mathbf{w}_1^{(3)}, \dots, \mathbf{w}_{q_3}^{(3)}$  of the full BN network.

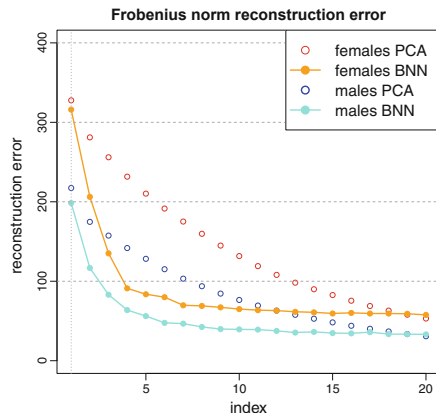
3. In the final step we fit the full BN network on the data  $\mathbf{Y}$  and use the preliminary estimates of the weights (of the previous two steps) as initialization of the gradient descent algorithm.

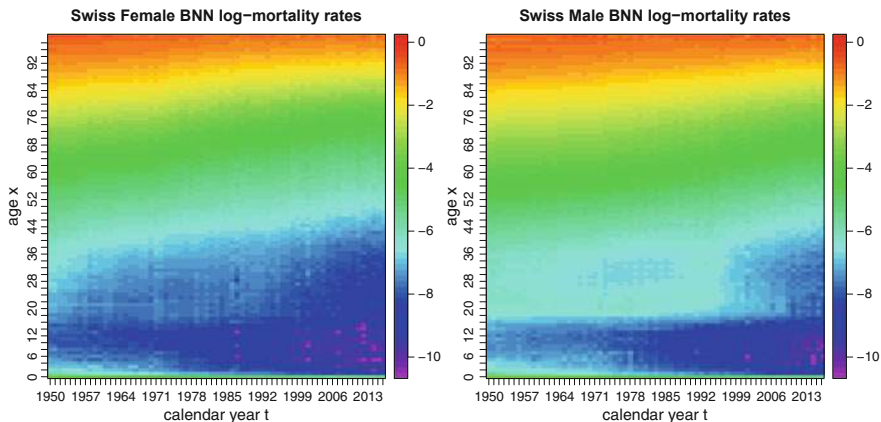
*Example 7.31 (BN Network Mortality Model)* We apply this BN network approach to modify the LC model of Sect. 7.5.4. Hainaut [178] considered such a BN network application. For computational reasons, Hainaut [178] proposed a calibration strategy different from Hinton–Salakhutdinov [186]. We use this latter calibration strategy as it has turned out to work well in our setting.

As BN network architecture we choose a FN network of depth  $d = 3$ . The input and output dimensions are equal to  $q_0 = q_4 = 67$ , this exactly corresponds to the number of available calendar years  $1950 \leq t \leq 2016$ , see Fig. 7.32. Then, we select a symmetric architecture around the central FN layer  $m = 2$  with  $q_1 = q_3 = 20$  neurons. That is, in a first step, the 67 calendar years are compressed to a 20-dimensional representation. For the bottleneck we then explore different numbers of neurons  $q_2 = p \in \{1, \dots, 20\}$ . These BN networks are implemented and fitted in R with the library *keras* [77]. We have fitted these models separately to the Swiss female and male populations. The raw log-mortality rates are illustrated in Fig. 7.32, and for comparability with the LC approach we have centered these log-mortality rates according to (7.64), and we use the squared Euclidean distance as the objective function.

Figure 7.37 compares the squared Frobenius reconstruction errors of the linear LC approximations  $\mathbf{Y}_p$  to their non-linear BN network counterparts with bottlenecks  $q_2 = p$ . We observe that the BN figures are clearly smaller saying that a non-linear auto-encoding provides a better reconstruction, this is true, in particular, for  $2 \leq q_2 < 20$ . For  $q_2 \geq 20$  the learning with the BN networks seems saturated, note that the outer layers have  $q_1 = q_3 = 20$  neurons which limits the learning at the bottleneck for bigger  $q_2$ . In view of Fig. 7.37 there seems to be a kink at  $q_2 = 4$ ,

**Fig. 7.37** Frobenius reconstruction errors  $\|\mathbf{Y} - \mathbf{Y}_p\|_F^2$  for  $1 \leq p = q_2 \leq 20$  in the linear LC approach and the non-linear BN approach





**Fig. 7.38** BN network  $(q_1, q_2, q_3) = (20, 2, 20)$  fitted log-mortality rates  $\log(\hat{\mu}_{x,t})$  for the calendar years  $1950 \leq t \leq 2016$  and the ages  $x_0 = 0 \leq x \leq x_1 = 99$  of Swiss females (left) and Swiss males (right); the plots use the same color scale as Fig. 7.32

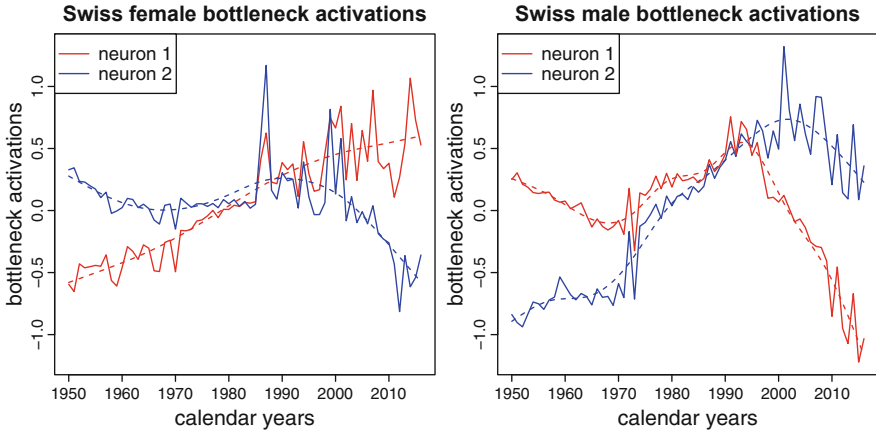
and an “elbow” criterion says that this is the critical bottleneck size that should not be exceeded.

The resulting estimated log-mortality surfaces for the bottleneck  $q_2 = 2$  are illustrated in Fig. 7.38. These strongly resemble the raw log-mortality rates in Fig. 7.32, in particular, for the male population we get a better fit for ages  $20 \leq x \leq 40$  from 1980 to 2000 compared to the LC model. In a further analysis we should check whether this BN network does not over-fit to the data. We could, e.g., explore drop-outs during calibration or smaller FN (compression) layers  $q_1 = q_3$ .

Finally, we analyze the resulting activations at the bottleneck by considering the BN encoder (7.66). Note that we assume  $\mathbf{y} \in \mathbb{R}^q$  in (7.66) with  $q = |\mathcal{T}|$  being the rank of the data matrix  $\mathbf{Y} \in \mathbb{R}^{n \times q}$ . Thus, the encoder takes a fixed age  $0 \leq x \leq 99$  and encodes the corresponding time-series observation  $\mathbf{y}_x \in \mathbb{R}^{|\mathcal{T}|}$  by the bottleneck activations. This parametrization has been inspired by the PCA which typically considers a data matrix that has more rows than columns. This results in at most  $q = \text{rank}(\mathbf{Y})$  singular values, supposed  $n \geq q$ . However, we can easily exchange the role of rows and columns, e.g., by transposing all matrices involved. For mortality forecasting it is advantageous to exchange these roles because we would like to extrapolate a time-series beyond  $\mathcal{T}$ . For this reason we set for the input dimension  $q_0 = q = 100$ , which provides us with  $|\mathcal{T}|$  observations  $\mathbf{y}_t \in \mathbb{R}^{100}$ . We then fit the BN encoder (7.66) to receive the bottleneck activations

$$\mathbf{Y} = (\mathbf{y}_t)_{t \in \mathcal{T}} \mapsto \Phi(\mathbf{Y}) = (\Phi(\mathbf{y}_t))_{t \in \mathcal{T}} \in \mathbb{R}^{q_2 \times |\mathcal{T}|}.$$

Figure 7.39 shows these figures for a bottleneck  $q_2 = 2$ . We observe that these bottleneck time-series  $(\Phi(\mathbf{y}_t))_{t \in \mathcal{T}}$  are much more difficult to understand than the LC/RH ones given in Fig. 7.35. Firstly, we see that we have quite some dependence



**Fig. 7.39** BN network  $(q_1, q_2, q_3) = (20, 2, 20)$ : bottleneck activations showing  $\Phi(y_t) \in \mathbb{R}^2$  for  $t \in \mathcal{T}$

between the components of the time-series. Secondly, in contrast to the LC/RH case of Fig. 7.35, there is not one component that dominates. Note that this dominance has been obtained by scaling the components of  $(b_x)_x$  to add up to 1 (which, of course, reflects the magnitudes of the singular values). In the non-linear case, these scales are hidden in the decoder which is more difficult to extract. Thirdly, the extrapolation may not work if the time-series has a trend and if we use the hyperbolic tangent activation function that has a bounded range. In general, a trend extrapolation has to be considered very carefully with FN networks with non-linear activation functions, and often there is no good solution to this problem within the FN network framework. We conclude that this approach improves in-sample mortality surface modeling, but it leaves open the question about forecasting the future mortality rates because an extrapolation seems more difficult. ■

*Remark 7.32* The concept of BN networks has also been considered in the actuarial literature to encode geographic information, see Blier-Wong et al. [39]. Since geographic information has a natural spatial component, these authors propose to use a convolutional neural network to encode the spatial information before processing the learned features through a BN network. The proposed decoder may have different forms, either it tries to reconstruct the whole (spatial) neighborhood of a given location or it only tries to reconstruct the site of a given location.

## 7.6 Model-Agnostic Tools

We collect some model-agnostic tools in this section that help us to better understand and analyze the networks, their calibrations and predictions. Model-agnostic tools are techniques that are not specific to a certain model type and can be used for any regression model. Most methods presented here are nicely presented in the tutorial of Lorentzen–Mayer [258]. There are several ways of getting a better understanding of a regression model. First, we can analyze variable importance which tries to answer similar questions to the GLM variable selection tools of Sect. 5.3 on model validation. However, in general, we cannot rely on any asymptotic likelihood theory for such an analysis. Second, we can try to understand the predictive model. For a GLM with the log-link function this is quite simple because the systematic effects are of a multiplicative nature. For networks this is much more complicated because we allow for much more general regression functions. We can either try to understand these functions on a global portfolio level (by averaging the effects over many insurance policies) or we can try to understand these functions locally for individual insurance policies. The latter refers to local sensitivities around a chosen feature value  $\mathbf{x} \in \mathcal{X}$ , and the former to global model-agnostics.

### 7.6.1 Variable Permutation Importance

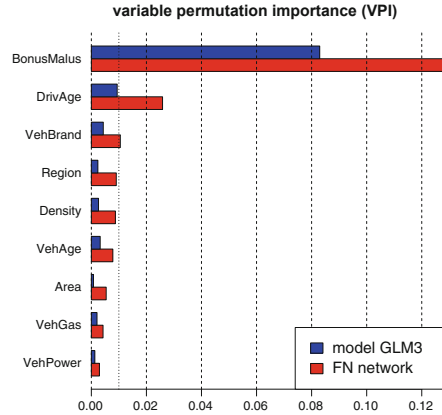
For GLMs we have studied the LRT and the Wald test that have been assisting us in reducing the GLM by the feature components that do not contribute sufficiently to the regression task at hand, see Sects. 5.3.2 and 5.3.3. These variable reduction techniques rely on an asymptotic likelihood theory. Here, we need to proceed differently, and we just aim at ranking the variables by their importance, similarly to a `drop1` analysis, see Listing 5.6.

For a given FN network regression model

$$\mathbf{x} \in \mathcal{X} \mapsto \mu(\mathbf{x}) = g^{-1} \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle,$$

we randomize one component of  $\mathbf{x} = (x_1, \dots, x_q)^\top$  at a time, and we study the resulting change in the objective function. More precisely, for given (learning) data  $\mathcal{L}$ , with features  $\mathbf{x}_1, \dots, \mathbf{x}_n$ , we select one feature component  $1 \leq j \leq q$  and permute  $(x_{i,j})_{1 \leq i \leq n}$  randomly across the entire portfolio  $1 \leq i \leq n$ . We denote by  $\mathcal{L}^{(j)}$  the resulting data with the  $j$ -th component being permuted. We then compare the resulting deviance loss  $\mathfrak{D}(\mathcal{L}^{(j)}, \mu)$  to the one  $\mathfrak{D}(\mathcal{L}, \mu)$  on the original data  $\mathcal{L}$  using the same regression model  $\mu$ . We call this approach variable permutation importance (VPI). Note that such a permutation does not only act on the marginal effects, but it also distorts the interaction effects of the different feature components.

**Fig. 7.40** VPI measured by the relative change  $\text{vpi}^{(j)}$ ,  $1 \leq j \leq q$ , of model Poisson GLM3 of Table 5.5 and the FN network regression model  $\hat{\mu}^{m=1}$  of Table 7.9



We calculate the VPI on the MTPL claim frequency data of model Poisson GLM3 of Table 5.5 and the FN network regression model  $\hat{\mu}^{m=1}$  of Table 7.9; we use this example throughout this section on model-agnostic tools. Figure 7.40 shows the relative increases

$$\text{vpi}^{(j)} = \frac{\mathcal{D}(\mathcal{L}^{(j)}, \mu) - \mathcal{D}(\mathcal{L}, \mu)}{\mathcal{D}(\mathcal{L}, \mu)},$$

of the deviance losses by permuting one feature component  $1 \leq j \leq q$  at a time.

Obviously, the BonusMalus level followed by DrivAge and VehBrand are the most important variables according to this VPI method. This is in alignment for both models. Thereafter, there are smaller disagreements between the two models. These disagreements may (also) be caused by a non-optimal feature pre-processing in the GLM where, for instance, we have to add the interaction effects manually, see (5.35). Overall, these VPI results are in line with the findings of the classical methods on GLMs, see for instance the `drop1` table in Listing 5.6.

One point that is worth mentioning (and which makes the VPI results not fully reliable) is the use of feature components that are highly correlated. In our case, Density and Area are highly correlated, see Fig. 13.12. Therefore, it may not make sense to randomly permute one component while keeping the other one unchanged. This issue will also arise in other methods described below.

**Remark 7.33 (Global Surrogate Model)** There are other machine learning methods that offer different measures of variable importance. For instance, (binary split) classification and regression trees (CARTs) offer popular methods for measuring variable importance; for binary split CARTs we refer to Breiman et al. [54] and Denuit et al. [100]. These CARTs select individual feature components for partitioning the feature space  $\mathcal{X}$ , and variable importance is measured by analyzing the contribution of each feature component to the total decrease of the objective



function. Binary split CARTs have the advantage that this can be done in an additive way.

More complex regression models like FN networks can then be analyzed by using a binary split regression tree as a global surrogate model. That is, we can fit a CART to the network regression function (as a surrogate model) and then analyze variable importance in this surrogate regression tree model using the tools of regression trees. We will not give an explicit example here because we have not formally introduced regression trees in this manuscript, but this concept is fairly straightforward and well-understood.

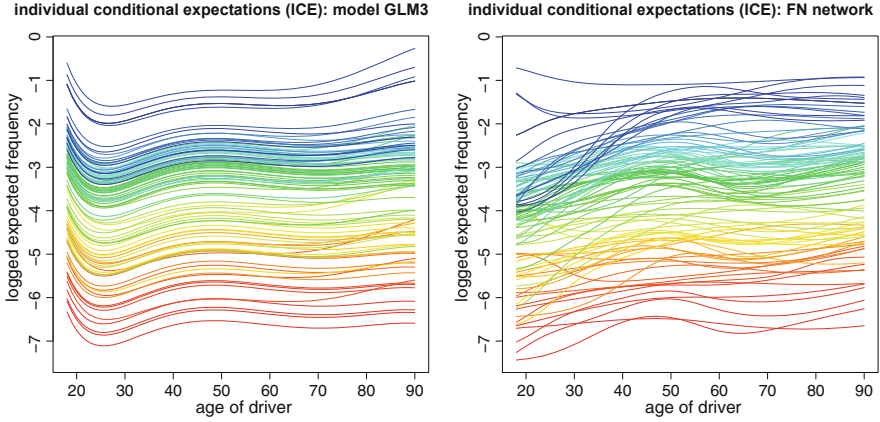
### 7.6.2 *Partial Dependence Plots*

There are several graphical tools that study the individual behavior in the feature components. Some of these tools select individual insurance policies and others study global portfolio properties. They have in common that they are based on marginal considerations, i.e., some sort of projection.

#### **Individual Conditional Expectation**

Individual conditional expectation (ICE) selects individual insurance policies  $(Y_i, \mathbf{x}_i, v_i)$  and varies the feature components of  $\mathbf{x}_i$  over their entire domain; we refer to Goldstein et al. [164]. Similarly to the VPI of Sect. 7.6.1, ICE does not respect collinearity in feature components, but it is rather an isolated view of individual components.

In Fig. 7.41 we provide the ICE plots of model Poisson GLM3 of Table 5.5 and the FN network regression model  $\hat{\mu}^{m=1}$  of Table 7.9 of 100 randomly selected insurance policies  $\mathbf{x}_i$ . For these randomly selected insurance policies we let the variable `DrivAge` vary over its domain  $\{18, \dots, 90\}$ . Each color corresponds to one insurance policy  $i$ , and the colors in the two plots coincide. In the GLM we observe that the lines are roughly parallel which reflects that we have an additive regression structure on the canonical scale (note that these plots are on the canonical parameter scale). The lines are not perfectly parallel because we allow for an interaction between `DrivAge` and `BonusMalus` in model Poisson GLM3, see (5.35). The plot of the FN network is more difficult to interpret. Overall the levels (colors) coincide in the two plots, but in the FN network plot the lines are not increasing for ages approaching 18, the reason for this is that we have interactions with other feature components that are important. In particular, for ages close to 18 we cannot have a `BonusMalus` level of 50% and, therefore, the FN network cannot be trained on this part of the feature space. Nevertheless, the ICE plot allows for such feature configurations (by just extrapolating the FN network regression function beyond the set of available insurance policies). This difficulty is confirmed



**Fig. 7.41** ICE plots of 100 randomly selected insurance policies  $\mathbf{x}_i$  of (lhs) model Poisson GLM3 and (rhs) FN network  $\hat{\mu}^{m=1}$  letting the variable `DrivAge` vary over its domain; the y-axis is on the canonical parameter scale

by exploiting the same plot only on insurance policies that have a `BonusMalus` level of at least 100%. In that case the lines for small ages are non-decreasing when approaching the age of 18, thus, providing a more reasonable interpretation. We conclude that if we have strong dependence and/or interactions between the feature components this method may not provide any reasonable interpretations.

### Partial Dependence Plot

Partial dependence plots (PDPs) have been introduced by Friedman [141], see also Zhao–Hastie [405]. PDPs are closely related to the do-operator in causal inference in statistics; we refer to Pearl [298] and Pearl et al. [299] for the do-operator. A PDP and the do-operator, respectively, are obtained by breaking the dependence structure between different feature components. Namely, we decompose the feature  $\mathbf{x} = (x_j, \mathbf{x}_{\setminus j})$  into two parts with  $\mathbf{x}_{\setminus j}$  denoting all feature components except of component  $x_j$ ; we will use a slight abuse of notation because the components need to be permuted correspondingly in the following regression function  $\mathbf{x} \rightarrow \mu(\mathbf{x}) = \mu(x_j, \mathbf{x}_{\setminus j})$ . Since, typically, there is dependence between  $x_j$  and  $\mathbf{x}_{\setminus j}$  one can infer  $\mathbf{x}_{\setminus j}$  from  $x_j$ , and vice versa. A PDP breaks this inference potential so that the sensitivity can be studied purely in  $x_j$ . In particular, the partial dependence profile is obtained by

$$x_j \mapsto \bar{\mu}^j(x_j) = \int \mu(x_j, \mathbf{x}_{\setminus j}) dp(\mathbf{x}_{\setminus j}), \quad (7.67)$$

where  $p(\mathbf{x}_{\setminus j})$  is the marginal (portfolio) distribution of the feature components  $\mathbf{x}_{\setminus j}$ . Observe that this differs from the conditional expectation which reads as

$$x_j \mapsto \mu(x_j) = \mathbb{E}_p [\mu(x_j, \mathbf{x}_{\setminus j}) | x_j] = \int \mu(x_j, \mathbf{x}_{\setminus j}) dp(\mathbf{x}_{\setminus j} | x_j),$$

the latter allowing for inferring  $\mathbf{x}_{\setminus j}$  from  $x_j$  through the conditional probability  $dp(\mathbf{x}_{\setminus j} | x_j)$ .

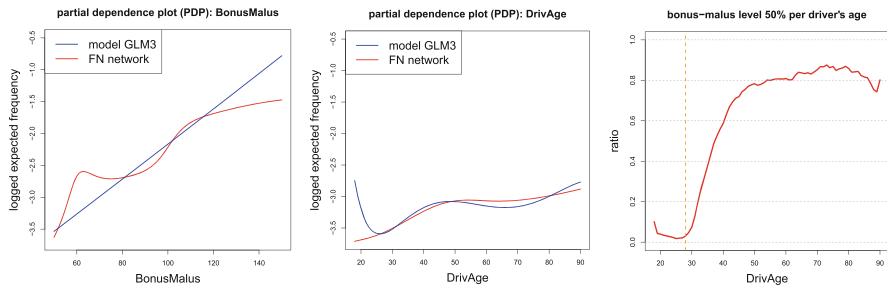
*Remark 7.34 (Discrimination-Free Insurance Pricing)* Recent actuarial literature discusses discrimination-free insurance pricing which aims at developing a pricing framework that is free of discrimination w.r.t. so-called protected characteristics such as gender and ethnicity; we refer to Guillén [174], Chen et al. [69, 70], Lindholm et al. [253] and Frees–Huang [136] for discussions on discrimination in insurance. In general, part of the problem also lies in the fact that one can often infer the protected characteristics from the non-protected feature information. This is called indirect discrimination or proxy discrimination. The proposal of Lindholm et al. [253] for achieving discrimination-free prices exactly follows the construction (7.67), by breaking the link, which infers the protected characteristics from the non-protected ones.

The partial dependence profile on our portfolio  $\mathcal{L}$  with given features  $\mathbf{x}_1, \dots, \mathbf{x}_n$  is now obtained by just using the portfolio distribution as an empirical distribution for  $p$  in (7.67). That is, for a selected component  $x_j$  of  $\mathbf{x}$ , we consider the partial dependence profile

$$x_j \mapsto \bar{\mu}^j(x_j) = \frac{1}{n} \sum_{i=1}^n \mu(x_j, \mathbf{x}_{i,\setminus j}) = \frac{1}{n} \sum_{i=1}^n \mu(x_{i,0}, x_{i,1}, \dots, x_{i,j-1}, x_j, x_{i,j+1}, \dots, x_{i,q}),$$

thus, we average the ICE plots over  $\mathbf{x}_{i,\setminus j}$  of our portfolio  $1 \leq i \leq n$ .

Figure 7.42 (lhs, middle) give the PDPs of the variables BonusMalus and DrivAge of model Poisson GLM3 and the FN network  $\hat{\mu}^{m=1}$ . Overall they



**Fig. 7.42** PDPs of (lhs) BonusMalus level and (middle) DrivAge; the y-axis is on the canonical parameter scale; (rhs) ratio of policies with a bonus-malus level of 50% per driver's age

look reasonable. However, we are again facing the difficulty that these partial dependence profiles consider feature configurations that should not appear in our portfolio. Roughly 57% of all insurance policies have a bonus-malus level of 50%, which means that these driver's did not suffer any claims in the past couple of years. Obviously a driver of age 18 cannot be on this bonus-malus level, simply because she/he is not in a state where she/he can have multiple years of driving experience without an accident. However, the PDP does not respect this fact, and just extrapolates the regression function into that part of the feature space. Therefore, the PDP at driver's age 18 is based on 57% of the insurance policies being on a bonus-malus level of 50% because this corresponds to the empirical portfolio distribution  $p(\mathbf{x}_{\setminus j})$  excluding the driver's age  $x_j = \text{DrivAge}$  information. Figure 7.42 (rhs) shows the ratio of insurance policies that have a bonus-malus level of 50%. We observe that this ratio is roughly zero up to age 28 (orange vertical dotted line), which indicates that a driver needs 10 successive accident-free years to reach the lowest bonus-malus level (starting from 100%). We consider it to be data error that this ratio below age 28 is not identically equal to zero. We conclude that these PDPs need to be interpreted very carefully because the insurance portfolio is not uniformly distributed across the feature space. In some parts of the feature space the regression function  $\mathbf{x} \mapsto \mu(\mathbf{x})$  may not even be well-defined because certain combinations of feature values  $\mathbf{x}$  may not exist (e.g., a driver of age 18 on bonus-malus level 50% or a boy at a girl's college).

### Accumulated Local Effects Profile

PDPs have the problem that they do not respect the dependencies between the feature components, as explained in the previous paragraphs. The accumulated local effects (ALE) profile tries to take account for these dependencies by only studying a local feature perturbation, we refer to Apley–Zhu [13]. We present a smooth (gradient-based) version of ALE because our regression functions are differentiable. Consider the local effect in the individual feature  $\mathbf{x}$  w.r.t. the component  $x_j$  by studying the partial derivative

$$\mu_j(\mathbf{x}) = \frac{\partial \mu(\mathbf{x})}{\partial x_j}. \quad (7.68)$$

The average local effect of component  $j$  is received by

$$x_j \mapsto \Delta_j(x_j; \mu) = \int \mu_j(x_j, \mathbf{x}_{\setminus j}) dp(\mathbf{x}_{\setminus j} | x_j). \quad (7.69)$$

ALE integrate the average local effects  $\Delta_j(\cdot)$  over their domain, and the ALE profile is defined by

$$x_j \mapsto \int_{x_{j0}}^{x_j} \Delta_j(z_j; \mu) dz_j = \int_{x_{j0}}^{x_j} \int \mu_j(z_j, \mathbf{x}_{\setminus j}) dp(\mathbf{x}_{\setminus j} | z_j) dz_j, \quad (7.70)$$

where  $x_{j_0}$  is a given initialization point. The difference between PDPs and ALE is that the latter correctly considers the dependence structure between  $x_j$  and  $\mathbf{x}_{\setminus j}$ , see (7.69).

**Listing 7.10** Local effects through the gradients of FN networks in keras [77]

---

```

1 Input = layer_input(shape = c(11), dtype = 'float32', name = 'Design')
2 #
3 Output = Input %>%
4   layer_dense(units=20, activation='tanh', name='FNLayer1') %>%
5   layer_dense(units=15, activation='tanh', name='FNLayer2') %>%
6   layer_dense(units=10, activation='tanh', name='FNLayer3') %>%
7   layer_dense(units=1, activation='linear', name='Network')
8 #
9 model = keras_model(inputs = c(Input), outputs = c(Output))
10 #
11 grad = Output %>%
12   layer_lambda(function(x) k_gradients(model$outputs, model$inputs))
13 model.grad = keras_model(inputs = c(Input), outputs = c(grad))
14 theta.grad <- data.frame(model.grad %>% predict(XX))

```

---

*Example* We come back to our MTPL claim frequency FN network example. The local effects (7.68) can directly be calculated in the R library keras [77] for a FN network, see Listing 7.10. In order to do so we need to drop the embedding layers, compared to Listing 7.4, and directly work on the learned embeddings. This gives an input layer of dimension  $q = 7 + 2 + 2 = 11$  because we have two categorical features that have been embedded into 2-dimensional Euclidean spaces  $\mathbb{R}^2$ . Then, we can formally calculate the gradient of the FN network w.r.t. its inputs which is done on lines 11–13 of Listing 7.10. Remark that we work on the canonical scale because we use the linear activation function on line 7 of the listing.

There remain the averaging (7.69) and the integration (7.70) which can be done empirically

$$x_j \mapsto \Delta_j(x_j; \mu) = \frac{1}{|\mathcal{E}(x_j)|} \sum_{i \in \mathcal{E}(x_j)} \mu_j(\mathbf{x}_i), \quad (7.71)$$

where  $\mathcal{E}(x_j)$  denotes the indices  $i$  of all cases  $\mathbf{x}_i$ ,  $1 \leq i \leq n$ , with  $x_{i,j} = x_j$ , assuming of having discrete feature data observations. Note that this empirical averaging respects the dependence within  $\mathbf{x}$ . The (uncentered) ALE profile is then obtained by aggregating these local effects, that is,

$$x_j \mapsto \tilde{\mu}_j(x_j) = \int_{x_{j_0}}^{x_j} \Delta_j(z_j; \mu) dz_j,$$

where this integration is typically understood in a discrete sense because the observed feature components  $x_{i,j}$  are discrete. Often, this uncentered ALE profile is still translated (centered) by the portfolio average.

*Remarks 7.35*

- We have only introduced ALE for continuous feature variables. For nominal categorical feature components it is not immediately clear how to reasonably integrate the average local effects  $\Delta_j(x_j; \mu)$ , and one typically directly analyzes these average local effects.
- For GLMs the ALEs are rather simple if we work on the canonical scale and under the canonical link, since

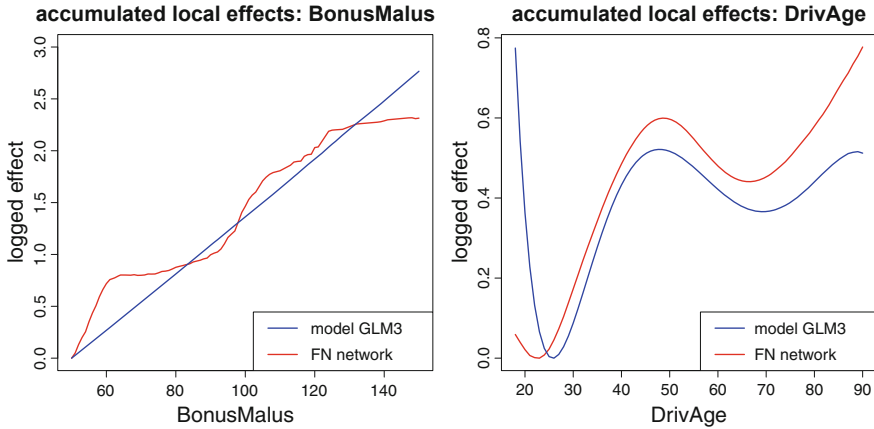
$$\theta_j(\mathbf{x}) = \frac{\partial \theta(\mathbf{x})}{\partial x_j} = \beta_j \equiv \Delta_j(x_j; \theta).$$

In the case of model Poisson GLM3 presented in Sect. 5.3.4 the situation is more delicate as we model the interactions in the GLM as follows, see (5.34) and (5.35),

$$\begin{aligned} &(\text{DrivAge}, \text{BonusMalus}) \\ &\mapsto \beta_l \text{DrivAge} + \beta_{l+1} \log(\text{DrivAge}) + \sum_{j=2}^4 \beta_{l+j} (\text{DrivAge})^j \\ &\quad + \beta_{l+5} \text{BonusMalus} + \beta_{l+6} \text{BonusMalus} \cdot \text{DrivAge} \\ &\quad + \beta_{l+7} \text{BonusMalus} \cdot (\text{DrivAge})^2. \end{aligned}$$

In that case, though we work with a GLM, the resulting local effects are different if we calculate the derivatives w.r.t. `DrivAge` and `BonusMalus`, respectively, because we explicitly (manually) include non-linear effects into the GLM.

Figure 7.43 shows the ALE profiles of the variables `BonusMalus` and `DrivAge`. The shapes of these profiles can directly be compared to the PDPs of Fig. 7.42 (the scale on the y-axis should be ignored because this will depend on the applied centering, however, we hold on to the canonical scale). The main difference between these two plots can be observed for the variable `DrivAge` at low ages. Namely, the ALE profiles have a different shape at low ages respecting the dependencies in the feature components by only considering real local feature configurations.



**Fig. 7.43** ALE profiles of (lhs) BonusMalus level and (rhs) DrivAge; the y-axis is on the log-scale

### 7.6.3 Interaction Strength

Next we are going to discuss pairwise interaction strength. Friedman–Popescu [143] made the following proposal. Roughly speaking, there is an interaction between the two feature components  $x_j$  and  $x_k$  of  $\mathbf{x}$  in the regression function  $\mathbf{x} \mapsto \mu(\mathbf{x})$  if

$$\mu_{j,k}(\mathbf{x}) = \frac{\partial^2 \mu(\mathbf{x})}{\partial x_j \partial x_k} \neq 0. \quad (7.72)$$

This means that the magnitude of a change of the regression function  $\mu(\mathbf{x})$  in  $x_j$  depends on the current value of  $x_k$ . If there is no such interaction, we can additively decompose the regression function  $\mu(\mathbf{x})$  into two independent terms. This then reads as  $\mu(\mathbf{x}) = \mu_{\setminus j}(x_{\setminus j}) + \mu_{\setminus k}(x_{\setminus k})$ . This motivation is now applied to the PDP profiles given in (7.67). We define the centered versions  $x_j \mapsto \check{\mu}^j(x_j)$  and  $x_k \mapsto \check{\mu}^k(x_k)$  of the PDP profiles by centering the PDP profiles  $x_j \mapsto \bar{\mu}^j(x_j)$  and  $x_k \mapsto \bar{\mu}^k(x_k)$  over the portfolio values  $x_i$ ,  $1 \leq i \leq n$ . Next, we consider an analogous two-dimensional version for  $(x_j, x_k)$ . Let  $(x_j, x_k) \mapsto \check{\mu}^{j,k}(x_j, x_k)$  be the centered version of a two-dimensional PDP profile  $(x_j, x_k) \mapsto \bar{\mu}^{j,k}(x_j, x_k)$ .

Friedman’s  $H$ -statistics measures the pairwise interaction strength between the components  $x_j$  and  $x_k$ , and it is defined by

$$H_{j,k}^2 = \frac{\sum_{i=1}^n (\check{\mu}^{j,k}(x_{i,j}, x_{i,k}) - \check{\mu}^j(x_{i,j}) - \check{\mu}^k(x_{i,k}))^2}{\sum_{i=1}^n \check{\mu}^{j,k}(x_{i,j}, x_{i,k})^2}, \quad (7.73)$$

we refer to formula (44) in Friedman–Popescu [143]. While  $H_{j,k}^2$  measures the proportion of the joint interaction effect, as we normalize by the variability of

the joint effect  $\sum_{i=1}^n \check{\mu}^{j,k}(x_{i,j}, x_{i,k})^2$ , sometimes also the absolute measure is considered by taking the square root of the enumerator in (7.73). Of course, this can be extended to interactions of three components, etc., we refer to Friedman–Popescu [143].

We do not give an example here, because calculating Friedman’s  $H$ -statistics can be computationally demanding if one has many feature components with many levels in FN network modeling.

### 7.6.4 Local Model-Agnostic Methods

The above methods like the PDP and the ALE profile have been analyzing the global behavior of the regression functions. We briefly mention some tools that describe the local sensitivity and explanation of regression results.

Probably the most popular method is the locally interpretable model-agnostic explanation (LIME) introduced by Ribeiro et al. [311]. This analyzes locally the expected response of a given feature  $\mathbf{x}$  by perturbing  $\mathbf{x}$ . In a nutshell, the idea is to select an environment  $\mathcal{E}(\mathbf{x}) \subset \mathcal{X}$  of a chosen feature  $\mathbf{x}$  and to study the regression function  $\mathbf{x}' \mapsto \mu(\mathbf{x}')$  in this environment  $\mathbf{x}' \in \mathcal{E}(\mathbf{x})$ . This is done by fitting a (much) simpler surrogate model to  $\mu$  on this environment  $\mathcal{E}(\mathbf{x})$ . If the environment is small, often a linear regression model is chosen. This then allows one to interpret the regression function  $\mu(\cdot)$  locally using the simpler surrogate model, and if we have a high-dimensional feature space, this linear regression is complemented with LASSO regularization to only select the most important feature components.

The second method considered in the literature is the Shapley additive explanation (SHAP). The SHAP is based on Shapley values [335] which is a method of allocating rewards to players in cooperative games, where a team of individual players jointly contributes to a potential success. Shapley values solve this allocation problem under the requirements of additivity and fairness. This concept can be translated to analyzing how individual feature components of  $\mathbf{x}$  contribute to the total prediction  $\mu(\mathbf{x})$  of a given case. Shapley values allow one to do such a contribution analysis in the aforementioned additive and fair way, see Lundberg–Lee [261]. The calculation of SHAP values is combinatorially demanding and therefore several approximations have been proposed, many of them having their own caveats, we refer to Aas et al. [1]. We will not further consider these but refer to the relevant literature.

### 7.6.5 Marginal Attribution by Conditioning on Quantiles

The above model-agnostic tools have mainly been studying the sensitivities of the expected response  $\mu(\mathbf{x})$  in the feature components of  $\mathbf{x}$ . This becomes apparent



from considering the partial derivatives (7.68) to calculate the local effects. Alternatively, we could try to understand how the feature components of  $\mathbf{x}$  contribute to a given response  $\mu(\mathbf{x})$ , see Ancona et al. [12]; this section follows Merz et al. [273]. The marginal attribution on an input component  $j$  of the response  $\mu(\mathbf{x})$  can be studied by the directional derivative

$$x_j \mapsto x_j \mu_j(\mathbf{x}) = x_j \frac{\partial \mu(\mathbf{x})}{\partial x_j}. \quad (7.74)$$

This was first proposed to the data science community by Shrikumar et al. [340]. Basically, it means that we replace the partial derivative  $\mu_j(\mathbf{x})$  by the directional derivative along the vector  $x_j \mathbf{e}_j = (0, \dots, 0, x_j, 0, \dots, 0)^\top \in \mathbb{R}^{q+1}$

$$\begin{aligned} \lim_{\epsilon \rightarrow 0} \frac{\mu(\mathbf{x} + \epsilon x_j \mathbf{e}_j) - \mu(\mathbf{x})}{\epsilon} \\ = \lim_{\epsilon \rightarrow 0} \frac{\mu((1, x_1, \dots, x_{j-1}, (1 + \epsilon)x_j, x_{j+1}, \dots, x_q)^\top) - \mu(\mathbf{x})}{\epsilon} = x_j \mu_j(\mathbf{x}), \end{aligned}$$

where  $\mathbf{e}_j$  is the  $(j + 1)$ -st basis vector in  $\mathbb{R}^{q+1}$  (index  $j = 0$  corresponds to the intercept component  $x_0 = 1$ ).

We start by recalling the sensitivity analysis of Hong [189] and Tsanakas–Millossovich [355] in the context of risk measurement. Assume the features have a portfolio distribution  $\mathbf{X} \sim p$ . This describes the random selection of an insurance policy  $\mathbf{X} = \mathbf{x}$  from the portfolio described by  $p$ . The average price over the entire portfolio is then given by

$$\mu = \mathbb{E}_p[\mu(\mathbf{X})] = \int \mu(\mathbf{x}) dp(\mathbf{x}).$$

We implicitly interpret  $\mu(\mathbf{X}) = \mathbb{E}[Y|\mathbf{X}]$  as the price of the response  $Y$ , here, though we do not need the response distribution in this section. Assume  $\mu(\mathbf{X})$  has a continuous distribution function  $F_{\mu(\mathbf{X})}$ ; and we drop the intercept component  $X_0 = x_0 = 1$  from these considerations (but we still keep it in the regression model). This implies that  $U_{\mu(\mathbf{X})} = F_{\mu(\mathbf{X})}(\mu(\mathbf{X}))$  is uniformly distributed on  $[0, 1]$ . Choosing a density  $\zeta$  on  $[0, 1]$  gives us a probability distortion  $\zeta(U_{\mu(\mathbf{X})})$  as we have the normalization

$$\mathbb{E}_p[\zeta(U_{\mu(\mathbf{X})})] = \int_0^1 \zeta(u) du = 1.$$

This allows us to define a distorted portfolio price in the sense of a Radon–Nikodým derivative, namely, we set for the distorted portfolio price

$$\varrho(\mu(\mathbf{X}); \zeta) = \mathbb{E}_p[\mu(\mathbf{X}) \zeta(U_{\mu(\mathbf{X})})].$$

This functional  $\varrho(\mu(\mathbf{X}); \zeta)$  is a so-called distortion risk measure. Our goal is to study the sensitivities of this distortion risk measure in the components of  $\mathbf{X}$ . Assume existence of the following directional derivatives for all  $1 \leq j \leq q$

$$S_j(\mu; \zeta) = \frac{\partial}{\partial \epsilon} \varrho \left( \mu \left( (1, X_1, \dots, X_{j-1}, (1 + \epsilon)X_j, X_{j+1}, \dots, X_q)^\top \right); \zeta \right) \Big|_{\epsilon=0}.$$

$S_j(\mu; \zeta)$  can be used to describe the sensitivities of the regression function  $\mathbf{X} \mapsto \mu(\mathbf{X})$  in the feature components  $X_j$ . Under different sets of assumptions, Hong [189] and Tsanakas–Millossovich [355] have proved the following identity

$$S_j(\mu; \zeta) = \mathbb{E}_p [X_j \mu_j(\mathbf{X}) \zeta(U_{\mu(\mathbf{X})})],$$

the right-hand side exactly uses the marginal attribution (7.74). There remains the freedom of the choice of the density  $\zeta$  on  $[0, 1]$ , which allows us to study the sensitivities of different distortion risk measures. For the uniform distribution  $\zeta \equiv 1$  on  $[0, 1]$  we simply have the average (best-estimate) price and its average marginal attributions

$$\varrho(\mu(\mathbf{X}); \zeta \equiv 1) = \mathbb{E}_p[\mu(\mathbf{X})] = \mu \quad \text{and} \quad S_j(\mu; \zeta \equiv 1) = \mathbb{E}_p[X_j \mu_j(\mathbf{X})].$$

If we want to consider a quantile risk measure, called value-at-risk (VaR), we choose a Dirac measure for the density  $\zeta$ . That is, choose a point measure of mass 1 in  $\alpha \in (0, 1)$ , i.e., the density  $\zeta$  is concentrated in the single point  $\alpha$ . In that case, the event  $\{F_{\mu(\mathbf{X})}(\mu(\mathbf{X})) = U_{\mu(\mathbf{X})} = \alpha\}$  receives probability one, and therefore we have the  $\alpha$ -quantile

$$\varrho(\mu(\mathbf{X}); \alpha) = F_{\mu(\mathbf{X})}^{-1}(\alpha),$$

and the corresponding sensitivities for  $1 \leq j \leq q$

$$S_j(\mu; \alpha) = \mathbb{E}_p \left[ X_j \mu_j(\mathbf{X}) \mid \mu(\mathbf{X}) = F_{\mu(\mathbf{X})}^{-1}(\alpha) \right]. \quad (7.75)$$

### Remarks 7.36

- In the introduction to this section we have assumed that  $\mu(\mathbf{X})$  has a continuous distribution function. This emphasizes that this sensitivity analysis is most suitable for continuous feature components. Categorical and discrete feature components can be embedded into a Euclidean space, e.g., using embedding layers, and then they can be treated as continuous variables.
- Sensitivities (7.75) respect the local portfolio structure as they are calculated w.r.t.  $p$ .
- In applications, we will work with the empirical portfolio distribution for  $p$  provided by  $(\mathbf{x}_i)_{1 \leq i \leq n}$ . This gives an empirical approximation to (7.75) and, in particular, it will require a choice of a bandwidth for the evaluation of the

conditional probability, conditioned on the event  $\{\mu(X) = F_{\mu(X)}^{-1}(\alpha)\}$ . This is done with a local smoother similarly to Listing 7.8.

In analogy to Merz et al. [273] we give a different interpretation to the sensitivities (7.75), which allows us to further expand this formula. We have 1st order Taylor expansion

$$\mu(\mathbf{x} + \boldsymbol{\epsilon}) = \mu(\mathbf{x}) + (\nabla_{\mathbf{x}} \mu(\mathbf{x}))^\top \boldsymbol{\epsilon} + o(\|\boldsymbol{\epsilon}\|_2) \quad \text{for } \|\boldsymbol{\epsilon}\|_2 \rightarrow 0.$$

Obviously, this is a local approximation in  $\mathbf{x}$ . Setting  $\boldsymbol{\epsilon} = -\mathbf{x}$ , we get (a possibly crude) approximation

$$\mu(\mathbf{0}) \approx \mu(\mathbf{x}) - (\nabla_{\mathbf{x}} \mu(\mathbf{x}))^\top \mathbf{x}.$$

By bringing the gradient term to the other side, using (7.75) and conditionally averaging, we receive the 1st order marginal attributions

$$F_{\mu(X)}^{-1}(\alpha) = \mathbb{E}_p \left[ \mu(X) \mid \mu(X) = F_{\mu(X)}^{-1}(\alpha) \right] \approx \mu(\mathbf{0}) + \sum_{j=1}^q S_j(\mu; \alpha). \quad (7.76)$$

Thus, the sensitivities  $S_j(\mu; \alpha)$  provide a 1st order description of the quantiles  $F_{\mu(X)}^{-1}(\alpha)$  of  $\mu(X)$ . We call this approach marginal attribution by conditioning on quantiles (MACQ) because it shows how the components  $X_j$  of  $\mathbf{X}$  contribute to a given quantile level.

*Example 7.37 (MACQ for Linear Regression)* The simplest case is the linear regression case because the 1st order marginal attributions (7.76) are exact in this case. Consider a linear regression function with regression parameter  $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = \langle \boldsymbol{\beta}, \mathbf{x} \rangle = \beta_0 + \sum_{j=1}^q \beta_j x_j.$$

The 1st order marginal attributions for fixed  $\alpha \in (0, 1)$  are given by

$$\begin{aligned} F_{\mu(X)}^{-1}(\alpha) &= \mu(\mathbf{0}) + \sum_{j=1}^q S_j(\mu; \alpha) \\ &= \beta_0 + \sum_{j=1}^q \beta_j \mathbb{E}_p \left[ X_j \mid \mu(X) = F_{\mu(X)}^{-1}(\alpha) \right]. \end{aligned} \quad (7.77)$$

That is, we replace the feature components  $X_j$  by their expected contributions on a given quantile level  $F_{\mu(X)}^{-1}(\alpha)$  in (7.77). We compare this explanation to the ALE

profile (7.70). Set initial value  $x_{j_0} = 0$ , the ALE profile for the linear regression model is given by

$$x_j \mapsto \int_0^{x_j} \Delta_j(z_j) dz_j = \beta_j x_j.$$

This is the sensitivity of the linear regression function in component  $x_j$ , whereas (7.77) describes the contribution of each feature component to an expected response level  $\mu(\mathbf{x})$ , in particular,  $\mathbb{E}_p[X_j | \mu(\mathbf{X}) = F_{\mu(\mathbf{X})}^{-1}(\alpha)]$  describes the average feature value in component  $j$  on a given quantile level. ■

A natural next step is to expand the 1st order attributions to 2nd orders. This allows us to consider the interaction terms. Consider the 2nd order Taylor expansion

$$\mu(\mathbf{x} + \boldsymbol{\epsilon}) = \mu(\mathbf{x}) + (\nabla_{\mathbf{x}} \mu(\mathbf{x}))^\top \boldsymbol{\epsilon} + \frac{1}{2} \boldsymbol{\epsilon}^\top \nabla_{\mathbf{x}}^2 \mu(\mathbf{x}) \boldsymbol{\epsilon} + o(\|\boldsymbol{\epsilon}\|_2^2) \quad \text{for } \|\boldsymbol{\epsilon}\|_2 \rightarrow 0.$$

Similar to (7.76), setting  $\boldsymbol{\epsilon} = -\mathbf{x}$ , this gives us the 2nd order marginal attributions

$$\begin{aligned} F_{\mu(\mathbf{X})}^{-1}(\alpha) &\approx \mu(\mathbf{0}) + \sum_{j=1}^q S_j(\mu; \alpha) - \frac{1}{2} \sum_{j,k=1}^q T_{j,k}(\mu; \alpha) \\ &= \mu(\mathbf{0}) + \sum_{j=1}^q \left( S_j(\mu; \alpha) - \frac{1}{2} T_{j,j}(\mu; \alpha) \right) - \sum_{1 \leq j < k \leq q} T_{j,k}(\mu; \alpha), \end{aligned} \quad (7.78)$$

where for  $1 \leq j, k \leq q$  we define  $\mu_{j,k}(\mathbf{x}) = \partial_{x_j} \partial_{x_k} \mu(\mathbf{x})$ , see (7.72), and

$$T_{j,k}(\mu; \alpha) = \mathbb{E}_p \left[ X_j X_k \mu_{j,k}(\mathbf{X}) \mid \mu(\mathbf{X}) = F_{\mu(\mathbf{X})}^{-1}(\alpha) \right]. \quad (7.79)$$

### Remarks 7.38

- The first line of (7.78) separates the 1st order attributions from the 2nd order attributions, the second line splits w.r.t. the individual component  $j$  attributions and the interaction attributions  $j \neq k$ .
- The 1st order attributions (7.75) have been motivated by considering the directional derivatives of the VaR distortion risk measure. Unfortunately, the 2nd order consideration has no simple equivalent motivation, as the 2nd order directional derivatives are much more involved, even in the linear case, we refer to Property 1 in Gouriéroux et al. [167].

- Interestingly, we can precisely evaluate the accuracy of approximation (7.78) by analyzing for a given regression function  $\mu(\cdot)$

$$\sup_{\alpha \in (0,1)} \left| F_{\mu(X)}^{-1}(\alpha) - \mu(\mathbf{0}) - \sum_{j=1}^q S_j(\mu; \alpha) + \frac{1}{2} \sum_{j,k=1}^q T_{j,k}(\mu; \alpha) \right|. \quad (7.80)$$

Intuitively, in order to have a uniform good approximation, the origin  $\mathbf{0}$  should be somehow centered in the feature distribution  $X \sim p$ . This will be studied next.

Above we have implicitly assumed that  $\mathbf{0}$  is a suitable reference point that makes the approximation error (7.80) small. For FN network fitting we typically normalize the features either using the MinMaxScaler (7.29) or we center and normalize the components of  $(x_i)_{1 \leq i \leq n}$  according to (7.30). That is, the reference point is chosen such that the gradient descent fitting works efficiently. However, this may not be an optimal reference point for studying the 2nd order attributions. Therefore, we analyze this question in more detail, and the following reparametrization can still be done after model fitting.

If we choose an arbitrary translation  $\mathbf{a} \in \mathbb{R}^q$ , we can set  $\boldsymbol{\epsilon} = \mathbf{a} - \mathbf{x}$  in the above 2nd order Taylor expansion to receive another 2nd order marginal attribution representation

$$\begin{aligned} F_{\mu(X)}^{-1}(\alpha) &\approx \mu(\mathbf{a}) - \mathbb{E}_p \left[ (\mathbf{a} - \mathbf{X})^\top \nabla_{\mathbf{x}} \mu(X) \mid \mu(X) = F_{\mu(X)}^{-1}(\alpha) \right] \\ &\quad - \frac{1}{2} \mathbb{E}_p \left[ (\mathbf{a} - \mathbf{X})^\top \nabla_{\mathbf{x}}^2 \mu(X) (\mathbf{a} - \mathbf{X}) \mid \mu(X) = F_{\mu(X)}^{-1}(\alpha) \right]. \end{aligned} \quad (7.81)$$

Essentially, this means that we shift the feature distribution  $p$  to considering the shifted random vectors  $\mathbf{X}^{\mathbf{a}} = \mathbf{X} - \mathbf{a}$  and while setting  $\mu^{\mathbf{a}}(\cdot) = \mu(\mathbf{a} + \cdot)$ , thus, this simply says that we pre-process the features differently. In view of approximation (7.81) we can now select a reference point  $\mathbf{a} \in \mathbb{R}^q$  that makes the 2nd order marginal attributions as precise as possible. Define the events  $\mathcal{A}_l = \{\mu(X) = F_{\mu(X)}^{-1}(\alpha_l)\}$  for a discrete quantile grid  $0 < \alpha_1 < \dots < \alpha_L < 1$ . We define the objective function

$$\begin{aligned} \mathbf{a} \mapsto G(\mathbf{a}; \mu) &= \sum_{l=1}^L \left( F_{\mu(X)}^{-1}(\alpha_l) - \mu(\mathbf{a}) + \mathbb{E}_p \left[ (\mathbf{a} - \mathbf{X})^\top \nabla_{\mathbf{x}} \mu(X) \mid \mathcal{A}_l \right] \right. \\ &\quad \left. + \frac{1}{2} \mathbb{E}_p \left[ (\mathbf{a} - \mathbf{X})^\top \nabla_{\mathbf{x}}^2 \mu(X) (\mathbf{a} - \mathbf{X}) \mid \mathcal{A}_l \right] \right)^2. \end{aligned} \quad (7.82)$$

Making this objective function  $G(\mathbf{a}; \mu)$  small in  $\mathbf{a}$  will provide us with a good reference point for the selected quantile levels  $(\alpha_l)_{1 \leq l \leq L}$ ; this is exactly the MACQ

proposal of Merz et al. [273]. A local minimum can be found by applying a gradient descent algorithm

$$\mathbf{a}^{(t)} \mapsto \mathbf{a}^{(t+1)} = \mathbf{a}^{(t)} - \delta_{t+1} \nabla_{\mathbf{a}} G(\mathbf{a}^{(t)}; \mu),$$

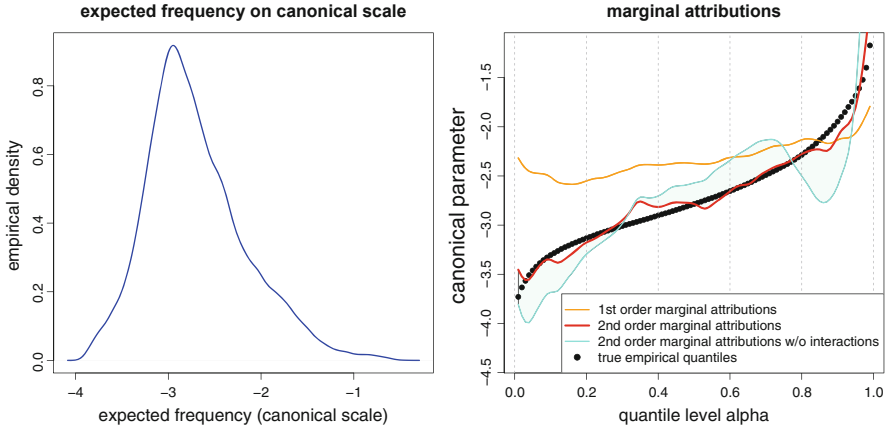
for tempered learning rates  $\delta_{t+1} > 0$ . The gradient of  $G$  w.r.t.  $\mathbf{a}$  is given by

$$\begin{aligned} \nabla_{\mathbf{a}} G(\mathbf{a}; \mu) = & 2 \sum_{l=1}^L \left( F_{\mu(X)}^{-1}(\alpha_l) - \mu(\mathbf{a}) + \mathbb{E}_p \left[ (\mathbf{a} - \mathbf{X})^\top \nabla_{\mathbf{x}} \mu(\mathbf{X}) \middle| \mathcal{A}_l \right] \right. \\ & \left. + \frac{1}{2} \mathbb{E}_p \left[ (\mathbf{a} - \mathbf{X})^\top \nabla_{\mathbf{x}}^2 \mu(\mathbf{X}) (\mathbf{a} - \mathbf{X})^\top \middle| \mathcal{A}_l \right] \right) \\ & \times \left( -\nabla_{\mathbf{a}} \mu(\mathbf{a}) + \mathbb{E}_p \left[ \nabla_{\mathbf{x}} \mu(\mathbf{X}) \middle| \mathcal{A}_l \right] \right. \\ & \left. - \mathbb{E}_p \left[ \mathbf{X}^\top \nabla_{\mathbf{x}}^2 \mu(\mathbf{X}) \middle| \mathcal{A}_l \right] + \frac{1}{2} \mathbf{a}^\top \mathbb{E}_p \left[ \nabla_{\mathbf{x}}^2 \mu(\mathbf{X}) \middle| \mathcal{A}_l \right] \right). \end{aligned}$$

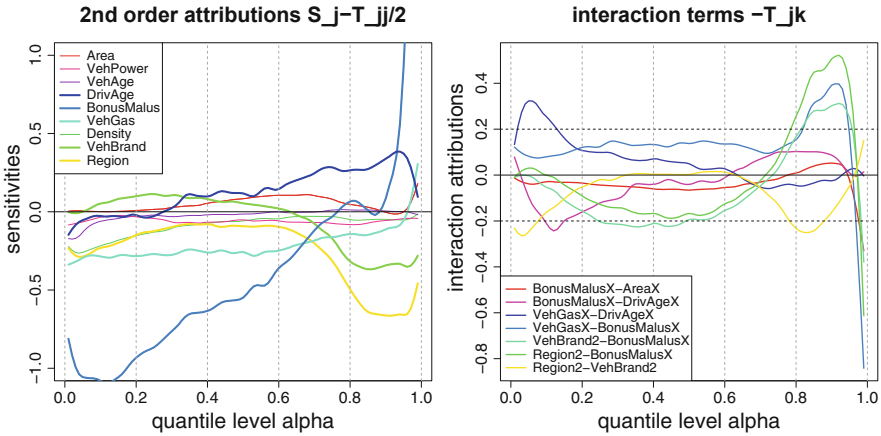
All subsequent considerations and interpretations are done w.r.t. an optimal reference point  $\mathbf{a} \in \mathbb{R}^q$  by minimizing the objective function (7.82) on the chosen quantile grid. Mathematically speaking, this optimal choice is w.l.o.g. because the origin  $\mathbf{0}$  of the coordinate system of the feature space  $\mathcal{X}$  is arbitrary, and any other origin can be chosen by a translation, see formula (7.81) and the subsequent discussion. For interpretations, however, the choice of the reference point  $\mathbf{a}$  matters because the directional derivative  $X_j \mu_j(\mathbf{X})$  can be small either because  $X_j$  is small or because  $\mu_j(\mathbf{X})$  is small. Having a small  $X_j$  means that this feature value is close to the chosen reference point.

*Example 7.39 (MACQ Analysis)* We revisit the MTPL claim frequency example using the FN network regression model of depth  $d = 3$  having  $(q_1, q_2, q_3) = (20, 15, 10)$  neurons. Importantly, we use the hyperbolic tangent as the activation function in the FN layers which provides smoothness of the regression function. Figure 7.40 shows the VPI plot of this fitted model. Obviously, the variable `BonusMalus` plays the most important role in this predictive model. Remark that the VPI plot does not properly respect the dependence structure in the features as it independently permutes each feature component at a time. The aim in this example is to determine variable importance by doing the MACQ analysis (7.78).

Figure 7.44 (lhs) shows the empirical density of the fitted canonical parameter  $\theta(\mathbf{x}_i)$ ,  $1 \leq i \leq n$ ; all plots in this example refer to the canonical scale. We then minimize the objective function (7.82) which provides us with an optimal reference point  $\mathbf{a} \in \mathbb{R}^q$ ; we choose equidistant quantile grid  $1\% < 2\% < \dots < 99\%$  and all conditional expectations in  $\nabla_{\mathbf{a}} G(\mathbf{a}; \mu)$  are empirically approximated by a local smoother similar to Listing 7.8. Figure 7.44 (rhs) gives the resulting marginal attributions w.r.t. this reference point. The orange line shows the 1st order marginal



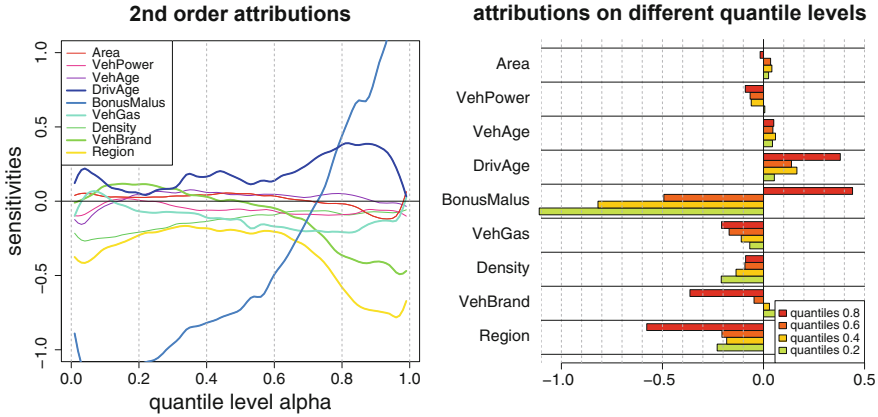
**Fig. 7.44** (lhs) Empirical density of the fitted canonical parameter  $\theta(x_i)$ ,  $1 \leq i \leq n$ , (rhs) 1st and 2nd order marginal attributions



**Fig. 7.45** (lhs) Second order marginal attributions  $S_j(\mu; \alpha) - \frac{1}{2}T_{j,j}(\mu; \alpha)$  excluding interaction terms, and (rhs) interaction terms  $-\frac{1}{2}T_{j,k}(\mu; \alpha)$ ,  $j \neq k$

attributions (7.76), and the red line the 2nd order marginal attributions (7.78). The cyan line drops the interaction terms  $T_{j,k}(\mu; \alpha)$ ,  $j \neq k$ , from the 2nd order marginal attributions. From the shaded cyan area we see the importance of the interaction terms. We note that the 2nd order marginal attributions (red line) match the true empirical quantiles (black dots) quite well for the chosen reference point  $\mathbf{a}$ .

Figure 7.45 gives the 2nd order marginal attributions  $S_j(\mu; \alpha) - \frac{1}{2}T_{j,j}(\mu; \alpha)$  of the individual components  $1 \leq j \leq q$  on the left-hand side, and the interaction terms  $-\frac{1}{2}T_{j,k}(\mu; \alpha)$ ,  $j \neq k$  on the right-hand side. We identify the following components as being important BonusMalus, DrivAge, VehGas, VehBrand and Region; these components show a behavior substantially different from being equal to 0, i.e.,



**Fig. 7.46** (lhs) Second order marginal attributions  $S_j(\mu; \alpha) - \frac{1}{2} \sum_{k=1}^q T_{j,k}(\mu; \alpha)$  including interaction terms, and (rhs) slices at the quantile levels  $\alpha \in \{20\%, 40\%, 60\%, 80\%\}$

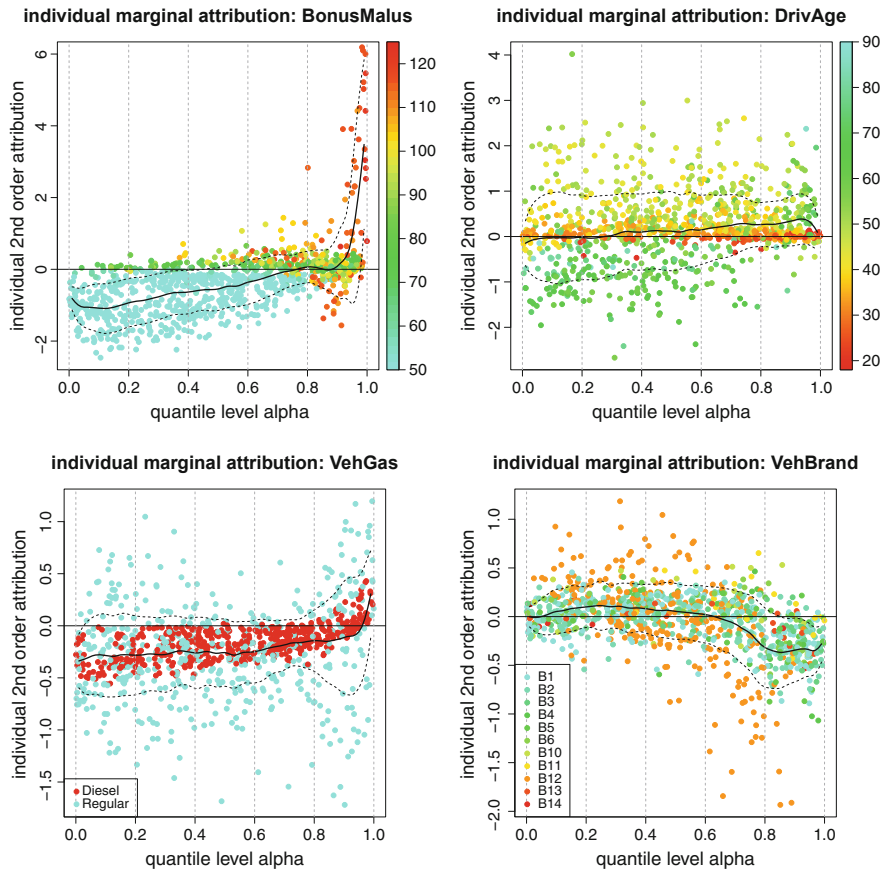
these components differentiate from the reference point  $\mathbf{a}$ . These components also have major interactions that contribute to the quantiles above the level 80%.

If we allocate the interaction terms to the corresponding components  $1 \leq j \leq q$  we receive the second order marginal attributions  $S_j(\mu; \alpha) - \frac{1}{2} \sum_{k=1}^q T_{j,k}(\mu; \alpha)$ . These are illustrated in Fig. 7.46 (lhs) and the quantile slices at the levels  $\alpha \in \{20\%, 40\%, 60\%, 80\%\}$  are given in Fig. 7.46 (rhs). These graphs illustrate variable importance on different quantile levels (and respecting the dependence within the features). In particular, we identify the main variables that distinguish the given quantile levels from the reference level  $\theta(\mathbf{a})$ , i.e., Fig. 7.46 (rhs) should be understood as the relative differences to the chosen reference level. Once more we see that BonusMalus is the main driver, but also other variables contribute to the differentiation of the high quantile levels.

Figure 7.47 shows the individual attributions  $x_{i,j} \mu_j(\mathbf{x}_i)$  of 1'000 randomly selected cases  $\mathbf{x}_i$  for the feature components  $j = \text{BonusMalus}, \text{DrivAge}, \text{VehGas}, \text{VehBrand}$ ; the colors illustrate the corresponding feature values  $x_{i,j}$  of the individual car drivers  $i$ , and the black solid line corresponds to  $S_j(\mu; \alpha) - \frac{1}{2} T_{j,j}(\mu; \alpha)$  excluding the interaction terms (the black dotted line is one empirical standard deviation around the black solid line). Focusing on the variable BonusMalus we observe that the lower quantiles are almost completely dominated by insurance policies on the lowest bonus-malus level. The bonus-malus levels 70–80 provide little sensitivity (are concentrated around the zero line) because the reference point  $\mathbf{a}$  reflects these bonus-malus levels, and, finally, the large quantiles are dominated by high bonus-malus levels (red dots).

The plot of the variable DrivAge is interpreted similarly. The reference point  $\mathbf{a}$  is close to the young drivers, therefore, young drivers are concentrated around the zero line. At the low quantile levels, higher ages contribute positively to the low expected frequencies, whereas these ages have an unfavorable impact at higher





**Fig. 7.47** Individual attributions  $x_{i,j}\mu_j(x_i)$  of 1'000 randomly selected cases  $x_i$  for  $j =$  BonusMalus, DrivAge, VehGas, VehBrand; the plots have different y-scales

quantile levels (this should be considered in combination with their bonus-malus levels). We also observe a few outliers in this plot, for instance, we can identify a driver of age 20 at a quantile level of 20%. Further inspection of this driver raises some doubts whether this data is correct since this driver is at a bonus-malus level of 68% (which should technically not be possible) and she/he has an exposure of 2 days. Surely, this insurance policy would need further investigation.

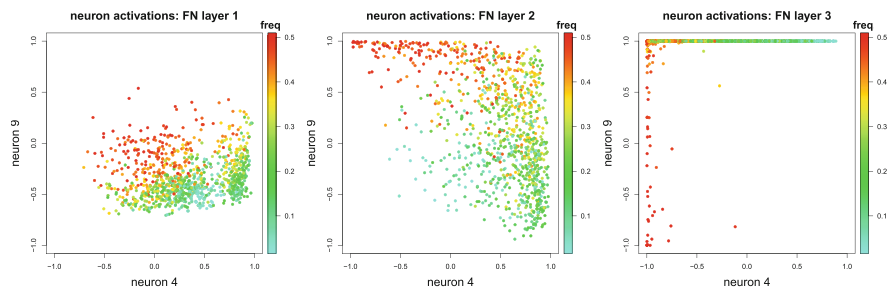
The plot of VehGas shows that the chosen reference level  $\theta(a)$  is closer to Diesel fuel cars as the red dots fluctuate less around the zero line; in different runs of the gradient descent algorithm (with different seeds) this order has been changing (as it depends on the reference point  $a$ ). We skip a detailed analysis of the variable VehBrand. ■

## 7.7 Lab: Analysis of the Fitted Networks

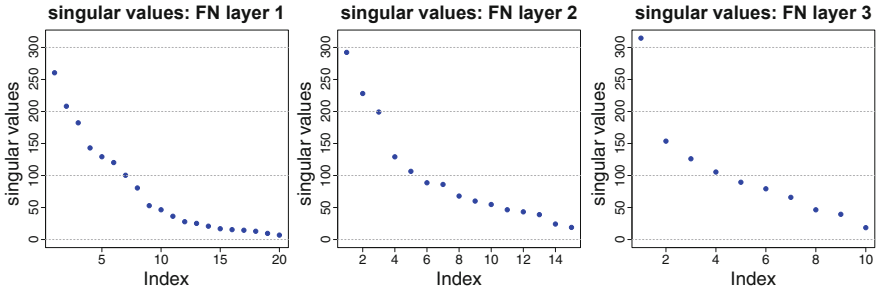
In the previous section we have studied some model-agnostic tools that can be used for any (differentiable) regression model. In this section we give some network specific plots. For simplicity we choose one specific example, namely, the FN network  $\hat{\mu} \stackrel{\text{def.}}{=} \hat{\mu}^{m=1}$  of Table 7.9. We start by analyzing the learned representations in the different FN layers, this links to our introduction in Sect. 7.1.

For any FN layer  $1 \leq m \leq d$  we can study the learned representations  $\mathbf{z}^{(m;1)}(\mathbf{x})$ . For Fig. 7.48 we select at random 1'000 insurance policies  $\mathbf{x}_i$ , and the dots show the activations of these insurance policies in neurons  $j = 4$  ( $x$ -axis) and  $j = 9$  ( $y$ -axis) in the corresponding FN layers. These neuron activations are in the interval  $(-1, 1)$  because we work with the hyperbolic tangent activation function for  $\phi$ . The color scale shows the resulting estimated frequencies  $\hat{\mu}(\mathbf{x}_i)$  of the selected policies. We observe that the layers are increasingly (in the depth of the network) separating the low frequency policies (light blue-green colors) from the high frequency policies (red color). This is a quite typical picture that we obtain here, though, this sparsity in the 3rd FN layer is not the case for every neuron  $1 \leq j \leq q_d$ .

In higher dimensional FN architectures it will be difficult to analyze the learned representations on each individual neuron, but at least one can try to understand the main effects learned. For this, on the one hand, we can focus on the important feature components, see, e.g., Sect. 7.6.1, and, on the other hand, we can try to study the main effects learned using a PCA in each FN layer, see Sect. 7.5.3. Figure 7.49 shows the singular values  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{q_m} > 0$  in each of the three FN layers  $1 \leq m \leq d = 3$ ; we center the neuron activations to mean zero before applying the SVD. These plots support the previously made statement that the layers are increasingly separating the high frequency from the low frequency policies. An elbow criterion tells us that in the first FN layer we have 8 important principal components (out of 20), in the second FN layer 3 (out of 15) and in the third FN layer 1 (out of 10). This is also reflected in Fig. 7.48 where we see more and more



**Fig. 7.48** Observed activations in the three FN layers  $m = 1, 2, 3$  (left-middle-right) in the corresponding neurons  $j = 4, 9$ , the color key shows the estimated frequencies  $\hat{\mu}(\mathbf{x}_i)$

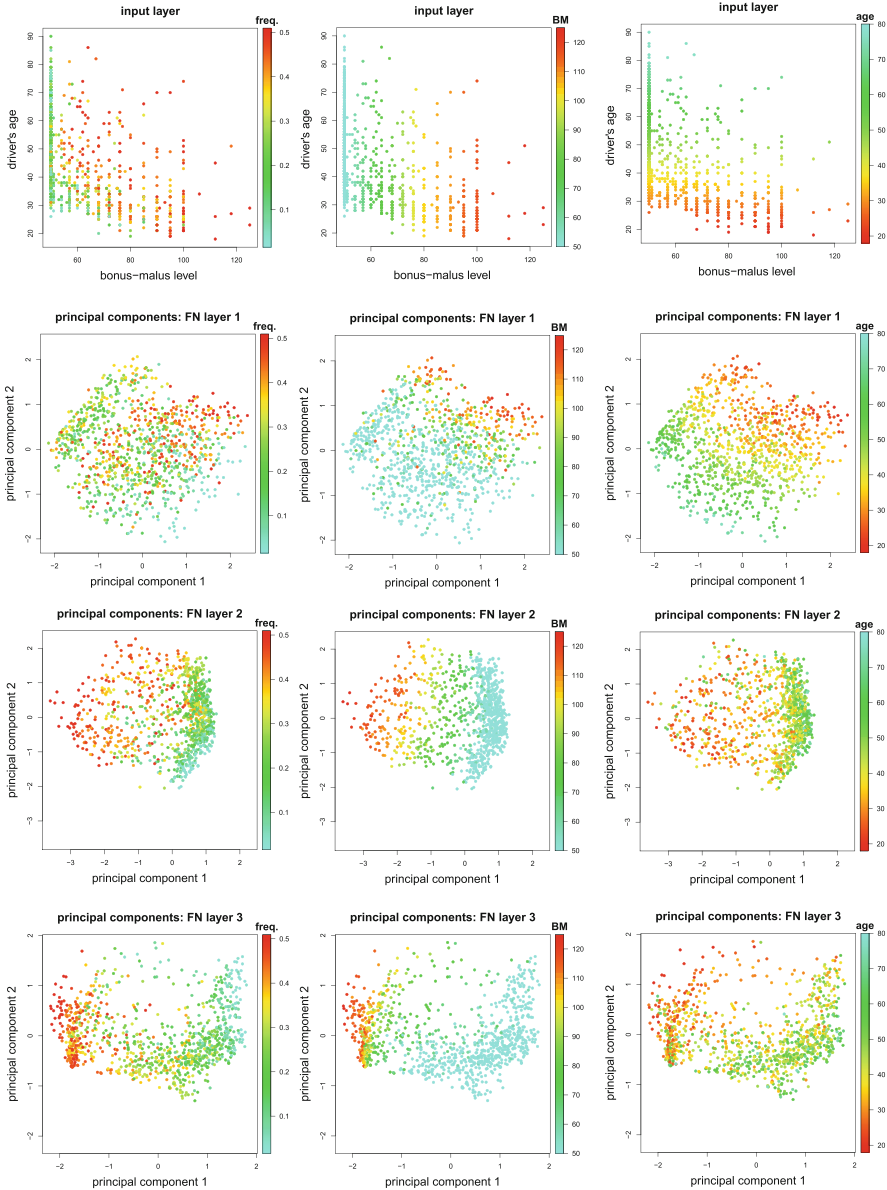


**Fig. 7.49** Singular values  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{q_m} > 0$  in the FN layers  $1 \leq m \leq d = 3$

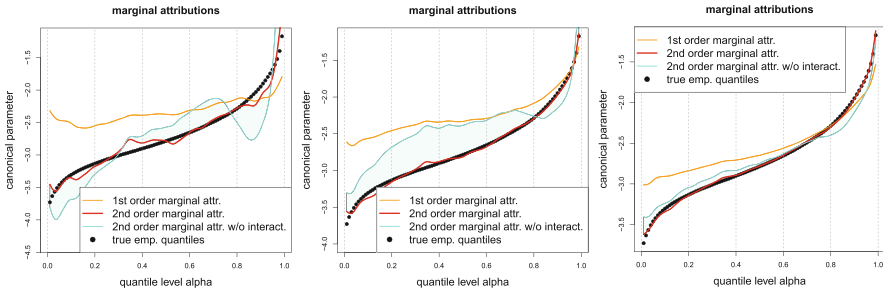
concentration in the neuron activations. It is important to notice that the chosen FN network calibration  $\hat{\mu}$  does not involve any drop-out layers during the gradient descent fitting, see Sect. 7.4.1. Drop-out layers prevent individual neurons to over-train to a specific task. Consequently, we will receive a network calibration that is more equally balanced across all neurons under drop-outs, because if one neuron drops out, the composite of the remaining neurons needs to be able to take over the task of the dropped out neuron. This leads to less sparsity and to singular values that are more similarly sized.

In Fig. 7.50 we analyze the first two principal components in each FN layer, thus, these are the two principal components that correspond to the two biggest singular values ( $\lambda_1, \lambda_2$ ) in each of the three FN layers. The first row shows the input variables ( $\text{BonusMalus}, \text{DrivAge}$ )  $\in [50, 125] \times [18, 90]$  of the 1'000 randomly selected policies  $\mathbf{x}_i$ ; these are the two most important feature components according to the VPI analysis. All three columns show the same data, however, in different color scales: (lhs) gives the color scale  $\hat{\mu}$ , (middle) gives the color scale  $\text{BonusMalus}$ , and (rhs) gives the color scale  $\text{DrivAge}$ . These color scales also apply to the other rows. The 2nd row shows the first two principal components in the 1st FN layer, the 3rd row in the 2nd FN layer, and the last row in the third FN layer. Focusing on the first column we observe that the layers cluster the high and the low frequency policies in the 1st principal component more and more across the FN layers. Not surprisingly this leads to a quite clear-cut separation w.r.t. the bonus-malus level which can be verified from the second column of Fig. 7.50. For the driver's age variable this sharp separation gets lost across the layers, see third column of Fig. 7.50, which indicates that the variable  $\text{DrivAge}$  does not influence the frequency monotonically and it interacts with the variable  $\text{BonusMalus}$ .

Figure 7.51 shows the second order marginal attributions (7.78) for the different inputs. The graph on the left-hand side shows the plot w.r.t. the original inputs  $\mathbf{x}_i$ , the graph in the middle w.r.t. the learned representations  $\mathbf{z}^{(1:1)}(\mathbf{x}_i) \in \mathbb{R}^{q_1}$  in the first FN layer, and on the right-hand side w.r.t. the learned representations  $\mathbf{z}^{(2:1)}(\mathbf{x}_i) \in \mathbb{R}^{q_2}$  in the second FN layer. We interpret these plots as follows: the FN network disentangles the different effects through the FN layers by making



**Fig. 7.50** (First row) Input variables (BonusMalus, DrivAge), (Second–fourth row) first two principal components in FN layers  $m = 1, 2, 3$ ; (lhs) gives the color scale of estimated frequency  $\hat{\mu}$ , (middle) gives the color scale BonusMalus, and (rhs) gives the color scale DrivAge



**Fig. 7.51** Second order marginal attributions: (lhs) w.r.t. the input layer  $\mathbf{x} \in \mathbb{R}^{q_0}$ , (middle) w.r.t. the first FN layer  $\mathbf{z}^{(1:1)}(\mathbf{x}) \in \mathbb{R}^{q_1}$ , and (rhs) w.r.t. the second FN layer  $\mathbf{z}^{(2:1)}(\mathbf{x}) \in \mathbb{R}^{q_2}$

the plots more smooth and making the interactions between the neurons smaller. Note that the learned representations  $\mathbf{z}^{(3:1)}(\mathbf{x}_i) \in \mathbb{R}^{q_3}$  in the last FN layer go into a classical GLM for the output layer, which does not have any interactions in the canonical predictor (because it is additive on the canonical scale), thus, being of the same type as the linear regression of Example 7.37. In the Poisson model with the log-link function, the interactions can only be of a multiplicative type in GLMs. Therefore, the network feature-engineers the input  $\mathbf{x}_i$  (in an automated way) such that the learned representation  $\mathbf{z}^{(d:1)}(\mathbf{x}_i)$  in the last FN layer is exactly in this GLM structure. This is verified by the small interaction part in Fig. 7.51 (rhs). This closes this part on model-agnostic tools.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.



## Chapter 8

# Recurrent Neural Networks



Chapter 7 has discussed fully-connected *feed-forward neural* (FN) networks. Feed-forward means that information is passed in a directed acyclic path from the input layer to the output layer. A natural extension is to allow these networks to have cycles. In that case, we call the architecture a *recurrent neural* (RN) network. A RN network architecture is particularly useful for time-series modeling. The discussion on time-series data also links to Sect. 5.8.1 on longitudinal and panel data. RN networks have been introduced in the 1980s, and the two most popular RN network architectures are the long short-term memory (LSTM) architecture proposed by Hochreiter–Schmidhuber [188] and the gated recurrent unit (GRU) architecture introduced by Cho et al. [76]. These two architectures will be described in detail in this chapter.

### 8.1 Motivation for Recurrent Neural Networks

We start from a deep FN network providing the regression function, see (7.2)–(7.3),

$$\mathbf{x} \mapsto \mu(\mathbf{x}) = g^{-1} \langle \boldsymbol{\beta}, \mathbf{z}^{(d:1)}(\mathbf{x}) \rangle, \quad (8.1)$$

with a composition  $\mathbf{z}^{(d:1)}$  of  $d$  FN layers  $\mathbf{z}^{(m)}$ ,  $1 \leq m \leq d$ , link function  $g$  and with output parameter  $\boldsymbol{\beta} \in \mathbb{R}^{qd+1}$ . In principle, we could directly use this FN network architecture for time-series forecasting. We explain here why this is not the best option to deal with time-series data.

Assume we want to predict a random variable  $Y_{T+1}$  at time  $T \geq 0$  based on the time-series information  $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T$ . This information is assumed to be available at time  $T$  for predicting the response  $Y_{T+1}$ . The past response information  $Y_t$ ,  $1 \leq$

$t \leq T$ , is typically included in  $\mathbf{x}_t$ .<sup>1</sup> Using the above FN network architecture we could directly try to predict  $Y_{T+1}$ , based on this past information. Therefore, we define the feature information  $\mathbf{x}_{0:T} = (\mathbf{x}_0, \dots, \mathbf{x}_T)$  and we aim at designing a FN network (8.1) for modeling

$$\mathbf{x}_{0:T} \mapsto \mu_T(\mathbf{x}_{0:T}) = \mathbb{E}[Y_{T+1}|\mathbf{x}_{0:T}] = \mathbb{E}[Y_{T+1}|\mathbf{x}_0, \dots, \mathbf{x}_T].$$

In principle we could work with such an approach, however, it has a couple of severe drawbacks. Obviously, the length of the feature vector  $\mathbf{x}_{0:T}$  depends on time  $T$ , that is, it will grow with every time step. Therefore, the regression function (network architecture)  $\mathbf{x}_{0:T} \mapsto \mu_T(\mathbf{x}_{0:T})$  is time-dependent. Consequently, with this approach we have to fit a network for every  $T$ . This deficiency can be circumvented if we assume a Markov property that does not require of carrying forward the whole past history. Assume that it is sufficient to consider a history of a certain length. Choose  $\tau \geq 0$  fixed, then, for  $T \geq \tau$ , we can set for the feature information  $\mathbf{x}_{T-\tau:T} = (\mathbf{x}_{T-\tau}, \dots, \mathbf{x}_T)$ , which has a fixed length  $\tau + 1 \geq 1$ , now. In this situation we could try to design a FN network

$$\mathbf{x}_{T-\tau:T} \mapsto \mu(\mathbf{x}_{T-\tau:T}) = \mathbb{E}[Y_{T+1}|\mathbf{x}_{T-\tau:T}] = \mathbb{E}[Y_{T+1}|\mathbf{x}_{T-\tau}, \dots, \mathbf{x}_T].$$

This network regression function can be chosen independent of  $T$  since the relevant history  $\mathbf{x}_{T-\tau:T}$  always has the same length  $\tau + 1$ . The time variable  $T$  could be used as a feature component in  $\mathbf{x}_{T-\tau:T}$ . The disadvantage of this approach is that such a FN network architecture does not respect the temporal causality. Observe that we feed the past history into the first FN layer

$$\mathbf{x}_{T-\tau:T} \mapsto \mathbf{z}^{(1)}(\mathbf{x}_{T-\tau:T}) \in \{1\} \times \mathbb{R}^{q_1}.$$

This operation typically does not respect any topology in the time index of  $\mathbf{x}_{T-\tau+1:T}$ . Thus, the FN network does not recognize that the feature  $\mathbf{x}_{t-1}$  has been experienced just before the next feature  $\mathbf{x}_t$ . For this reason we are looking for a network architecture that can handle the time-series information in a temporal causal way.

---

<sup>1</sup> More mathematically speaking, we assume to have a filtration  $(\mathcal{A}_t)_{t \geq 0}$  on the probability space  $(\Omega, \mathcal{A}, \mathbb{P})$ . The basic assumption then is that both sequences  $(\mathbf{x}_t)_t$  and  $(Y_t)_t$  are  $(\mathcal{A}_t)_t$ -adapted, and we aim at predicting  $Y_{T+1}$ , based on the information  $\mathcal{A}_T$ . In the above case this information  $\mathcal{A}_T$  is generated by  $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T$ , where  $\mathbf{x}_t$  typically includes the observation  $Y_t$ . We could also shift the time index in  $\mathbf{x}_t$  by one time unit, and in that case we would assume that  $(\mathbf{x}_t)_t$  is previsible w.r.t. the filtration  $(\mathcal{A}_t)_t$ . We do not consider this shift in time index as it only makes the notation unnecessarily more complicated, but the results remain the same by including the information correspondingly into the features.

## 8.2 Plain-Vanilla Recurrent Neural Network

### 8.2.1 Recurrent Neural Network Layer

We explain the basic idea of RN networks in a shallow network architecture, and deep network architectures will be discussed in Sect. 8.2.2, below. We start from the time-series input variable  $\mathbf{x}_{0:T} = (\mathbf{x}_0, \dots, \mathbf{x}_T)$ , all components having the same structure  $\mathbf{x}_t \in \mathcal{X} \subset \{1\} \times \mathbb{R}^{q_0}$ ,  $0 \leq t \leq T$ . The aim is to design a network architecture that allows us to predict the random variable  $Y_{T+1}$ , based on this time-series information  $\mathbf{x}_{0:T}$ .

The main idea is to feed one component  $\mathbf{x}_t$  of the time-series  $\mathbf{x}_{0:T}$  at a time into the network, and at the same time we use the output  $\mathbf{z}_{t-1}$  of the previous loop as an input for the next loop. This variable  $\mathbf{z}_{t-1}$  carries forward a memory of the past variables  $\mathbf{x}_{0:t-1}$ . We explain this with a single RN layer having  $q_1 \in \mathbb{N}$  neurons. A RN layer is given (recursively) by a mapping,  $t \geq 1$ ,

$$\begin{aligned} \mathbf{z}^{(1)} : \{1\} \times \mathbb{R}^{q_0} \times \mathbb{R}^{q_1} &\rightarrow \mathbb{R}^{q_1}, \\ (\mathbf{x}_t, \mathbf{z}_{t-1}) &\mapsto \mathbf{z}_t = \mathbf{z}^{(1)}(\mathbf{x}_t, \mathbf{z}_{t-1}), \end{aligned} \quad (8.2)$$

where the RN layer  $\mathbf{z}^{(1)}$  has the same structure as the FN layer given in (7.5), but based on feature input  $(\mathbf{x}_t, \mathbf{z}_{t-1}) \in \mathcal{X} \times \mathbb{R}^{q_1} \subset \{1\} \times \mathbb{R}^{q_0} \times \mathbb{R}^{q_1}$ , and not including an intercept component  $\{1\}$  in the output.

More formally, a *RN layer* with activation function  $\phi$  is a mapping

$$\begin{aligned} \mathbf{z}^{(1)} : \{1\} \times \mathbb{R}^{q_0} \times \mathbb{R}^{q_1} &\rightarrow \mathbb{R}^{q_1} \\ (\mathbf{x}, \mathbf{z}) &\mapsto \mathbf{z}^{(1)}(\mathbf{x}, \mathbf{z}) = \left( z_1^{(1)}(\mathbf{x}, \mathbf{z}), \dots, z_{q_1}^{(1)}(\mathbf{x}, \mathbf{z}) \right)^\top, \end{aligned} \quad (8.3)$$

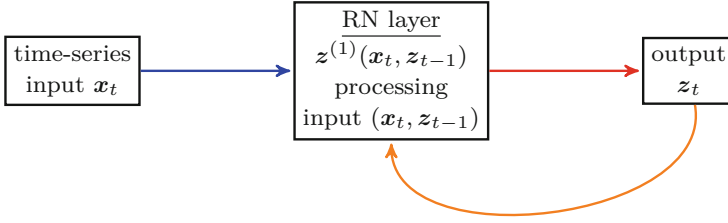
having neurons,  $1 \leq j \leq q_1$ ,

$$z_j^{(1)}(\mathbf{x}, \mathbf{z}) = \phi \left( \left\langle \mathbf{w}_j^{(1)}, \mathbf{x} \right\rangle + \left\langle \mathbf{u}_j^{(1)}, \mathbf{z} \right\rangle \right), \quad (8.4)$$

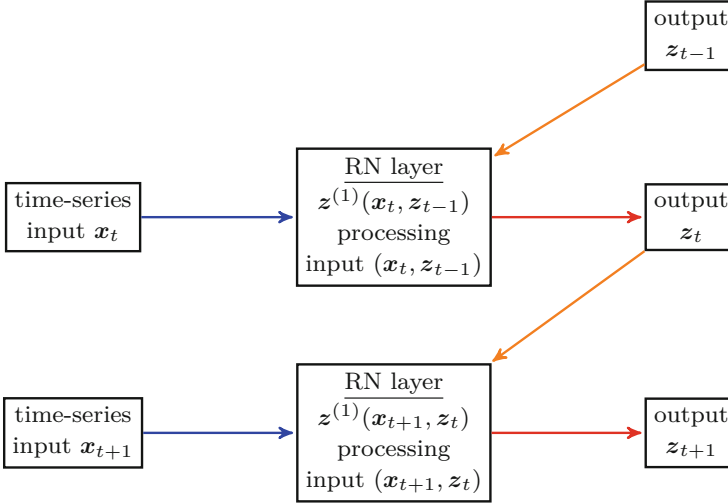
for given network weights  $\mathbf{w}_j^{(1)} \in \mathbb{R}^{q_0+1}$  and  $\mathbf{u}_j^{(1)} \in \mathbb{R}^{q_1}$ .

Thus, the FN layers (7.5)–(7.6) and the RN layers (8.3)–(8.4) are structurally equivalent, only the input  $\mathbf{x} \in \mathcal{X}$  is adapted to the time-series structure  $(\mathbf{x}_t, \mathbf{z}_{t-1}) \in \mathcal{X} \times \mathbb{R}^{q_1}$ . Before giving more interpretation and before explaining how this single RN network structure can be extended to a deep RN network we illustrate this RN layer.





**Fig. 8.1** RN layer  $z^{(1)}$  processing the input  $(x_t, z_{t-1})$



**Fig. 8.2** Unfolded representation of RN layer  $z^{(1)}$  processing the input  $(x_t, z_{t-1})$

Figure 8.1 shows an RN layer  $z^{(1)}$  processing the input  $(x_t, z_{t-1})$ , see (8.2). From this graph, the recurrent structure becomes clear since we have a loop (cycle) feeding the output  $z_t$  back into the RN layer to process the next input  $(x_{t+1}, z_t)$ .

Often one depicts the RN architecture in a so-called unfolded way. This is done in Fig. 8.2. Instead of plotting the loop (cycle) as in Fig. 8.1 (orange arrow in the colored version), we unfold this loop by plotting the RN layer multiple times. Note that this RN layer in Fig. 8.2 uses always the same network weights  $w_j^{(1)}$  and  $u_j^{(1)}$ ,  $1 \leq j \leq q_1$ , for all  $t$ . Moreover, the use of the colors of the arrows (in the colored version) in the two figures coincides.

#### Remarks 8.1

- The neurons of the RN layer (8.4) have the following structure

$$z_j^{(1)}(x, z) = \phi \left( \langle w_j^{(1)}, x \rangle + \langle u_j^{(1)}, z \rangle \right) = \phi \left( w_{0,j}^{(1)} + \sum_{l=1}^{q_0} w_{l,j}^{(1)} x_l + \sum_{l=1}^{q_1} u_{l,j}^{(1)} z_l \right).$$

The network weights  $W^{(1)} = (\mathbf{w}_j^{(1)})_{1 \leq j \leq q_1} \in \mathbb{R}^{(q_0+1) \times q_1}$  include an intercept component  $w_{0,j}^{(1)}$  and the network weights  $U^{(1)} = (\mathbf{u}_j^{(1)})_{1 \leq j \leq q_1} \in \mathbb{R}^{q_1 \times q_1}$  do not include an intercept component, otherwise we would have a redundancy.

- The RN network architecture generates a new process  $(\mathbf{z}_t)_t$ . This process encodes the part of the past history  $(\mathbf{x}_{0:t})_t$  which is relevant for forecasting the next step. Thus,  $(\mathbf{z}_t)_t$  can be interpreted as a (latent) *memory process*, or as the process of learned (relevant) time-series representation giving us  $\mathbf{z}_t = \mathbf{z}_t(\mathbf{x}_{0:t})$ .
- The same activation function  $\phi$  and the same network weights  $(\mathbf{w}_j^{(1)})_{1 \leq j \leq q_1}$  and  $(\mathbf{u}_j^{(1)})_{1 \leq j \leq q_1}$  are shared across all time periods  $t \geq 0$ . This means that we assume a stationary (stochastic) process.
- The upper index  $^{(1)}$  indicates the fact that this is the first (and single) RN layer in this example. In this sense, Figs. 8.1 and 8.2 show a shallow RN network. In the next section we are going to discuss deep RN networks, and below we are also going to discuss how the output is modeled, i.e., how the response  $Y_{T+1}$  is predicted based on the pre-processed features  $(\mathbf{z}_t)_{0 \leq t \leq T} \in \mathbb{R}^{q_1 \times (T+1)}$ .

### 8.2.2 Deep Recurrent Neural Network Architectures

There are many different ways of extending a shallow RN network to a deep RN network. Assume we want to model a RN network of depth  $d \geq 2$ . A first (obvious) way of receiving a deep RN network architecture is

$$\mathbf{z}_t^{[1]} = \mathbf{z}^{(1)}(\mathbf{x}_t, \mathbf{z}_{t-1}^{[1]}) \in \mathbb{R}^{q_1}, \quad (8.5)$$

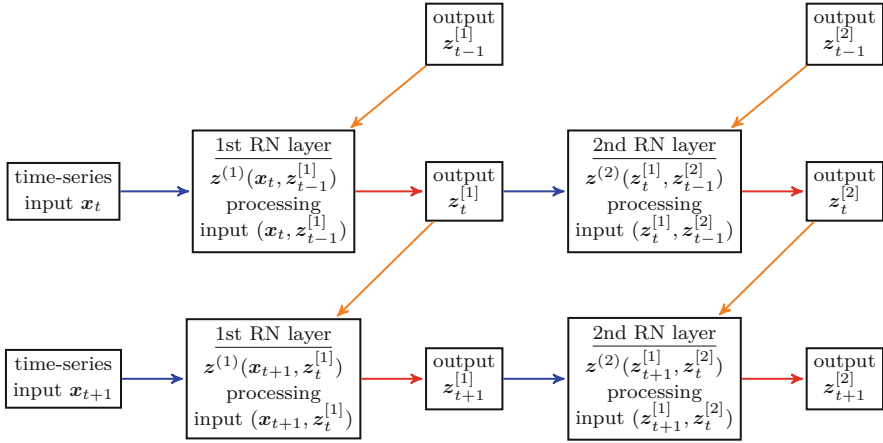
$$\mathbf{z}_t^{[m]} = \mathbf{z}^{(m)}(\mathbf{z}_t^{[m-1]}, \mathbf{z}_{t-1}^{[m]}) \in \mathbb{R}^{q_m} \quad \text{for } 2 \leq m \leq d, \quad (8.6)$$

where all RN layers  $\mathbf{z}^{(m)}$ ,  $1 \leq m \leq d$ , are of type (8.3)–(8.4), and additionally we include an intercept component in the RN layers  $\mathbf{z}^{(m)}$ ,  $2 \leq m \leq d$ . We add the upper indices (in square brackets  $[\cdot]$ ) to the time-series  $(\mathbf{z}_t^{[m]})_t$  to indicate which RN layer outputs these learned representations (memory processes). In fact, we could also write  $\mathbf{z}_t^{[m:1]}$  instead of  $\mathbf{z}_t^{[m]}$ , because in  $\mathbf{z}_t^{[m:1]}$  the feature input  $\mathbf{x}_{0:t}$  has been processed through  $m$  RN layers  $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}$ . For simplicity, we just use the notation  $\mathbf{z}_t^{[m]} = \mathbf{z}_t^{[m]}(\mathbf{x}_{0:t})$ .

We are going to use the following abbreviation for a RN layer  $m \geq 1$

$$\mathbf{z}_t^{[m]} = \mathbf{z}^{(m)} \left( \mathbf{z}_t^{[m-1]}, \mathbf{z}_{t-1}^{[m-1]} \right) = \phi \left( \left\langle \mathbf{W}^{(m)}, \mathbf{z}_t^{[m-1]} \right\rangle + \left\langle \mathbf{U}^{(m)}, \mathbf{z}_{t-1}^{[m-1]} \right\rangle \right) \in \mathbb{R}^{q_m}, \quad (8.7)$$

where the weights  $\mathbf{W}^{(m)} = (\mathbf{w}_1^{(m)}, \dots, \mathbf{w}_{q_m}^{(m)}) \in \mathbb{R}^{(q_{m-1}+1) \times q_m}$  include the intercept components, and the weights  $\mathbf{U}^{(m)} = (\mathbf{u}_1^{(m)}, \dots, \mathbf{u}_{q_m}^{(m)}) \in \mathbb{R}^{q_m \times q_m}$  do not include any intercept components. The scalar product is understood column-wise in the weight matrices  $\mathbf{W}^{(m)}$  and  $\mathbf{U}^{(m)}$ , and the activation  $\phi$  is understood component-wise. Moreover, we initialize for the input  $\mathbf{z}_t^{[0]} = \mathbf{x}_t$ .



**Fig. 8.3** Unfolded representation of a RN network architecture of depth  $d = 2$

Figure 8.3 shows the RN network architecture of depth  $d = 2$  defined in (8.5)–(8.6). The dimension of the input  $\mathbf{z}_t^{[0]} = \mathbf{x}_t \in \mathcal{X} \subseteq \{1\} \times \mathbb{R}^{q_0}$  is  $q_0 + 1$ , the first RN layer has  $q_1$  neurons and the second RN layer  $q_2$  neurons. From this graph it becomes clear how a RN network architecture of any depth  $d \in \mathbb{N}$  can be constructed (recursively).

*Remark 8.2* There are many alternative ways in building deep RN networks. E.g., we can add a loop that connects the output of the second RN layer back to the first one

$$\begin{aligned} \mathbf{z}_t^{[1]} &= \mathbf{z}^{(1)} \left( \mathbf{x}_t, \mathbf{z}_{t-1}^{[1]}, \mathbf{z}_{t-1}^{[2]} \right), \\ \mathbf{z}_t^{[2]} &= \mathbf{z}^{(2)} \left( \mathbf{z}_t^{[1]}, \mathbf{z}_{t-1}^{[2]} \right), \end{aligned}$$

or we can add a skip connection from the input variable  $\mathbf{x}_t$  to the second RN layer

$$\begin{aligned} \mathbf{z}_t^{[1]} &= \mathbf{z}^{(1)}(\mathbf{x}_t, \mathbf{z}_{t-1}^{[1]}), \\ \mathbf{z}_t^{[2]} &= \mathbf{z}^{(2)}(\mathbf{x}_t, \mathbf{z}_t^{[1]}, \mathbf{z}_{t-1}^{[2]}). \end{aligned}$$

We refrain from explicitly studying such RN network variants any further.

### 8.2.3 Designing the Network Output

There remains to explain how to predict the response variable  $Y_{T+1}$  based on the pre-processed features (memory processes)  $\mathbf{z}_T^{[1]}, \dots, \mathbf{z}_T^{[d]}$ , outputted by the RN network of depth  $d \geq 1$ . Typically, only the final output of the last RN layer  $\mathbf{z}_T^{[d]} = \mathbf{z}_T^{[d]}(\mathbf{x}_{0:T}) \in \mathbb{R}^{q_d}$  is considered to predict the response  $Y_{T+1}$ . We take this output and feed it into a FN network  $\bar{\mathbf{z}}^{(D:1)} : \{1\} \times \mathbb{R}^{q_d} \rightarrow \{1\} \times \mathbb{R}^{\tilde{q}_D}$  of depth  $D \in \mathbb{N}$  and with FN layers  $\bar{\mathbf{z}}^{(m)}$ ,  $1 \leq m \leq D$ , given by (7.5). Moreover, we choose a strictly monotone and smooth link function  $g$ .

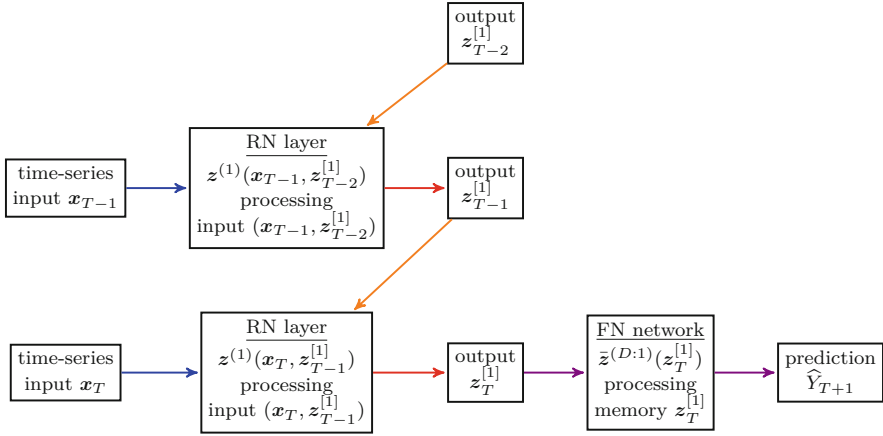
This then provides us with the regression function, see (7.7)–(7.8),

$$\mathbf{x}_{0:T} \mapsto \mathbb{E}[Y_{T+1} | \mathbf{x}_{0:T}] = \mu(\mathbf{x}_{0:T}) = g^{-1} \left\langle \boldsymbol{\beta}, \bar{\mathbf{z}}^{(D:1)} \left( \mathbf{z}_T^{[d]}(\mathbf{x}_{0:T}) \right) \right\rangle. \quad (8.8)$$

Thus, we first process the time-series features  $\mathbf{x}_{0:T}$  through a RN network to receive the learned representation  $\mathbf{z}_T^{[d]}(\mathbf{x}_{0:T}) \in \mathbb{R}^{q_d}$  at time  $T$ . This learned representation is then used as a feature input to a FN network  $\bar{\mathbf{z}}^{(D:1)}$  that allows us to predict the response  $Y_{T+1}$ . This is illustrated in Fig. 8.4 for depth  $d = 1$ .

#### Remarks 8.3

- From the graph in Fig. 8.4 it also becomes apparent that we can consider different insurance policies  $1 \leq i \leq n$  having different lengths of the corresponding histories  $\mathbf{x}_{i,T-\tau_i:T} \in \mathbb{R}^{(q_0+1) \times (\tau_i+1)}$ ,  $\tau_i \in \{0, \dots, T\}$ . The stationarity assumption allows us to enter the network in Fig. 8.4 at any time  $T - \tau_i$ . The RN network encodes this history into a learned feature  $\mathbf{z}_T^{[1]}(\mathbf{x}_{i,T-\tau_i:T})$  which is then decoded by the FN network  $\bar{\mathbf{z}}^{(D:1)}$  to forecast  $Y_{i,T+1}$ .
- If there is additional insurance policy dependent feature information  $\tilde{\mathbf{x}}_i$  that is not of a time-series structure, we can concatenate the feature information  $(\mathbf{z}_T^{[d]}(\mathbf{x}_{i,0:T}), \tilde{\mathbf{x}}_i)$  which then enters the FN network (8.8).



**Fig. 8.4** Forecasting the response  $Y_{T+1}$  using a RN network (8.8) based on a single RN layer  $d = 1$  and on a FN network of depth  $D$

There remains to fit this network architecture having  $d$  RN layers and  $D$  FN layers to the available data. The RN layers involve the network weights  $W^{(m)} \in \mathbb{R}^{(q_{m-1}+1) \times q_m}$  and  $U^{(m)} \in \mathbb{R}^{q_m \times q_m}$ , for  $1 \leq m \leq d$ , and the FN layers involve the network weights  $(\tilde{w}_j^{(m)})_{1 \leq j \leq \tilde{q}_m} \in \mathbb{R}^{(\tilde{q}_{m-1}+1) \times \tilde{q}_m}$ , for  $1 \leq m \leq D$ , and with  $\tilde{q}_0 = q_d$ . Moreover, we have an output parameter  $\beta \in \mathbb{R}^{\tilde{q}_D+1}$ . The fitting is again done by a gradient descent algorithm minimizing the corresponding objective function.

Assume we have independent (in  $i$ ) data  $(Y_{i,T+1}, \mathbf{x}_{i,0:T}, v_{i,T+1})$  of the cases  $1 \leq i \leq n$ . We then assume that the responses  $Y_{i,T+1}$  can be modeled by a fixed member of the EDF having unit deviance  $\mathfrak{d}$ . We consider the deviance loss function, see (4.9),

$$\boldsymbol{\vartheta} \mapsto \mathfrak{D}(Y_{T+1}, \boldsymbol{\vartheta}) = \frac{1}{n} \sum_{i=1}^n \frac{v_{i,T+1}}{\varphi} \mathfrak{d}\left(Y_{i,T+1}, \mu_{\boldsymbol{\vartheta}}(\mathbf{x}_{i,0:T})\right), \quad (8.9)$$

for the observations  $\mathbf{Y}_{T+1} = (Y_{1,T+1}, \dots, Y_{n,T+1})^\top$ , and where  $\boldsymbol{\vartheta}$  collects all the RN and FN network weights/parameters of the regression function (8.8). This model can now be fitted using a variant of the gradient descent algorithm. The variant uses back-propagation through time (BPTT) which is an adaption of the back-propagation method to calculate the gradient w.r.t. the network parameter  $\boldsymbol{\vartheta}$ .

### 8.2.4 Time-Distributed Layer

There is a special feature in RN network modeling which is called a *time-distributed layer*. Observe from Fig. 8.4 that the deviance loss function (8.9) only focuses on the

final observation  $Y_{i,T+1}$ . However, the stationarity assumption allows us to output and study any (previous) observation  $Y_{i,t+1}$ ,  $0 \leq t \leq T$ . A time-distributed layer considers applying the deep FN network (8.8) *simultaneously* at all time points  $0 \leq t \leq T$ ; simultaneously meaning that we use the same FN network weights for all  $t$ . The latter is justified under the assumption of having stationarity.

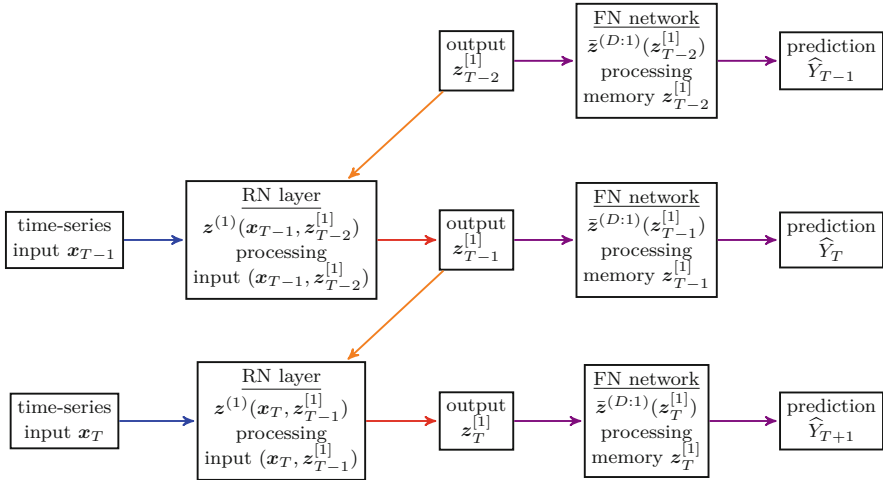
This then provides us with the regressions

$$\mathbf{x}_{0:t} \mapsto \mathbb{E}[Y_{t+1}|\mathbf{x}_{0:t}] = \mu(\mathbf{x}_{0:t}) = g^{-1}\left(\boldsymbol{\beta}, \bar{z}^{(D:1)}\left(z_t^{[d]}(\mathbf{x}_{0:t})\right)\right) \quad \text{for all } t \geq 0. \quad (8.10)$$

Figure 8.5 illustrates a time-distributed output where we predict  $(Y_{t+1})_t$  based on the history  $(\mathbf{x}_{0:t})_t$ , and we always apply the same FN network  $\bar{z}^{(D:1)}$  to the memory  $z_t^{[1]} = z_t^{[1]}(\mathbf{x}_{0:t})$ .

A time-distributed layer changes the fitting procedure. Instead of considering the objective function (8.9) for the final observation  $Y_{i,T+1}$ , we now include all observations  $\mathbf{Y} = (Y_{i,t+1})_{0 \leq t \leq T, 1 \leq i \leq n}$  into the objective function. This results in studying the deviance loss function

$$\boldsymbol{\vartheta} \mapsto \mathfrak{D}(\mathbf{Y}, \boldsymbol{\vartheta}) = \frac{1}{n} \sum_{i=1}^n \frac{1}{T+1} \sum_{t=0}^T \frac{v_{i,t+1}}{\varphi} \mathfrak{d}\left(Y_{i,t+1}, \mu_{\boldsymbol{\vartheta}}(\mathbf{x}_{i,0:t})\right). \quad (8.11)$$



**Fig. 8.5** Forecasting  $(Y_{t+1})_t$  using a RN network (8.10) based on a single RN layer  $d = 1$  and using a time-distributed FN layer for the outputs

Note that this can easily be adapted if the different cases  $1 \leq i \leq n$  have different lengths in their histories. An example is provided in Listing 10.8, below.

### 8.3 Special Recurrent Neural Networks

In the plain-vanilla RN networks introduced above we have defined the memory processes  $(z_t^{[m]})_{t \geq 0}$ ,  $1 \leq m \leq d$ , which encode the information history  $(\mathbf{x}_t)_{t \geq 0}$  through different RN layers in a temporal causal way. This is naturally done through the use of a time-series structure as illustrated, e.g., in Fig. 8.5. There are more specific RN network architectures that allow the memory processes to be of a long memory or a short memory type. In this section, we present the two most popular architectures that pay a special attention to the memory storage. This is the long short-term memory (LSTM) architecture introduced by Hochreiter–Schmidhuber [188] and the gated recurrent unit (GRU) architecture proposed by Cho et al. [76].

#### 8.3.1 Long Short-Term Memory Network

The LSTM network of Hochreiter–Schmidhuber [188] is the most commonly used RN network architecture. The LSTM network uses simultaneously three different activation functions for different purposes, the sigmoid and hyperbolic tangent activation functions, respectively,

$$\phi_\sigma(x) = \frac{1}{1 + e^{-x}} \in (0, 1) \quad \text{and} \quad \phi_{\tanh}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \in (-1, 1),$$

and a general activation function  $\phi : \mathbb{R} \rightarrow \mathbb{R}$ , see also Table 7.1.

The LSTM network relies on several RN layers that are of the same structure as the plain-vanilla RN layer given in (8.7). We start by defining three different so-called *gates* that all have the RN layer structure (8.7). These three gates are used to model the memory cell of the LSTM network. Choose a layer index  $m \geq 1$  and assume that  $z_t^{[m-1]}$  is modeled by the previous layer  $m - 1$ ; for  $m = 1$  we initialize  $z_t^{[0]} = \mathbf{x}_t$ . The three gates are then defined as follows, set  $t \geq 1$ :

- The *forget gate* models the loss of memory rate

$$f_t^{[m]} = f^{(m)}(z_t^{[m-1]}, z_{t-1}^{[m]}) = \phi_\sigma^f(\langle W_f^{(m)}, z_t^{[m-1]} \rangle + \langle U_f^{(m)}, z_{t-1}^{[m]} \rangle) \in (0, 1)^{q_m},$$

with the network weights  $W_f^{(m)} \in \mathbb{R}^{(q_{m-1}+1) \times q_m}$  and  $U_f^{(m)} \in \mathbb{R}^{q_m \times q_m}$ , and with the sigmoid activation function  $\phi_\sigma^f = \phi_\sigma$ , we also refer to (8.7).

- The *input gate* models the memory update rate

$$\mathbf{i}_t^{[m]} = \mathbf{i}^{(m)} \left( \mathbf{z}_t^{[m-1]}, \mathbf{z}_{t-1}^{[m]} \right) = \phi_\sigma^i \left( \left\langle \mathbf{W}_i^{(m)}, \mathbf{z}_t^{[m-1]} \right\rangle + \left\langle \mathbf{U}_i^{(m)}, \mathbf{z}_{t-1}^{[m]} \right\rangle \right) \in (0, 1)^{q_m},$$

with the network weights  $\mathbf{W}_i^{(m)} \in \mathbb{R}^{(q_{m-1}+1) \times q_m}$  and  $\mathbf{U}_i^{(m)} \in \mathbb{R}^{q_m \times q_m}$ , and with the sigmoid activation function  $\phi_\sigma^i = \phi_\sigma$ .

- The *output gate* models the release of memory information rate

$$\mathbf{o}_t^{[m]} = \mathbf{o}^{(m)} \left( \mathbf{z}_t^{[m-1]}, \mathbf{z}_{t-1}^{[m]} \right) = \phi_\sigma^o \left( \left\langle \mathbf{W}_o^{(m)}, \mathbf{z}_t^{[m-1]} \right\rangle + \left\langle \mathbf{U}_o^{(m)}, \mathbf{z}_{t-1}^{[m]} \right\rangle \right) \in (0, 1)^{q_m}, \quad (8.12)$$

with the network weights  $\mathbf{W}_o^{(m)} \in \mathbb{R}^{(q_{m-1}+1) \times q_m}$  and  $\mathbf{U}_o^{(m)} \in \mathbb{R}^{q_m \times q_m}$ , and with the sigmoid activation function  $\phi_\sigma^o = \phi_\sigma$ .

These gates have outputs in  $(0, 1)$ , and they determine the relative amount of memory that is updated and released in each step. The so-called *cell state process*  $(\mathbf{c}_t^{[m]})_t$  is used to store the relevant memory. Given  $\mathbf{z}_t^{[m-1]}$ ,  $\mathbf{z}_{t-1}^{[m]}$  and  $\mathbf{c}_{t-1}^{[m]}$ , the updated cell state is defined by

$$\begin{aligned} \mathbf{c}_t^{[m]} &= \mathbf{c}^{(m)} \left( \mathbf{z}_t^{[m-1]}, \mathbf{z}_{t-1}^{[m]}, \mathbf{c}_{t-1}^{[m]} \right) \\ &= \mathbf{f}_t^{[m]} \odot \mathbf{c}_{t-1}^{[m]} + \mathbf{i}_t^{[m]} \odot \phi_{\tanh} \left( \left\langle \mathbf{W}_c^{(m)}, \mathbf{z}_t^{[m-1]} \right\rangle + \left\langle \mathbf{U}_c^{(m)}, \mathbf{z}_{t-1}^{[m]} \right\rangle \right) \in \mathbb{R}^{q_m}, \end{aligned} \quad (8.13)$$

with the network weights  $\mathbf{W}_c^{(m)} \in \mathbb{R}^{(q_{m-1}+1) \times q_m}$  and  $\mathbf{U}_c^{(m)} \in \mathbb{R}^{q_m \times q_m}$ , and  $\odot$  denotes the Hadamard product. This defines how the memory (cell state) is updated and passed forward using the forget and the input gates  $\mathbf{f}_t^{[m]}$  and  $\mathbf{i}_t^{[m]}$ , respectively.

The neuron activations  $\mathbf{z}_t^{[m]}$  are updated, given  $\mathbf{z}_t^{[m-1]}$ ,  $\mathbf{z}_{t-1}^{[m]}$  and  $\mathbf{c}_t^{[m]}$ , by

$$\mathbf{z}_t^{[m]} = \mathbf{z}^{(m)} \left( \mathbf{z}_t^{[m-1]}, \mathbf{z}_{t-1}^{[m]}, \mathbf{c}_t^{[m]} \right) = \mathbf{o}_t^{[m]} \odot \phi \left( \mathbf{c}_t^{[m]} \right) \in \mathbb{R}^{q_m}, \quad (8.14)$$

with the cell state  $\mathbf{c}_t^{[m]}$  given in (8.13) and the output gate  $\mathbf{o}_t^{[m]}$  defined in (8.12). Figure 8.6<sup>2</sup> shows a LSTM cell (8.13)–(8.14) which includes four RN layers (8.7) for the forget gate  $\mathbf{f}^{(m)}$ , the input gate  $\mathbf{i}^{(m)}$ , the output gate  $\mathbf{o}^{(m)}$  and in the cell state update (8.13). These RN layers are combined using the Hadamard product  $\odot$  resulting in the updated cell state  $\mathbf{c}_t^{[m]}$  and the learned representation  $\mathbf{z}_t^{[m]}$  both being functions of the inputs  $\mathbf{x}_{0:t}$ .

<sup>2</sup> This figure is based on colah's blog explaining LSTMs <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>.