Transformation models for correlated observations using “Template Model Builder”

Master Thesis in Biostatistics (STA495)

by

Bálint Tamási
17-730-706

supervised by

Prof. Dr. Torsten Hothorn

Zurich, June 2019

Transformation models for correlated observations using "Template Model Builder"

Bálint Tamási

Version June 26, 2019

Abstract

Transformation models provide a general and flexible approach to model the distribution function of the response variable of interest. The maximum likelihood estimation described by Hothorn *et al.* (2018) assumes independent observations, and thus needs to be modified to accommodate data with correlated outcomes, which are abundant in the statistical practice. This master thesis proposes a possible extension of the model class to incorporate mixed effects that can be used to estimate transformation models on grouped data. We also implement maximum likelihood estimation and inference in this set of models using the Template Model Builder package by Kristensen *et al.* (2016) in the `R` programming language. Our implementation provides a formula-based interface to specify stratified linear transformation mixed models with complex random effects structures. Through a series of examples on various datasets and model specifications, we present the features of our package and compare the results to estimates from available alternatives. The new mixed effects transformation model approach not only poses a viable general alternative of several popular mixed effects regression packages, but also provides features that are not available elsewhere.

Contents

Chapter 1

Introduction

All regression models aim to capture some important aspect of a conditional distribution. Some approaches model the conditional mean (e.g. normal linear regression), while others parameterize different properties of the distribution, such as the odds, the hazard, quantile, or the density. The choice of this characteristic implies the properties of the resulting statistical model as well as the conditions the model has to fulfill. In the paper *Most Likely Transformations*, Hothorn *et al.* (2018) describe a general approach that parameterizes the cumulative distribution function of an outcome variable directly, using a flexible monotone increasing transformation function that is estimated from the data. They not only show that many regression approaches for at least ordered outcomes can be regarded as special cases of the transformation model framework, but also provide a full likelihood procedure to estimate models of this class.

In modern applied statistical practice, problems involving grouped data structures are very common. Longitudinal studies, hierarchical designs, multiple measurements can all give rise to grouped data. A common property of these potentially complicated structures is that the observations are expected to be correlated within groups defined by some grouping factors and cannot be treated as independent. To make the inference valid, the statistical model has to somehow take the correlation structure of the observations into account. One common approach to incorporate the correlation into the statistical model is the use of mixed effects, where the model includes random common effects that take on the same values for observations in the same group. This way not only the correlation structure of the data can be modelled, but also various sources of variabilities (group-level or idiosyncratic) can be distinguished.

The goal of this study is to provide an extension to the transformation model framework for grouped data, where the observations within groups are correlated, by introducing mixed effects. Moreover, we aim to implement the extended framework in the statistical software `R` using the Template Model Builder by Kristensen *et al.* (2016). Finally, as a proof of concept, we compare results calculated with our implementation to those from other available `R` packages.

1.1 Outline

The structure of the thesis is as follows: Chapter 2 gives a quick overview of transformation models and introduces a possible extension of the model with mixed effects. Numerical aspects of the maximum likelihood estimation and inference are also discussed. In Chapter 3, we outline a possible implementation of the estimation of the proposed mixed effects model in the `R` programming language. Chapter 4 demonstrates, through a series of examples, that our implementation is a viable alternative of various other `R` packages in estimating different types of regression models, furthermore, for some specific problems, it is the only currently available tool. Chapter 5 lays out possible directions for future work and concludes.

1.2 Notation

The mathematical notation used in the study is standard in the statistical literature. Latin letters typically denote data, while Greek letters specify parameters. To distinguish from scalar values, we use boldface letters for vectors, e.g. ϑ and $\boldsymbol{\vartheta}$. Parameters with hats (e.g. $\widehat{\boldsymbol{\vartheta}}$) indicate estimates. Random variables are denoted by capital letters, and boldface capitals stand for, in most of the cases, *random vectors*, e.g. $\boldsymbol{\Gamma} = (\Gamma_1, \Gamma_2, \dots, \Gamma_q)^\top$. In some exceptional cases, boldface capital letters can also indicate *constant matrices*, which are always explicitly indicated.

$F_{Y|\mathbf{X}=\boldsymbol{x}}(y \mid \boldsymbol{x})$ stands for the cumulative distribution function of the conditional distribution of Y, i.e. $F_{Y|\mathbf{X}=\boldsymbol{x}}(y \mid \boldsymbol{x}) = \mathbb{P}(Y \leq y \mid \mathbf{X} = \boldsymbol{x})$. Similarly, $f_{Y|\mathbf{X}=\boldsymbol{x}}(y \mid \boldsymbol{x})$ denotes the conditional probability density (or mass) function.

General mixed effects models are usually defined using only one set of random effects. The reason for this is that the case with arbitrary number of grouping factors is notationally cumbersome, and the simplification normally does not lose the generality of the definition.

1.3 Software

The calculations in this study are carried out using the `R` statistical software (R Core Team, 2018). Further details on packages and software versions are given below.

- R version 3.5.2 (2018-12-20), `x86_64-apple-darwin18.2.0`
- Running under: `macOS Mojave 10.14.5`
- Matrix products: default
- BLAS/LAPACK: `/usr/local/Cellar/openblas/0.3.5/lib/libopenblasp-r0.3.5.dylib`
- Base packages: base, datasets, graphics, grDevices, methods, splines, stats, utils
- Other packages: basefun 1.0-4, bdsmatrix 1.3-3, boot 1.3-20, coxme 2.2-10, Deriv 3.8.5, knitr 1.21, lme4 1.1-21, Matrix 1.2-15, mlt 1.0-4, optimx 2018-7.10, ordinal 2019.3-9, ordinalCont 2.0.1, parfm 2.7.6, survival 2.43-3, TMB 1.7.15, tram 0.2-5, tramm 0.0.0.9000, variables 1.0-1, xtable 1.8-3
- Loaded via a namespace (and not attached): alabama 2015.3-1, BB 2014.10-1, codetools 0.2-16, compiler 3.5.2, coneproj 1.14, digest 0.6.18, evaluate 0.12, expm 0.999-4, Formula 1.2-3, grid 3.5.2, highr 0.7, lattice 0.20-38, magrittr 1.5, MASS 7.3-51.1, minqa 1.2.4, mnormt 1.5-5, msm 1.6.7, multcomp 1.4-10, mvtnorm 1.0-10, nlme 3.1-137, nloptr 1.2.1, numDeriv 2016.8-1, orthopolynom 1.0-5, polynom 1.4-0, quadprog 1.5-5, Rcpp 1.0.1, sandwich 2.5-1, sn 1.5-3, stats4 3.5.2, stringi 1.2.4, stringr 1.3.1, TH.data 1.0-10, tools 3.5.2, ucminf 1.1-4, xfun 0.4, zoo 1.8-5

Chapter 2

Mixed Effects Transformation Models

This chapter provides an overview of the class of transformation models, and proposes a possible extension to include mixed effects. By introducing random effects in the transformation model framework, we can account for correlated observations. Section 2.1 summarizes the concept of transformation models with a focus on linear transformation models. Section 2.2 introduces mixed effects in this latter model class. Section 2.3 describes the theory of maximum likelihood estimation of these models.

2.1 Transformation Models

The concept of transformation models was first described in the seminal paper by Box and Cox (1964), who introduced non-linear transformations of the response variable to conform to the normality assumption of a regression model. Since then, the idea has been generalized and extended to other functional forms and distributional assumptions. In fact, many of the most commonly applied statistical models can be interpreted as transformation models.

Hothorn *et al.* (2018) summarize the core idea of transformation models as reformulating an unknown distribution function by the application of a strictly monotonic transformation, $h(y)$, i.e. $\mathbb{P}(Y \leq y) = \mathbb{P}(h(Y) \leq h(y))$, where the transformation function itself is estimated from the data.

In the context of regression models, transformation models can be understood as models that directly parameterize the conditional cumulative distribution function of an outcome variable, Y, by introducing an unknown, non-linear transformation, $h(y \mid \boldsymbol{x})$, which is estimated from the data:

$$\mathbb{P}\left(Y \leq y \mid \mathbf{X} = \boldsymbol{x}\right) = F_Z(h(y \mid \boldsymbol{x})). \tag{2.1}$$

The term $h(y \mid \boldsymbol{x})$ in (2.1) is referred to as the (conditional) *transformation function*, which typically maps from the sample space of Y, Ξ, to $\mathbb{R}$. The function F_Z is assumed to be strictly monotonic increasing on the interval $(-\infty, \infty)$, with $F_Z(-\infty) = 0$ and $F_Z(\infty) = 1$, and has a known functional form. The typical choices of F_Z are the cumulative distribution function of the standard normal, standard logistic, minimum extreme value, and maximum extreme value distributions, hence F_Z is often called the *error distribution*.

The conditional transformation function can take on various forms, and treatments of the general case can be found for example in Hothorn *et al.* (2014) or Hothorn *et al.* (2018). In this thesis, we focus on (partially) *linear transformation models*, where $h(y \mid \boldsymbol{x})$ has an additive structure: $h(y \mid \boldsymbol{x}) = h_Y(y) + h_{\boldsymbol{x}}(\boldsymbol{x})$. More specifically, we consider the linear transformation models,

$$\begin{aligned}\mathbb{P}\left(Y \leq y \mid \mathbf{X} = \boldsymbol{x}\right) &= F_Z(h(y \mid \boldsymbol{x})) \\ &= F_Z(h_Y(y) - \boldsymbol{x}^\top \boldsymbol{\beta}) \\ &= F_Z(\boldsymbol{a}(y)^\top \boldsymbol{\vartheta} - \boldsymbol{x}^\top \boldsymbol{\beta}). \end{aligned} \tag{2.2}$$

As Hothorn *et al.* (2018) point out, the transformation model can be represented by the triple $(F_Z, (\boldsymbol{a}^\top, \boldsymbol{x}^\top)^\top, (\boldsymbol{\vartheta}^\top, -\boldsymbol{\beta}^\top)^\top)$, which in turn fully defines the conditional distribution $F_{Y|\mathbf{X}=\boldsymbol{x}}$.

In (2.2), the *baseline transformation* function is reformulated with the help of a *basis transformation* of the outcome variable: $h_Y(y) = \boldsymbol{a}(y)^\top \boldsymbol{\vartheta}$. The actual choice of $\boldsymbol{a}(y)$ depends on the type of the response variable and the complexity of the model. While for certain classes of models, such as regression models for ordinal responses, explicit parametrization is common in the literature (see e.g. Tutz (2011), Chapter 9), other cases call for less restrictive formulations. Hothorn (2018) approximates the unknown baseline transformation with a general, flexible, and smooth function using the Bernstein polynomials. The various choices of the basis transformations in some specific cases are discussed in more detail in Chapter 4.

In the linear transformation model given by (2.2), the higher moments of the conditional distribution $F_{Y|\mathbf{X}=\boldsymbol{x}}$ are fully determined by the baseline transformation function, $h_Y(y)$, and are not affected by the explanatory variables ($\mathbf{X}$) of the model, which only act as shifting factors on the location of the distribution. To relax this restrictive assumption, Hothorn (2019) introduces *stratum variables* ($\mathbf{S}$) in the model:

$$\begin{aligned}\mathbb{P}\left(Y \leq y \mid \mathbf{S} = \boldsymbol{s}, \mathbf{X} = \boldsymbol{x}\right) &= F_Z(h_Y(y \mid \boldsymbol{s}) - \boldsymbol{x}^\top \boldsymbol{\beta}) \\ &= F_Z(\boldsymbol{c}(y, \boldsymbol{s})^\top \boldsymbol{\vartheta} - \boldsymbol{x}^\top \boldsymbol{\beta}),\end{aligned}$$

where $\boldsymbol{c}(y, \boldsymbol{s}) = \boldsymbol{a}(y) \otimes \boldsymbol{b}(\boldsymbol{s})$, i.e. the Kronecker product between the baseline transformation of the outcome variable and a basis function of the stratum variable. In the specific case of a discrete factor $\mathbf{S}$, $b(\boldsymbol{s})$ represents a row of the design matrix corresponding to the coding of the given stratum variable.

The direct estimation of the conditional distribution function of Y requires additional restrictions on the model parameters: To ensure that $F_{Y|\mathbf{X}=\boldsymbol{x}}$ is monotonic increasing, the baseline transformation needs to be a monotone increasing function of y, which in turn introduces restrictions on the $\boldsymbol{\vartheta}$ parameters in (2.2). When the basis $\boldsymbol{a}(y)$ is defined by the Bernstein polynomials of order M, i.e. $\boldsymbol{a} = \boldsymbol{a}_{\mathrm{Bs},M}$, $h_Y(y)$ is monotonic increasing when the elements of $\boldsymbol{\vartheta}$ are monotone increasing.

2.2 Mixed Effects Transformation Models

Grouped data structures are very common in applied statistics, which can arise from repeated measurements, longitudinal studies, and hierarchical designs. An important property of these structures is that the observations within groups are correlated, and cannot be treated as independent if we want to make valid inference based on them. Mixed effects models have become important tools to deal with correlated observations. Moreover, as Demidenko (2004, Chapter 1) points out, with mixed effects models, we can distinguish different sources of variations, deal with parameter multidimensionality, and they represent a compromise between frequentist and Bayesian approaches by obtaining shrinkage estimates of regression coefficients.

The linear transformation model can be extended to include mixed effects in a straightforward way: Assuming a single grouping factor for the random effects (and ignoring stratification),

$$\begin{gathered}\mathbb{P}\left(Y_{ij} \leq y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i\right) = F_Z(h_Y(y_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i), \qquad (2.3) \\ \boldsymbol{\Gamma}_i \sim \mathcal{N}_q\left(\mathbf{0}, \boldsymbol{\Sigma}\right), \\ i = 1, \ldots, m, \quad j = 1, \ldots, n_i,\end{gathered}$$

where, for measurement j in group i, $\boldsymbol{x}_{ij}$ and $\boldsymbol{u}_{ij}$ denote the covariate vectors corresponding to the fixed and random effects, respectively. $\boldsymbol{\Gamma}_i$ is the vector of random effects for the group (or subject) i, and it can contain (possibly correlated) random intercepts and slopes.

An important aspect of the model described in (2.3) is the independence of the random effects between various groups. The assumption that $\boldsymbol{\Gamma}_i$ is uncorrelated with $\boldsymbol{\Gamma}_{i'}$ can be restrictive in certain cases. In Section 4.4.2 we will present such a case, and show how the implementation used throughout this thesis can be modified to accommodate this extension. Otherwise we will restrict our attention to the models with independent identically distributed random effects.

The mixed effects transformation model in (2.3) is not the only conceivable way of extending the transformation model framework to incorporate random effects. Due to its additive structure, the random effects, just as the fixed effects, do not affect the higher moments of the conditional distribution of the outcome variable. One of the main advantages of this structure is that we do not have to introduce additional restrictions on the values of the random effects in order to maintain the monotonicity of the conditional distribution function.

Although the assumption that the random effects are normally distributed is very common in the literature of mixed effects models, in some specific model classes, e.g. frailty models for time-to-event outcomes, other distribution types are also frequently used. Nevertheless, the normally distributed random effects offer a general and numerically appealing option, hence we focus only on them.

In this thesis we will concentrate solely on models described in (2.3), and leave the possible extensions to other directions for further research. In any case, as we will see, this model structure provides great flexibility and several widely applied regression models can be regarded as its special cases. The various models that arise from the specific choices of the error distribution (F_Z) and baseline transformation ($h_Y(y)$) as well as the structure of the covariance matrix of the random effects ($\boldsymbol{\Sigma}$) are discussed in more detail in Chapter 4.

The mixed effects transformation models are *conditional models*, in the sense that, by the inclusion of random effects, we are conditioning on the unobserved characteristics of the groups. The observations are assumed to be *conditionally* (i.e. conditioned on the random effects) *independent*. This is in contrast with the marginal models, which represent the other main approach of modeling grouped data. Marginal models do not condition on group-specific random effects, and do not assume conditional independence of the observations, but treat their covariance structure as nuisance parameters of the model. In their comparison of the two modeling types, Muff *et al.* (2016) explain that the main difference lies in the interpretation of the fixed effects parameters: In marginal models, the parameters can be regarded as averaged (over various groups) effects, which are sometimes, and somewhat misleadingly, interpreted as population averages. In contrast, fixed effects coefficients in conditional models, i.e. $\boldsymbol{\beta}$ parameters in the transformation model (2.3), must be interpreted conditionally on the group-specific values of the random effects ($\boldsymbol{\gamma}_i$).

2.3 Estimation and Inference

2.3.1 Log-likelihood Function

Hothorn *et al.* (2018) describe the maximum likelihood estimation and inference in the class of transformation models with fixed effects only. In this section, we adapt the most important results of the article to mixed effects transformation models as presented in Equation 2.3.

The marginal log-likelihood function of a general (and possibly non-linear) mixed effects model can be derived by integrating the joint likelihood over the random effects, as described e.g. by McCulloch and Searle (2001, Section 8.4) in the context of generalized linear mixed models. Without explicitly conditioning on the fixed vector $\boldsymbol{x}_{ij}$, we can write the log-likelihood function, given the data, as

$$\ell(\boldsymbol{\vartheta}, \boldsymbol{\beta}, \boldsymbol{\Sigma}) = \log \left(\prod_{i=1}^{m} \int_{\boldsymbol{\gamma}_i \in \mathbb{R}^q} \prod_{j=1}^{n_i} f_{Y_{ij} \mid \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i}(y_{ij} \mid \boldsymbol{\gamma}_i; \boldsymbol{\vartheta}, \boldsymbol{\beta}) f_{\boldsymbol{\Gamma}_i}(\boldsymbol{\gamma}_i; \boldsymbol{\Sigma}) \, \mathsf{d}\boldsymbol{\gamma}_i \right). \tag{2.4}$$

To derive the conditional likelihood contribution of a singe observation, i.e. $\mathcal{L}_{ij}(\boldsymbol{\vartheta}, \boldsymbol{\beta} \mid \boldsymbol{\gamma}_i) = f_{Y_{ij} \mid \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i}(y_{ij} \mid \boldsymbol{\gamma}_i; \boldsymbol{\vartheta}, \boldsymbol{\beta})$ in the context of transformation models, under an independent censoring mechanism, we distinguish among four cases: In the case of "exact continuous" observations, we can measure the outcome variable precisely. In the case of "right censored" observations, we only know that the outcome is larger than a certain value, i.e. $y_{ij} \in (\underline{y}_{ij}, \infty)$. Conversely, with "left censored" data we have $y_{ij} \in (-\infty, \bar{y}_{ij}]$, while the "interval censored" case can be considered as the combination of the latter two: Instead of directly observing the value of y_{ij}, we only know that it falls within the interval $(\underline{y}_{ij}, \bar{y}_{ij}]$. Since the mixed effects transformation model parameterizes the conditional distribution function of the outcome variable, the censored cases can be expressed easily, and the "exact continuous" observations can be regarded as special cases of the "interval censored" observations on an infinitesimally short interval. Using basic properties of the cumulative distribution function, the likelihood contributions are

$$\mathcal{L}_{ij}(\boldsymbol{\vartheta}, \boldsymbol{\beta} \mid \boldsymbol{\gamma}_i) = \begin{cases} f_Z(h_Y(y_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i) h'_Y(y_{ij}) & y_{ij} \in \Xi & \text{‘exact continuous’} \\ 1 - F_Z(h_Y(\underline{y}_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i) & y_{ij} \in (\underline{y}_{ij}, \infty) \cap \Xi & \text{‘right censored’} \\ F_Z(h_Y(\bar{y}_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i) & y_{ij} \in (-\infty, \bar{y}_{ij}] \cap \Xi & \text{‘left censored’} \\ \begin{aligned} & F_Z(h_Y(\bar{y}_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i) \\ & \quad - F_Z(h_Y(\underline{y}_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i) \end{aligned} & y_{ij} \in (\underline{y}_{ij}, \bar{y}_{ij}] \cap \Xi & \text{‘interval censored’,} \end{cases}$$

where f_Z denotes the probability density function of the error distribution.

The likelihood contributions can be modified to include truncated observations in the interval $(\underline{y}^*_{ij}, \bar{y}^*_{ij}] \subset \Xi$ as

$$\mathcal{L}^*_{ij}(\boldsymbol{\vartheta}, \boldsymbol{\beta} \mid \boldsymbol{\gamma}_i) = \frac{\mathcal{L}_{ij}(\boldsymbol{\vartheta}, \boldsymbol{\beta} \mid \boldsymbol{\gamma}_i)}{F_Z(h_Y(\bar{y}^*_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i) - F_Z(h_Y(\underline{y}^*_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i)},$$

when $\underline{y}^*_{ij} < \underline{y}_{ij} < \bar{y}_{ij} \leq \bar{y}^*_{ij}$.

By incorporating these special cases, mixed effects transformation models can be applied on a broad range of problems in practice. The fact that the suitable conditional likelihood contribution can be plugged into (2.4) depending on whether the corresponding observation is exact, censored or truncated, highlights that these details only concern the estimation of the statistical model in question and they are not essential parts of it. This approach, which is ultimately due to defining our statistical model by parameterizing the cumulative distribution function, lends a great flexibility to the model class described in this thesis.

2.3.2 Laplace Approximation

The possibly multidimensional integral in (2.4) is analytically intractable in the general case, and requires numerical techniques to be evaluated. One of the most common methods to numerically evaluate such integrals in mixed effects models is the *Laplace approximation*.

Pinheiro and Bates (1995) give a detailed description of the method in the context of nonlinear mixed effects models. They point out that the Laplacian approximation is based on the quadratic Taylor expansion of the logarithm of the joint likelihood function (i.e. the integrand in (2.4)) about its mode. Let $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1{}^\top, \boldsymbol{\gamma}_2{}^\top, \ldots, \boldsymbol{\gamma}_m{}^\top)^\top$, moreover, let $\boldsymbol{\theta}$ denote all of the model parameters to be estimated, i.e. $\boldsymbol{\theta} = (\boldsymbol{\vartheta}^\top, \boldsymbol{\beta}^\top, \mathrm{vech}(\boldsymbol{\Sigma})^\top)^\top$. The marginal likelihood of the model can be written then as

$$\mathcal{L}(\boldsymbol{\theta}) = \int_{\boldsymbol{\gamma} \in \mathbb{R}^{mq}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\gamma}) \, \mathsf{d}\boldsymbol{\gamma}. \tag{2.5}$$

The conditional mode of the random effects as a function of $\boldsymbol{\theta}$ is given by

$$\widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta}) = \underset{\boldsymbol{\gamma}}{\operatorname{argmax}} \, \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\gamma}),$$

and the second order Taylor expansion of the joint log-likelihood about this mode is

$$\ell(\boldsymbol{\theta}, \boldsymbol{\gamma}) \approx \ell(\boldsymbol{\theta}, \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})) - \frac{1}{2}(\boldsymbol{\gamma} - \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta}))^\top \mathbf{H}(\boldsymbol{\theta})(\boldsymbol{\gamma} - \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})), \tag{2.6}$$

where

$$\mathbf{H}(\boldsymbol{\theta}) = -\left.\frac{\partial^2 \ell(\boldsymbol{\theta}, \boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}^2}\right|_{\boldsymbol{\gamma} = \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})}.$$

Note, that the first order term disappears from the Taylor approximation, because $\widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})$ is also a maximizer of $\ell(\boldsymbol{\theta}, \boldsymbol{\gamma}) = \log \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\gamma})$, hence $\partial \ell(\boldsymbol{\theta}, \boldsymbol{\gamma}) / \partial \boldsymbol{\gamma}|_{\boldsymbol{\gamma} = \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})} = \mathbf{0}$.

Using (2.6), we get the approximation of (2.5) as

$$\begin{aligned}
\mathcal{L}^{\mathrm{LA}}(\boldsymbol{\theta}) &\approx \int_{\boldsymbol{\gamma} \in \mathbb{R}^{mq}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\gamma}) \, \mathsf{d}\boldsymbol{\gamma} \\
&\approx \int_{\boldsymbol{\gamma} \in \mathbb{R}^{mq}} e^{\ell(\boldsymbol{\theta}, \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})) - \frac{1}{2}(\boldsymbol{\gamma} - \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta}))^\top \mathbf{H}(\boldsymbol{\theta})(\boldsymbol{\gamma} - \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta}))} \, \mathsf{d}\boldsymbol{\gamma} \\
&= \mathcal{L}(\boldsymbol{\theta}, \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})) \int_{\boldsymbol{\gamma} \in \mathbb{R}^{mq}} e^{-\frac{1}{2}(\boldsymbol{\gamma} - \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta}))^\top \mathbf{H}(\boldsymbol{\theta})(\boldsymbol{\gamma} - \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta}))} \, \mathsf{d}\boldsymbol{\gamma} && (2.7) \\
&= \mathcal{L}(\boldsymbol{\theta}, \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})) |\mathbf{H}(\boldsymbol{\theta})^{-1}|^{\frac{1}{2}} (2\pi)^{\frac{mq}{2}} \\
&= \mathcal{L}(\boldsymbol{\theta}, \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})) |\mathbf{H}(\boldsymbol{\theta})|^{-\frac{1}{2}} (2\pi)^{\frac{mq}{2}}. && (2.8)
\end{aligned}$$

The integrand in (2.7) is the kernel of a multivariate Gaussian distribution with the *precision matrix* $\mathbf{H}(\theta)$. By multiplying and dividing by a suitable normalizing constant, we arrive at the expression in Equation (2.8).

When the log of the integrand to be approximated by Laplace's method is quadratic in the random effects, i.e. the integrand $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\gamma})$ is the kernel of a multivariate normal distribution, as it is in the case of normal linear mixed effects models, the approximation is exact. In other cases maximizing the marginal likelihood approximated with this method will result in some errors in the parameter estimates. While the Laplace approximation is asymptotically exact, i.e. the joint log-likelihood converges to a quadratic function as the number of observations *per group* increases, in practical applications the accuracy of the approximation can be imprecise, especially when the outcome variable is discrete or the group sizes are small. For detailed evaluations of the accuracy of this method, see Pinheiro and Chao (2006) and Joe (2008).

One of the main advantages of using Laplace's method over its alternatives, such as adaptive Gaussian quadrature algorithms, is that it is computationally very efficient. For this reason, it is used in many of the most popular software implementations to estimate non-linear mixed effects models. In the implementation to be presented in Chapter 3, we also use the Laplace approximation to evaluate the integral in the log-likelihood function of the mixed effects transformation models.

2.3.3 Maximum Likelihood Estimation and Inference

Given the data, the baseline transformation (h_Y), and the error distribution (F_Z), the parameters of the mixed effects transformation model in (2.4) can be estimated by numerically maximizing the log of the Laplace-approximated marginal likelihood in (2.8),

$$\begin{aligned}
\widehat{\boldsymbol{\theta}}_{\mathrm{ML}} &= \operatorname*{argmax}_{\boldsymbol{\theta}} \ell^{\mathrm{LA}}(\boldsymbol{\theta}), \\
\ell^{\mathrm{LA}}(\boldsymbol{\theta}) &= \ell(\boldsymbol{\theta}, \widehat{\boldsymbol{\gamma}}(\boldsymbol{\theta})) - \frac{1}{2}\log|\mathbf{H}(\boldsymbol{\theta})| + \frac{mq}{2}\log(2\pi), \\
\boldsymbol{\theta} &= (\boldsymbol{\vartheta}^\top, \boldsymbol{\beta}^\top, \operatorname{vech}(\boldsymbol{\Sigma})^\top)^\top.
\end{aligned}$$

Some additional details about the numerical optimization are discussed in Chapter 3, where we present the computer implementation of the estimation procedure.

Since the model proposed in this study is fully parametric, standard asymptotic maximum likelihood theory may be employed to test hypotheses and calculate confidence intervals of the parameters, with the additional assumption that the error distribution is log-concave. The covariance matrix of the parameters can be estimated as the inverse of the Fisher information matrix evaluated at $\widehat{\boldsymbol{\theta}}_{\mathrm{ML}}$:

$$\operatorname{Cov}\left(\widehat{\boldsymbol{\theta}}_{\mathrm{ML}}\right) = -\left(\left.\frac{\partial^2 \ell^{\mathrm{LA}}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^2}\right|_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}_{\mathrm{ML}}}\right)^{-1}.$$

In Chapter 4, by demonstrating the features of the computer implementation, we give examples how standard likelihood methods can be applied to make inference in the class of mixed effects transformation models.

Chapter 3

Implementation

This chapter presents the implementation of the estimation and inference in mixed effects transformation models in the `R` programming language. First, the main building blocks of our approach are discussed: Section 3.1 gives a description of the `R` packages `mlt` and `tram` that provide a versatile and flexible framework for defining and estimating fixed effects transformation models. Section 3.2 presents the Template Model Builder, a tool for estimating general and possibly highly complex, non-linear mixed effects models in `R`, which is the main workhorse behind our implementation of transformation mixed model estimation. Section 3.3 summarises the formula module of the package `lme4` that allows the user to define mixed effects models in an intuitive way by extending the model formulas of base `R`. Finally, in Section 3.4 we introduce the package `tramm` that combines the functionalities of the previous three building blocks to implement estimation and inference in the class of mixed effects transformation models of the form presented in Chapter 2.

3.1 Transformation Models with the `mlt` and `tram` Packages

The `mlt` package by Hothorn (2018) implements the maximum likelihood estimation in transformation models, which is described by Hothorn *et al.* (2018). The framework allows the user to specify and estimate either unconditional (i.e. $F_Y(y) = F_Z(h_Y(y))$) or conditional transformation models ($F_{Y|\mathbf{X}=\boldsymbol{x}}(y \mid \boldsymbol{x}) = F_Z(h_Y(y \mid \boldsymbol{x}))$). In addition to model specification and estimation, `mlt` also provides tools for model diagnostics, interpretation, inference, and simulation.

Building on the infrastructure implemented by the helper package `basefun`, the user can specify flexible transformation models based on various basis functions. Applying either discrete basis functions to discrete or discretized continuous responses, or Bernstein bases to continuous outcomes, results in very flexible statistical models that do not impose excessively restrictive assumptions on the unconditional or conditional transformation functions. These models can potentially capture very complex relationships between the dependent variable and covariates in a regression setting.

With the help of the `variables` package, the properties (such as its support, bounds and unit) of the outcome variable can be defined in advance of the actual modeling phase. Moreover, `mlt` allows the user to define censored or truncated values for the outcome variable. Since transformation models parameterize the cumulative distribution function of the outcome, the evaluation of this function, and hence the estimation on censored and truncated data is straightforward and computationally inexpensive.

The `tram` package by Hothorn (2019), building on the functionality of `mlt`, provides a formula-based interface for specifying a broad range of stratified linear transformation models. The package focuses on the most useful types of specifications that fit in the class of transformation models defined in Section 2.1. The estimation and inference of the specified models are done using the functions of the `mlt` package.

The main advantage of `tram` is that it makes it easier for the user to specify the models. Utilizing formulas, extended with a notation to incorporate stratum variables, stratified linear specifications can take the form

```
resp | s1 + ... + sk ~ x1 + ... + xn
```

where `resp` indicates the response variable, `s1 + ... + sk` denotes the specification of the stratum variables and `x1 + ... + xn` defines the specification of the shift effects.

The choice of the error distribution and the baseline transformation is specified through various functions in `tram`. Table 3.1 summarizes the available choices in the form of the triple $(F_Z, (\boldsymbol{a}^\top, \boldsymbol{x}^\top)^\top, (\boldsymbol{\vartheta}^\top, \boldsymbol{\beta}^\top)^\top)$ for the general model $\mathbb{P}(Y \le y \mid \mathbf{X} = \boldsymbol{x}) = F_Z(\boldsymbol{a}(y)^\top \boldsymbol{\vartheta} + \boldsymbol{x}^\top \boldsymbol{\beta})$.

Table 3.1: Model specifications available in the `tram` package. Error distributions: Φ denotes the normal cumulative distribution function, F_{SL} the standard logistic, F_{MEV} the minimum extreme value, and F_{Gumbel} the maximum extreme value (Gumbel) distribution. F_Z indicates that multiple options are available in the given model class. Baseline transformations: $\boldsymbol{a}_{\text{Bs},M}$ stands for the Bernstein basis of order M, $\boldsymbol{e}_{K-1}(k)$ denotes the discrete basis of K levels, while $\boldsymbol{a}$ indicates that multiple options are available in the given model class.

Function	Name	Specification
`Lm`	Normal linear regression	$(\Phi, (y, 1, \boldsymbol{x}^\top)^\top, (\vartheta_1, \vartheta_2, -\boldsymbol{\beta}^\top)^\top)$
`BoxCox`	Box-Cox model	$(\Phi, (\boldsymbol{a}_{\text{Bs},M}{}^\top, \boldsymbol{x}^\top)^\top, (\boldsymbol{\vartheta}^\top, -\boldsymbol{\beta}^\top)^\top)$
`Colr`	Continuous outcome logistic regression	$(F_{SL}, (\boldsymbol{a}_{\text{Bs},M}{}^\top, \boldsymbol{x}^\top)^\top, (\boldsymbol{\vartheta}^\top, \boldsymbol{\beta}^\top)^\top)$
`Polr`	Regression models for ordinal outcomes	$(F_Z, (\boldsymbol{e}_{K-1}(k)^\top, \boldsymbol{x}^\top)^\top, (\boldsymbol{\vartheta}^\top, -\boldsymbol{\beta}^\top)^\top)$
`Coxph`	Cox proportional hazards model	$(F_{MEV}, (\boldsymbol{a}_{\text{Bs},M}{}^\top, \boldsymbol{x}^\top)^\top, (\boldsymbol{\vartheta}^\top, \boldsymbol{\beta}^\top)^\top)$
`Survreg`	Parametric survival models	$(F_Z, (\boldsymbol{a}^\top, \boldsymbol{x}^\top)^\top, (\boldsymbol{\vartheta}^\top, -\boldsymbol{\beta}^\top)^\top)$
`Lehmann`	Lehmann-alternative linear regression	$(F_{Gumbel}, (\boldsymbol{a}_{\text{Bs},M}{}^\top, \boldsymbol{x}^\top)^\top, (\boldsymbol{\vartheta}^\top, -\boldsymbol{\beta}^\top)^\top)$

As Table 3.1 shows, many different types of regression models can be estimated within the `tram` framework. Some of the functions estimate models implied by a specific choice of error distribution and baseline transformation pair, such as `Lm` (Gaussian error, linear transformation), `Colr` (standard logistic error, Bernstein polynomial basis) or `Coxph` (minimum extreme value error distribution, Bernstein polynomial basis). Other functions allow to select the error distribution or the baseline transformation: Depending on the choice of F_Z, `Polr` estimates the proportional odds logistic regression, probit regression, or proportional hazards regression (i.e. sequential model) for ordinal outcomes. The function `Survreg` specifies various well known parametric survival regression models, for example Weibull, exponential, Rayleigh, log-normal, log-logistic, etc.

More details on the various model types and the interpretations of the parameters are discussed in Chapter 4, where we consider the same model specifications extended with random effects.

3.2 Template Model Builder

The Template Model Builder (`TMB`) by Kristensen *et al.* (2016) is an `R` package for fitting general non-linear mixed effects models. It was built on state-of-the-art `C++` libraries (such as `CppAD`, `Eigen`), and as a result, it estimates statistical models with possibly complex random effects structures very efficiently. For this reason, `TMB` has become a popular tool for estimating mixed models, and even some `R` packages build on its functionality. A notable example is the `glmmTMB` package by Brooks *et al.* (2017).

In essence, the `TMB` package performs two tasks: First, it integrates out the random effects of a given model by approximating the marginal likelihood using Laplace's method. Section 2.3 described the *Laplace approximation* in the context of a general mixed effects model and applied it to the problem of estimating transformation mixed models.

Moreover, `TMB` calculates the first and second derivatives of the objective function with *automatic* (or algorithmic) *differentiation* (AD). Unlike symbolic differentiation, AD calculates the even higher derivatives of general functions efficiently; and in contrast with numerical differentiation techniques (e.g. finite differences), AD does not suffer from truncation and cancellation errors. AD evaluates the derivatives of a function given by a computer program by breaking it down to basic computational steps and elementary functions and applying the *chain rule* to numerical values at each of these steps. The various modes and technical details of AD can be found in many textbooks, interested readers are referred to, for example, Griewank and Walther (2008).

The user has to supply the objective function, i.e. the negative log-likelihood, written in `C++`, but the processing of the data before model fitting and the post-processing of the results is typically done in `R` through the interface `TMB` provides to it. Using AD, the program calculates the gradient and Hessian functions, which can be used in turn to find the parameter estimates utilizing standard `R` optimization functions. Moreover, `TMB` provides methods to calculate parameter covariance matrices, standard errors of derived parameters using the *delta method*, as well as methods for calculating profile likelihoods. With this functionality, it is possible to perform asymptotic likelihood inference on the models estimated with this package.

3.2.1 A Simple Simulated Example

To demonstrate the usage of the `TMB` package, we estimate the simple random effects model,

$$\begin{aligned} Y_{ij} \mid U_i = u_i &= \mu + u_i + \epsilon_{ij}, \\ U_i &\sim \mathcal{N}(0, \sigma_u), \\ \epsilon_{ij} &\sim \mathcal{N}(0, \sigma_\epsilon), \end{aligned}$$

on a simulated dataset.

The negative log-likelihood function is defined in `C++`:

```
if (!file.exists("code/re1w.cpp")) {
  writeLines({"
  // One-way random effects model
  #include <TMB.hpp>

  template<class Type>
  Type objective_function<Type>::operator() ()
  {
    DATA_VECTOR(y);          // Data vector from R
    DATA_FACTOR(group);      // Group vector from R
    PARAMETER(mu);           // Population mean
    PARAMETER_VECTOR(u);     // Random effects
    PARAMETER(log_sig_u);    // log-SD of u
    PARAMETER(log_sig_e);    // log-SD of e
    Type sigma_u = exp(log_sig_u);
    Type sigma_e = exp(log_sig_e);
    ADREPORT(sigma_u); // Report the transformed parameters
    ADREPORT(sigma_e);

    Type nll = Type(0); // negative log-likelihood
    nll -= dnorm(u, Type(0), Type(1), true).sum();
    for (int i = 0; i < y.size(); i++) {
      nll -= dnorm(y(i), mu + sigma_u * u(group(i)), sigma_e, true);
    }
    return nll;
  }
  "}, con = "code/re1w.cpp")
}
```

Note, that the model is parameterized using the logarithms of the standard deviations in order to avoid optimizing over a restricted parameter space. These parameters are then transformed back to their original scale by the program.

A balanced dataset of 40 groups and five observations per group is generated using `R`.

```
set.seed(123)
n <- 40 ## # of groups
k <- 5  ## # of observations
sig_u <- 1.5
sig_e <- 1
mu <- 1
u <- rep(rnorm(n, 0, sig_u), each = k)
y <- mu + u + rnorm(n*k, 0, sig_e)
```

To use the previously defined objective function, we need to compile and load the `C++` code.

```
library("TMB")
invisible(capture.output(compile("code/re1w.cpp")))
dyn.load(dynlib("code/re1w"))
```

Finally, we have to define the data structure and the initial values of the parameters. It should be noted that so far we have not distinguished between random effects and fixed effects. To get the (negative) marginal log-likelihood, we have to indicate which parameters shall be treated as random effects (in our case `u`) and integrated out from the joint log-likelihood using the Laplace approximation (see Section 2.3 for more details).

```
data <- list(y = y, group = factor(rep(1:n, each = k)))
pars <- list(mu = 0, u = rep(0, n), log_sig_u = 0, log_sig_e = 0)
obj <- MakeADFun(data, pars, random = "u", DLL = "re1w", silent = TRUE)
opt <- nlminb(obj$par, obj$fn, obj$gr)
rep <- sdreport(obj)
```

As the example shows, we use a standard `R` optimization function, `nlminb`, to find the maximum likelihood estimates. Of course, other optimization libraries could also be applied, since `TMB` provides access to the objective function and its gradient (and also its Hessian) within `R`.

The estimated value of the population mean ($\widehat{\mu}$) and its standard error are

```
c(rep$par.fixed["mu"], sqrt(diag(rep$cov.fixed)["mu"]))

##        mu        mu
## 1.0522821 0.2288484
```

while the point estimates of the variance components ($\widehat{\sigma}_u^2$ and $\widehat{\sigma}_\epsilon^2$) are

```
rep$value^2

##   sigma_u   sigma_e
## 1.9030215 0.9592136
```

3.3 Model Formulation

The `R` package `lme4` by Bates *et al.* (2015), which implements the estimation of linear and generalized linear mixed effects models, uses a special notation to define the random effects structure. As an extension of the original Wilkinson-Rogers type notation of base `R` (R Core Team, 2018), the `(expr | group)` syntax was originally introduced by Pinheiro *et al.* (2018) in the `nlme` package.

By using the `(expr | group)` operator, the formula module of `lme4` defines the interaction between the model matrix implied by the expression `expr` and the grouping factor `group`. With these interactions, the columns of the model matrix get different effect sizes for each level of the grouping variable. In this regard, the random effects are similar to any other fixed effects of the model. The main difference is in how the random effects are treated in the estimation of the model.

A general mixed effects model using the formula notation could be written as

```
resp ~ FE_expr + (RE_expr1 | group1) + (RE_expr2 | group2) + ...
```

where `resp` is the response variable, `FE_expr` stands for the fixed effects structure, and the model contains *fully crossed* random effects defined by the grouping variables `group1`, `group2`.

Table 3.2 summarizes the main options of random effects specifications using the formula module of `lme4`. These basic structures can then be combined to define more complex mixed effects models.

Table 3.2: Examples of mixed effects specifications using the formula module of `lme4`. `x` denotes a covariate, while `group`, `group1`, `group2` are grouping factors for random effects.

Formula	Specification
`(1 | group)`	Random intercept
`x + (x | group)`	Correlated random intercept and slope
`x + (x || group)`	Uncorrelated random intercept and slope
`(1 | group1) + (1 | group2)`	Crossed random intercepts
`(1 | group1/group2)`	Nested random intercepts (`group2` within `group1`)

3.4 The `tramm` Package

In this section, we introduce an implementation of the mixed effects transformation model estimation discussed in Section 2.2 and Section 2.3. The `R` code accompanying this thesis is organized into a package called `tramm`, standing for "transformation mixed models".

The package utilizes the functionalities discussed in the previous three sections to provide a set of functions with formula-based interfaces to define and estimate models of the form (2.3). Note that, as it was discussed is Section 2.2, the random effects only enter the transformation model specification as shift terms, and in turn they do not affect the higher moments of the conditional distribution of the outcome. Furthermore, the current implementation of the estimation only allows normally distributed random effects.

Combining the formula capabilities of the `tram` and `lme4` packages, we can define stratified linear transformation mixed models with the notation

```
resp | s1 + ... + sk ~ x1 + ... + xn + (RE_spec1 | group1) +
  (RE_spec2 | group2) + ...
```

where `resp` is the response variable, `s1 + ... + sk` and `x1 + ... + xn` denote the specification of the stratum variables and fixed shift effects, respectively, while `(RE_spec1 | group1) + (RE_spec2 | group2) + ...` stands for crossed random effects defined by various grouping factors.

The choice of error distributions and baseline transformations, just as in the `tram` package, is specified through various function calls. In fact, `tramm` can be regarded as an extension of the `tram` package, which contains the same functions, with an extra `ME` suffix (i.e. `LmME`, `ColrME`, `CoxphME`, etc.).

Programmatically, the design matrix corresponding to the random effects structure is created by the formula module of `lme4`, while the model matrices implied by the specifications of the baseline transformation and the fixed effect shift terms are generated by the `mlt` package. Moreover, to set up the necessary data structures for the mixed effects model, in the first step, a fixed effects only `tram` model is estimated. The parameter estimates of this model are used as initial values for the numerical optimization of the corresponding transformation mixed model.

The numerical evaluation of the marginal log-likelihood function and its gradient, using automatic differentiation, is done by the `TMB` package. Accordingly, the negative log-likelihood function of the mixed effects transformation model is specified in `C++`. Appendix A.1 presents the program code used in the `tramm` package. The inputs of the log-likelihood function, i.e. the data vectors and matrices along with the parameter vectors, are defined and pre-processed in `R`. The `C++` code then evaluates the negative log-likelihood, given a set of fixed and random effects parameters, allowing potentially correlated random effects. Finally, it returns the value of the negative log-likelihood, alongside the transformed values of some of the parameters that can be used in `R` for optimization and reporting purposes.

Since the parameters in the vector $\boldsymbol{\vartheta}$ have to be monotone increasing in order to ensure the monotonicity of the conditional cumulative distribution function, we have to use a constrained optimization algorithm to find the maximum likelihood estimates. The `tramm` package, similarly to `mlt`, uses the Augmented Lagrangian and Adaptive Barrier Minimization Algorithm for this task, which is implemented by the function `auglag` of the `R` package `alabama` by Varadhan (2015).

The set of inferential and model checking tools currently available in the `tramm` package is restricted. The inference is primarily done using asymptotic likelihood theory. The values of the likelihood functions and likelihood ratio tests are applied for model comparisons. Besides the point estimates of the parameters, their covariance matrix is also available, which is calculated from the Hessian matrix of the log-likelihood supplied by the Template Model Builder. Wald and profile confidence intervals are used for inference on model parameters. Since the individual values of the parameters in the baseline transformation function ($\boldsymbol{\vartheta}$) are usually not of interest, the whole transformation function, which is a linear combination of these parameters, i.e. $h_Y(y) = \boldsymbol{a}(y)^\top \boldsymbol{\vartheta}$, can be evaluated at a grid of y values and plotted alongside its pointwise confidence interval.

Examples for the usage of the functions in the `tramm` package, along with demonstration of other features, are presented in Chapter 4. There we also give comparisons with other available `R` implementations for certain specific regression models.

Chapter 4

Applications

The goal of this chapter is twofold: First, we want to showcase the features of the `tramm` package using a series of regression problems with different types of outcome variables and various model complexities. Moreover, when it is possible, we compare our implementation with alternative packages to test the correctness of `tramm`.

For this reason, we first estimate normal linear mixed effects models in Section 4.1, which is extended in Section 4.2 with general baseline transformation to get a flexible parametric version of the Box-Cox mixed regression model. By changing the error distribution to standard logistic, we arrive at the continuous outcome logistic mixed model, which is discussed in Section 4.3. Mixed effects regression models for discrete ordinal outcomes are presented through two examples in Section 4.4. Finally, we estimate models for time-to-event outcomes (Cox regression and parametric survival models) in Section 4.5.

It must be emphasised that the examples presented here are, by no means, intended as complete analyses. In many cases, the modeling choices were driven by illustration purposes, and the interpretations of the results are omitted. The broad range of applications presented in the chapter demonstrates the versatility of the mixed effects transformation model approach, which is not only a viable alternative of several packages, but also represents an extension on the mixed effects regression models currently available in `R`.

4.1 Normal Linear Mixed Model

The first example is the normal linear regression model with mixed effects. The conditional model can be written as

$$\begin{aligned} Y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \widetilde{\boldsymbol{\Gamma}}_i = \widetilde{\boldsymbol{\gamma}}_i &= \alpha + {\boldsymbol{x}_{ij}}^\top \widetilde{\boldsymbol{\beta}} + {\boldsymbol{u}_{ij}}^\top \widetilde{\boldsymbol{\gamma}}_i + \epsilon_{ij}, \qquad (4.1)\\ \widetilde{\boldsymbol{\Gamma}}_i &\sim \mathcal{N}_q\left(\mathbf{0}, \widetilde{\boldsymbol{\Sigma}}\right), \\ \epsilon_{ij} &\sim \mathcal{N}(0, \sigma^2), \end{aligned}$$

where $\boldsymbol{u}_{ij}$ denotes the covariates associated with the random effects, and $\widetilde{\boldsymbol{\Gamma}}_i$ the q dimensional vector of normally distributed random effects for group i. The normal linear mixed model described in (4.1) can be reformulated as a transformation model,

$$\begin{aligned} \mathbb{P}\left(Y_{ij} \leq y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i\right) &= \Phi\left(\frac{y_{ij} - \alpha - {\boldsymbol{x}_{ij}}^\top \widetilde{\boldsymbol{\beta}} - {\boldsymbol{u}_{ij}}^\top \widetilde{\boldsymbol{\gamma}}_i}{\sigma}\right) \\ &= \Phi(\vartheta_1 y_{ij} + \vartheta_2 - {\boldsymbol{x}_{ij}}^\top \boldsymbol{\beta} - {\boldsymbol{u}_{ij}}^\top \boldsymbol{\gamma}_i), \qquad (4.2)\\ \boldsymbol{\Gamma}_i &\sim \mathcal{N}_q\left(\mathbf{0}, \boldsymbol{\Sigma}\right), \end{aligned}$$

with Φ denoting the cumulative distribution function of the standard normal. Based on (4.1) and (4.2), the parameters of the two formulations can be expressed from each other as $\vartheta_1 = \sigma^{-1}$, $\vartheta_2 = -\sigma^{-1}\alpha$, $\boldsymbol{\beta} = \sigma^{-1}\widetilde{\boldsymbol{\beta}}$, $\boldsymbol{\gamma}_i = \sigma^{-1}\widetilde{\boldsymbol{\gamma}}_i$, and $\boldsymbol{\Sigma} = \sigma^{-2}\widetilde{\boldsymbol{\Sigma}}$.

The `R` package `lme4` by Bates *et al.* (2015) implements estimation procedures for (generalized) linear mixed effects models. We will compare estimation results from the transformation model approach to estimates obtained by `lme4`, using the `sleepstudy` example dataset of this package. The dataset contains ten observations for each of the participants of a sleep deprivation study. The continuous outcome variable is the reaction time and the only covariate denotes the number of days of sleep deprivation.

The normal linear mixed regression is estimated with the function `lmer` as

```r
library("lme4")
fit_lmer <- lmer(Reaction ~ Days + (Days | Subject), data = sleepstudy,
                 REML = FALSE)
```

The model contains *correlated* random intercepts and slopes for each individual in the study. Note, that with the `REML = FALSE` option, we get the maximum likelihood estimates instead of the default restricted likelihood ones to make the results comparable to the transformation mixed model approach.

To define the equivalent mixed effects transformation model, we can use the same notation as in the previous case:

```r
library("tramm")
fit_tramm <- LmME(Reaction ~ Days + (Days | Subject), data = sleepstudy)
```

Both approaches use the Laplace approximation to evaluate the integral in the likelihood function (2.4), which gives the exact value in the normal linear case (see Section 2.3 for more details). Hence, although the two implementations rely on different optimization algorithms, the results should be numerically comparable.

Table 4.1: Comparison of parameter estimates, standard errors, and the values of the log-likelihood using `lme4` and the transformation model approach (`tramm`) on the `sleepstudy` dataset. The estimated parameters are the coefficients, the maximum likelihood estimate of the standard deviation of the error term ($\widehat{\sigma}$), the standard deviations of the random intercepts ($\widehat{\theta}_0$) and slopes ($\widehat{\theta}_1$), and the correlation between the latter two ($\widehat{\rho}$).

	`lme4`		`tramm`	
	coef	SE	coef	SE
Intercept	251.4051	6.6321	251.4051	6.6323
Days	10.4673	1.5022	10.4673	1.5022
$\widehat{\sigma}$	25.5919		25.5918	
$\widehat{\theta}_0$	23.7798		23.7806	
$\widehat{\theta}_1$	5.7168		5.7168	
$\widehat{\rho}$	0.0813		0.0803	
Log-likelihood	-875.9697		-875.9697	

Table 4.1 shows the parameter estimates, standard errors, as well as the values of the log-likelihood from the two approaches up to four decimal places. The parameter estimates of the mixed effects transformation model are rescaled with the error standard deviation to be comparable with the `lme4` parameters. The standard errors of these transformed parameters are calculated using the delta method (see e.g. Held and Bové (2014), p. 331). The results

are very close to each other and the discrepancies can be attributed to the different numerical optimization algorithms.

Based on the results in Table 4.1, the estimated correlation between the random intercepts and slopes is very small. We can investigate whether a model with *independent* random effects terms, gives a significantly different model fit. In the notation of `lme4`, we can define such model with the `( || )` operator, which is also available in the **`tramm`** package. Calling the method **`anova`** on the resulting models returns the information criteria as well as the likelihood ratio test for these nested specifications.

```
fit_lmer2 <- lmer(Reaction ~ Days + (Days || Subject), data = sleepstudy,
                  REML = FALSE)
fit_tramm2 <- LmME(Reaction ~ Days + (Days || Subject), data = sleepstudy)
sel_lmer <- anova(fit_lmer, fit_lmer2)
sel_tramm <- anova(fit_tramm, fit_tramm2)
```

The model comparisons, presented in Table 4.2, show once again identical results up to four digits for the two implementations. The likelihood ratio test supports the more parsimonious model, i.e. the one with independent random slopes and intercepts.

Table 4.2: Comparison of models with and without correlated random intercepts and slopes using `lme4` and **`tramm`**.

	df	Log-likelihood	AIC	BIC	χ^2	df(χ^2)	p-value
`fit_lmer2`	5	-876.0016	1762.0033	1777.9680			
`fit_lmer`	6	-875.9697	1763.9393	1783.0971	0.0639	1	0.8004
`fit_tramm2`	5	-876.0016	1762.0033	1777.9680			
`fit_tramm`	6	-875.9697	1763.9393	1783.0971	0.0639	1	0.8004

An advantage of writing the normal linear mixed effects model in its transformation model form, i.e. parameterizing the cumulative distribution function directly, is that it makes it easy to incorporate censored or truncated observations in the estimation. The following hypothetical scenario can serve as a demonstration: Let us assume that the shortest reaction time the timer could measure is 250 milliseconds. Moreover, we also want to take into account the fact that reaction times cannot take on negative values. In this setting, we have either exact observations, where the reaction time exceeds 250 ms, or interval-censored outcomes, where we only know that the actual reaction time falls somewhere between zero and 250 ms. The transformation model described in (4.2) is able to take this information into account in the estimation as it was detailed in Section 2.3. It is clear that treating these 36 observations out of the total 180 data points as exact observations would lead to biased estimates. For defining censored observations for the outcome variable, we can use the `Surv` function of the `survival` package.

```
library("survival")
sleepstudy$lc <- sleepstudy$rc <- sleepstudy$Reaction
sleepstudy$lc[sleepstudy$Reaction <= 250] <- 0
sleepstudy$rc[sleepstudy$Reaction <= 250] <- 250
sleepstudy$Reactionc <-  with(sleepstudy, Surv(time = lc, time2 = rc,
  type = "interval2")) ## censoring
fit_nocens <- lmer(rc ~ Days + (Days | Subject), data = sleepstudy,
                   REML = FALSE)
fit_cens <- LmME(Reactionc ~ Days + (Days | Subject), data = sleepstudy)
```

As Table 4.3 shows, the parameter estimates from the transformation model approach, which takes the censored nature of the observations correctly into account, differ from the naive approach.

Table 4.3: Comparison of results from approaches without taking (estimated with `lme4`) and taking censoring (`tramm`) into account using the `sleepstudy` dataset with censored observations at ≤ 250. The parameters are the coefficients, the maximum likelihood estimate of the standard deviation of the error term ($\widehat{\sigma}$), the standard deviations of the random intercepts ($\widehat{\theta}_0$) and slopes ($\widehat{\theta}_1$), and the correlation between the latter two ($\widehat{\rho}$).

	No censoring		Censoring	
	coef	SE	coef	SE
Intercept	259.2287	4.8090	247.6452	8.6083
Days	9.6902	1.5272	10.6398	1.8173
$\widehat{\sigma}$	24.9021		27.3654	
$\widehat{\theta}_0$	14.2147		30.256	
$\widehat{\theta}_1$	5.8709		6.7787	
$\widehat{\rho}$	0.0022		-0.1094	
Log-likelihood	-866.1154		-733.8499	

4.2 Box-Cox Mixed Model

A possible extension of the normal linear mixed effects model as formulated in (4.2) allows the baseline transformation, which was linear in y_{ij} in the normal linear case, i.e. $h_Y(y_{ij}) = \vartheta_1 y_{ij} + \vartheta_2$, to be a flexible, smooth function of the outcome. The `tram` and `tramm` packages both use Bernstein polynomials to capture this general function. With this generalization, (4.2) becomes

$$\begin{aligned}\mathbb{P}\left(Y_{ij} \leq y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i\right) &= \Phi\left(h_Y(y_{ij}) - {\boldsymbol{x}_{ij}}^\top \boldsymbol{\beta} - {\boldsymbol{u}_{ij}}^\top \boldsymbol{\gamma}_i\right), \\ &= \Phi\left({\boldsymbol{a}_{\mathrm{Bs},M}}^\top \boldsymbol{\vartheta} - {\boldsymbol{x}_{ij}}^\top \boldsymbol{\beta} - {\boldsymbol{u}_{ij}}^\top \boldsymbol{\gamma}_i\right),\end{aligned} \tag{4.3}$$

which can be regarded as an extension to the approach that Box and Cox (1964) proposed in their seminal article.

Fitting the Box-Cox mixed effects regression model to the `sleepstudy` dataset and comparing the results to the normal linear regression approach allows us to inspect deviations of the conditional distribution of the outcome from normality.

```r
data("sleepstudy", package = "lme4")
fit_nl <- LmME(Reaction ~ Days + (Days | Subject), data = sleepstudy)
fit_bc <- BoxCoxME(Reaction ~ Days + (Days | Subject), data = sleepstudy,
                   order = 6)
```

Note that the attribute `order` in the function call `BoxCoxME` sets the order of the Bernstein polynomials (M).

Figure 4.1 depicts the baseline transformation function in the Box-Cox model. Small departure normality can be detected, especially in the lower tail of the conditional distribution, as the baseline transformation does not fall on a straight line.

Table 4.4 compares the estimation results of the normal linear and Box-Cox mixed effects regression models. Although the fixed effect estimates are very close to each other, the correlations of the random effects differ between the two models. The higher value of the log-likelihood suggests that the Box-Cox model fits the data better than the normal linear approach.

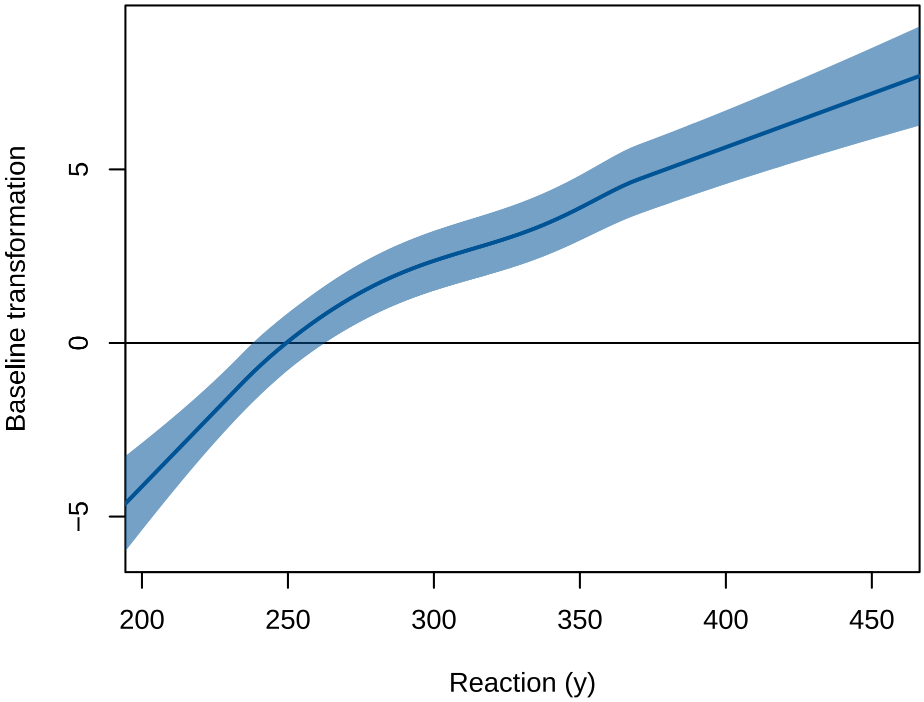

Figure 4.1: Baseline transformation in the Box-Cox model of the `sleepstudy` dataset. The solid line denotes the point estimate and the blue area is the 95% pointwise confidence interval.

Table 4.4: Comparison of the normal linear mixed effects transformation model and the Box-Cox regression model of the `sleepstudy` dataset. The estimated parameters are the coefficients, the standard deviations of the random intercepts ($\widehat{\sigma}_0$) and slopes ($\widehat{\sigma}_1$), and the correlation between the latter two ($\widehat{\rho}$).

	`LmME`		`BoxCoxME`	
	coef	SE	coef	SE
Days	0.4090	0.0635	0.4147	0.0641
$\widehat{\sigma}_0$	0.9292		1.612	
$\widehat{\sigma}_1$	0.2234		0.2236	
$\widehat{\rho}$	0.0803		-0.1956	
Log-likelihood	-875.9697		-859.5455	

4.3 Continuous Outcome Logistic Regression

A disadvantage of the Box-Cox model in (4.3), compared to the normal linear mixed model, is that the direct interpretation of the fixed effects coefficients is cumbersome. As a possible solution, one can change the error distribution, F_Z, to the standard logistic distribution and switch the signs of the fixed and random effects,

$$\begin{aligned}\mathbb{P}\left(Y_{ij} \leq y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i\right) &= F_{SL}\left(h_Y(y_{ij}) + {\boldsymbol{x}_{ij}}^\top \boldsymbol{\beta} + {\boldsymbol{u}_{ij}}^\top \boldsymbol{\gamma}_i\right), \\ &= \frac{\exp\left(h_Y(y_{ij}) + {\boldsymbol{x}_{ij}}^\top \boldsymbol{\beta} + {\boldsymbol{u}_{ij}}^\top \boldsymbol{\gamma}_i\right)}{1 - \exp\left(h_Y(y_{ij}) + {\boldsymbol{x}_{ij}}^\top \boldsymbol{\beta} + {\boldsymbol{u}_{ij}}^\top \boldsymbol{\gamma}_i\right)},\end{aligned} \tag{4.4}$$

to get the mixed effects version of the *continuous outcome logistic regression* by Lohse *et al.* (2017). Using the Bernstein polynomial representation of the unknown baseline transformation function, the equation (4.4) can be rewritten as

$$\frac{\mathbb{P}\left(Y_{ij} \leq y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i\right)}{\mathbb{P}\left(Y_{ij} > y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i\right)} = \exp\left(\boldsymbol{a}_{\mathrm{Bs},M}{}^{\top}\boldsymbol{\vartheta} + \boldsymbol{x}_{ij}{}^{\top}\boldsymbol{\beta} + \boldsymbol{u}_{ij}{}^{\top}\boldsymbol{\gamma}_i\right),$$

which describes the conditional odds at every possible cut-off points, y_{ij}. From this form it follows that the fixed effects parameters denote log-odds ratios simultaneously for all possible logistic regressions with cut-points y_{ij}, *conditionally* on the vector of random effects. Moreover, the baseline transformation function, $h_Y(y) = \boldsymbol{a}_{\mathrm{Bs},M}{}^{\top}\boldsymbol{\vartheta}$, maps the individual cut-points to the intercepts in the corresponding logistic regressions.

The model in (4.4) is very similar to the ordinal regression for continuous variables proposed by Manuguerra and Heller (2010) to analyze outcomes from visual analog scales. The main difference between their approach and the transformation model implementation of `tramm` is in the approximation of the baseline transformation function: While Manuguerra and Heller (2010) use B-splines to approximate $h_Y(y)$, `tramm` utilizes Bernstein polynomials as proposed by Hothorn *et al.* (2018). Manuguerra *et al.* (2017) describe the penalized likelihood approach to estimate such models with the `R` package `ordinalCont`. Because the approach used in the `tramm` package is very similar to the model in `ordinalCont`, we can compare estimation results from the two using an example.

The dataset `neck_pain` in the `ordinalCont` package consists of repeated measurements of 88 individuals with chronic neck pain in a randomized, placebo-controlled study of low-level laser therapy. The outcome variable is the self-recorded neck pain on a visual analogue scale, normalized to $(0, 1)$. The measurements were recorded at baseline, after 7 weeks (at the end of the treatment), and after 12 weeks.

We estimate continuous outcome logistic mixed regression models using the functions `ocm` and `ColrME` in the packages `ordinalCont` and `tramm`, respectively.

```r
library("ordinalCont")
fit_ocm <- ocm(vas ~ laser * time + (1 | id), data = neck_pain, scale = c(0, 1))
fit_tramm <- ColrME(vas ~ laser * time + (1 | id), data = neck_pain,
                    bounds = c(0, 1), support = c(0, 1))
```

Note that we are using the default settings of both implementations to model the baseline transformation, and explicitly stating the bounds of the scale of the outcome variable in the two function calls.

Table 4.5 compares the parameter estimates, the standard errors and the general model fit based on log-likelihood values. The model estimates are similar, but not exactly the same, which is not surprising, given that the two implementations differ in how they capture the baseline transformation. The higher log-likelihood value of the model implemented by `ordinalCont::ocm` indicates better model fit, but this may be due to the higher model complexity of this approach.

The baseline transformations approximated by the two approaches are plotted in Figure 4.2. As the two curves show, the fitted baseline function using the default setting (`order=6`) of the transformation model framework is much smoother than the other one based on splines, which indicates different levels of default model complexity in the two implementations.

Figure 4.3 compares the estimated log-odds ratios and their 95% Wald confidence intervals from the two approaches. As in the case of the baseline transformations, the estimates are very close to each other, although some differences can be observed. The log-odds ratios associated with the interaction terms suggest that the laser treatment increases the odds for a patient to have a lower pain level after the treatment, even in the presence of some baseline imbalances.

Table 4.5: Comparison of model estimates using `ordinalCont` and `tramm` packages on the `neck_pain` dataset. The estimated parameters are the coefficients and the standard deviations of the random intercepts ($\widehat{\sigma}$).

	`ordinalCont`		`tramm`	
	coef	SE	coef	SE
laser = yes	-2.1374	0.6388	-2.3421	0.7037
time = 2	-0.5929	0.3528	-0.6540	0.3772
time = 3	-0.1898	0.3443	-0.2076	0.3684
(laser = yes) (time = 2)	4.4547	0.5556	4.9432	0.6195
(laser = yes) (time = 3)	3.3787	0.5184	3.7474	0.5725
$\widehat{\sigma}$	2.4638		2.7277	
Log-likelihood	205.5287		84.1368	

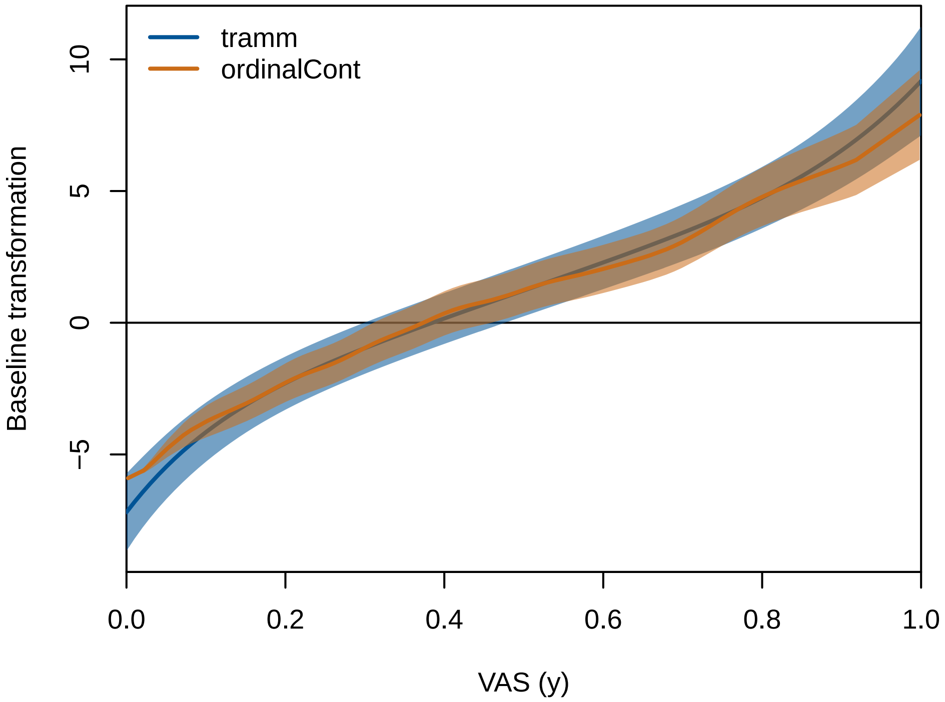

Figure 4.2: Baseline transformations in continuous outcome logistic regressions estimated with the `tramm` and `ordinalCont` packages on the `neck_pain` dataset. The solid lines denote the point estimates and the areas indicate the 95% point-wise confidence intervals.

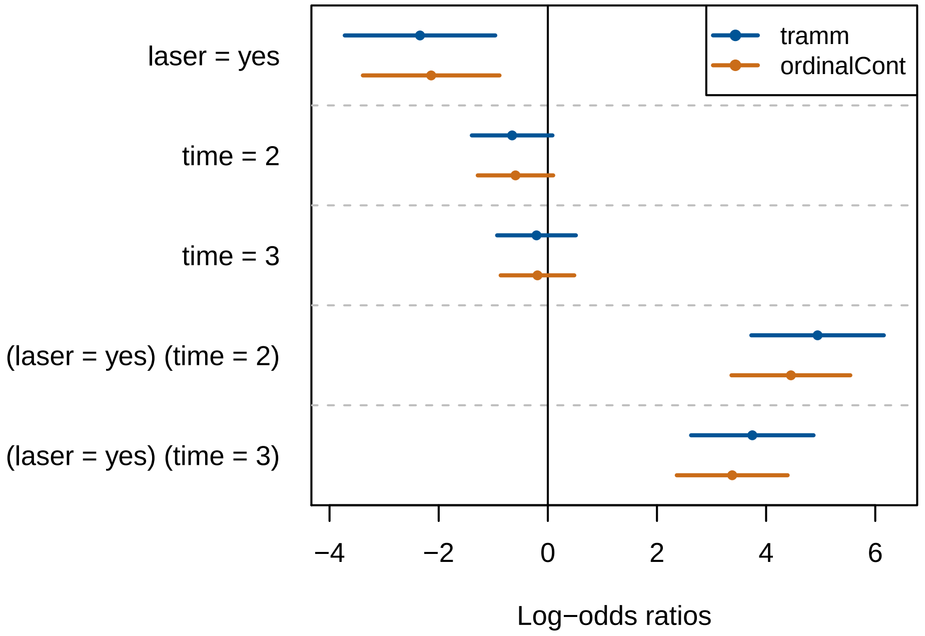

Figure 4.3: Point estimates and 95% Wald confidence intervals of the log-odds ratios from the continuous outcome logistic regressions estimated with the `tramm` and `ordinalCont` packages on the `neck_pain` dataset.

4.4 Mixed Models for Ordinal Outcomes

For ordered categorical responses with K levels, $y_1 < y_2 < \cdots < y_K$, one can write a mixed effects regression model in the transformation model form as

$$\begin{aligned} \mathbb{P}\left(Y_{ij} \leq y_k \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i\right) &= F_Z\left(h_Y(y_k) - \boldsymbol{x}_{ij}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}^\top \boldsymbol{\gamma}_i\right) \\ &= F_Z\left(\vartheta_k - \boldsymbol{x}_{ij}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}^\top \boldsymbol{\gamma}_i\right), \qquad (4.5) \\ 1 \leq k < K,& \\ \vartheta_1 < \vartheta_2 < \cdots < \vartheta_{K-1}.& \end{aligned}$$

The baseline transformation $(h_Y(y_k))$ maps each level — except the last one — to a separate parameter, with the additional restriction that the values of these parameters are increasing.

By specifying the error distribution, F_Z in (4.5), we get various familiar mixed effects regression models for ordinal outcomes: Setting F_Z to the standard logistic distribution, the resulting model is the *proportional odds* model. With $F_Z = F_{MEV}$ we arrive at the discrete version of the *proportional hazards* model, while $F_Z = \Phi$ results in the cumulative *probit* model. See e.g. Tutz (2011, Chapter 9) for a discussion of the various types of ordinal regression models.

4.4.1 Comparison with the `ordinal` Package

The `ordinal` package by Christensen (2019) implements various types of mixed effects models for ordered categorical outcomes. In the following section we compare estimates from the mixed effects transformation model approach to the results calculated with this package.

One of the example datasets in the `ordinal` package is called `wine` that collects five-level ratings of wines by nine judges at different levels of two treatment factors (`temperature` and `contact`). We can estimate a cumulative probit model with random intercepts for the judges using the function `ordinal::clmm2`.

```
library("ordinal")
fit_clmm <- clmm2(rating ~ temp + contact, random = judge, data = wine,
                  Hess = TRUE, nAGQ = 1, link = "probit")
```

By explicitly setting `nAGQ = 1` in the function call, we use the Laplace-approximated values of the marginal likelihood function for the estimation to make the results comparable to the estimates from `TMB`.

The same mixed effects model can be fitted using the transformation model implementation. By setting the `method` attribute of the `PolrME` function of **tramm**, we can specify the error distribution in the model.

```
fit_tramm <- PolrME(rating ~ temp + contact + (1 | judge), data = wine,
                    method = "probit")
```

Table 4.6 compares the parameter estimates, the standard errors, and the values of the log-likelihood function from the two approaches up to four decimal places.

Table 4.6: Comparison of parameter estimates, standard errors, and the values of the log-likelihood using `ordinal::clmm2` and `tramm::PolrME`. The parameters ϑ_i denote the threshold parameters, and σ is the standard deviation of the random intercept.

	`ordinal`		`tramm`	
	coef	SE	coef	SE
$\widehat{\vartheta}_1$	-0.9265	0.3882	-0.9265	0.3882
$\widehat{\vartheta}_2$	0.8894	0.3485	0.8894	0.3485
$\widehat{\vartheta}_3$	2.4676	0.4469	2.4676	0.4469
$\widehat{\vartheta}_4$	3.5367	0.5256	3.5367	0.5256
temp $=$ warm	1.8000	0.3269	1.8000	0.3269
contact $=$ yes	1.0482	0.2855	1.0482	0.2855
$\widehat{\sigma}$	0.6632		0.6625	
Log-likelihood	-80.9306		-80.9306	

The model in (4.5) can be extended to incorporate nominal effects, i.e. effect sizes that change with the level of the outcome variable for certain nominal covariates. These nominal effects correspond to partial proportional odds in the ordinal logistic regression framework (i.e. when the error distribution is the standard logistic). This can be implemented by assigning separate sets of threshold parameters (ϑ_i) to the different levels of the given variable, which is basically equivalent to stratifying the baseline transformation by the nominal variable. The **tramm** package uses the same notation as **tram** to specify stratum variables.

```
fit_tramm2 <- PolrME(rating | contact ~ temp + (1 | judge), data = wine,
                     method = "probit")
fit_clmm2 <- clmm2(rating ~ temp, nominal = ~ contact, random = judge,
                   data = wine, Hess = TRUE, nAGQ = 1, link = "probit")
sel_clmm <- anova(fit_clmm, fit_clmm2)
sel_tramm <- anova(fit_tramm, fit_tramm2)
```

In our example, we stratify on the variable `contact`. As Table 4.7 shows, the estimation results are, once again, essentially the same. Similarly, the likelihood-ratio tests that compare the stratified and non-stratified specifications are also the same (Table 4.8). Based on these tests, there is no evidence that the saturated model fits the data better, hence the results favor the more parsimonious model.

Table 4.7: Comparison of parameter estimates, standard errors, and the values of the log-likelihood using `ordinal::clmm2` and `tramm::PolrME`. The parameters $\vartheta_{i,j}$ denote the stratified threshold parameters, and σ is the standard deviation of the random intercept.

	`ordinal`		`tramm`	
	coef	SE	coef	SE
$\widehat{\vartheta}_{0,1}$	-0.9610	0.4266	-0.9610	0.4266
$\widehat{\vartheta}_{0,2}$	0.9056	0.3669	0.9056	0.3669
$\widehat{\vartheta}_{0,3}$	2.5725	0.4955	2.5725	0.4955
$\widehat{\vartheta}_{0,4}$	3.3716	0.5986	3.3716	0.5986
$\widehat{\vartheta}_{1,1}$	-0.9175	0.6256	-0.9176	0.6256
$\widehat{\vartheta}_{1,2}$	-1.0698	0.3789	-1.0698	0.3789
$\widehat{\vartheta}_{1,3}$	-1.2147	0.4296	-1.2147	0.4296
$\widehat{\vartheta}_{1,4}$	-0.8045	0.5405	-0.8045	0.5405
temp = warm	1.8077	0.3293	1.8077	0.3293
$\widehat{\sigma}$	0.6571		0.6564	
Log-likelihood	-80.6175		-80.6175	

Table 4.8: Comparison of models with and without stratifying on the variable `contact` using `ordinal` and `tramm`.

	df	Log-likelihood	AIC	BIC	χ^2	df(χ^2)	p-value
`fit_clmm`	7	-80.9306	175.8612	191.7979			
`fit_clmm2`	10	-80.6175	181.2350	204.0017	0.6262	3	0.8904
`fit_tramm`	7	-80.9306	175.8612	191.7979			
`fit_tramm2`	10	-80.6175	181.2350	204.0017	0.6262	3	0.8904

4.4.2 Reproducing the Results by Seibold *et al.* (2015)

As a second example for mixed effects transformation models for ordinal outcomes, we reproduce the estimation results by Seibold *et al.* (2015) using `tramm`.

The authors model the extinction risk of 1025 saproxylic beetle species using the German Red List extinction risk status as an ordinal outcome, and include various covariates describing the species' biology, required resources, and distributions as fixed effects, along with phylogenetically *correlated* random effects in a proportional odds logistic regression framework.

The model described in (4.5) needs to be sightly changed in order to accommodate correlated random effects. The modified specification for this setting can be written in its vectorized form as

$$\mathbb{P}(\mathbf{Y} \leq \boldsymbol{y}_k \mid \mathbf{X}, \widetilde{\boldsymbol{\Gamma}} = \widetilde{\boldsymbol{\gamma}}) = F_{SL}\left(\boldsymbol{\vartheta}_k + \mathbf{X}\boldsymbol{\beta} + \sigma\mathbf{U}\mathbf{R}\widetilde{\boldsymbol{\gamma}}\right), \tag{4.6}$$
$$\widetilde{\boldsymbol{\Gamma}} \sim \mathcal{N}_{1025}\left(\mathbf{0}, \mathbf{I}_{1025}\right),$$

where $\mathbf{U}$ denotes the 1025×1025 design matrix of random effects, $\mathbf{R}$ is the 1025×1025 lower triangular Cholesky factor of the *fixed* phylogenetic correlation matrix, σ is the (scalar) standard deviation of the random intercepts, $\widetilde{\boldsymbol{\Gamma}}$ is a column vector of independent standard normal random variables, and F_{SL} indicates the cumulative distribution function of the standard logistic. The vector of random intercepts, $\mathbf{\Gamma} = \sigma \mathbf{R} \widetilde{\mathbf{\Gamma}}$, will then be correlated with a correlation matrix $\mathbf{R}\mathbf{R}^\top$ and variance σ^2.

Seibold *et al.* (2015) estimate the regression model in (4.6) by modifying the `clmm` function of the `ordinal` package to include correlated random effects. The procedure is described in detail in Seibold *et al.* (2015, Appendix 2).

Appendix A.2 presents the code that modifies the basic functionality of the `tramm` package to incorporate correlated random effects with a fixed correlation matrix. Table 4.9 shows the results of the estimation. The parameter estimates and the standard errors as well as the value of the log-likelihood function at its maximum are essentially the same as the results of the article. The profile confidence interval of the standard deviation of the random effect confirms one of the main findings of the analysis that polygenetic information introduces a significant source of variability in the extinction risk status of the various species.

Table 4.9: Estimation results from the transformation model to reproduce the results by Seibold *et al.* (2015). The parameter σ denotes the standard deviation of the random effects.

	coef	SE
mean body size	0.0689	0.0209
mean elev	-0.4847	0.1888
tree = l	0.6525	0.3237
tree = n	-1.4840	0.421
flowers	-0.4247	0.2811
feeding = d	0.8069	0.7652
feeding = m	-0.2064	0.4869
feeding = x	0.5682	0.3723
habitat = wood and bark	-0.7519	0.4163
habitat = fungi	-0.0560	0.5555
niche = decay	-0.0750	0.1723
niche = diam	1.1030	0.1649
niche = canopy	-0.9371	0.2815
distribution	-0.3656	0.0252
	param	95% CI
$\widehat{\sigma}$	1.4034	from 0.94 to 1.96
Log-likelihood	-837.1604	

4.5 Mixed Models for Time-to-Event Data

Various types of regression models for time-to-event outcomes can also be expressed in the transformation model framework. By setting F_Z to the minimum extreme value distribution, we get models that satisfy the assumption of *proportional hazards*. Ignoring the random effects and the indices for the sake of brevity, and without the loss of generality, the conditional cumulative distribution function in this case is

$$\begin{aligned} F_Y(y \mid \boldsymbol{x}) &= F_{MEV}\left(h_Y(y) + \boldsymbol{x}^\top \boldsymbol{\beta}\right) \\ &= 1 - \exp\left(-\exp(h_Y(y) + \boldsymbol{x}^\top \boldsymbol{\beta})\right), \end{aligned}$$

and thus the conditional survivor function, probability density function, and the hazard function become

$$
\begin{aligned}
S_Y(y \mid \boldsymbol{x}) &= \exp\left(-\exp(h_Y(y) + \boldsymbol{x}^\top \boldsymbol{\beta})\right), \\
f_Y(y \mid \boldsymbol{x}) &= \frac{d}{dy} F_Y(y \mid \boldsymbol{x}) \\
&= -\exp\left(-\exp(h_Y(y) + \boldsymbol{x}^\top \boldsymbol{\beta})\right)\left(-\exp(h_Y(y) + \boldsymbol{x}^\top \boldsymbol{\beta})\right) h'_Y(y), \\
\lambda_Y(y \mid \boldsymbol{x}) &= \frac{f_Y(y \mid \boldsymbol{x})}{S_Y(y \mid \boldsymbol{x})} \\
&= \exp(h_Y(y)) h'_Y(y) \exp(\boldsymbol{x}^\top \boldsymbol{\beta}), \qquad (4.7)
\end{aligned}
$$

respectively. From (4.7) it can be seen, that the changes in the covariates affect the conditional hazard proportionally. The baseline hazard is $\lambda_Y(y) = \exp(h_Y(y))h'_Y(y)$ hence the baseline transformation function, $h_Y(y)$, denotes the log of the cumulative baseline hazard.

By capturing $h_Y(y)$ with a general, smooth function using the Bernstein polynomials, we arrive at the fully parameterized version of the Cox proportional hazards model:

$$
\begin{aligned}
\mathbb{P}(Y_{ij} \le y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_i = \boldsymbol{\gamma}_i) &= F_{MEV}\left(h_Y(y_{ij}) + {\boldsymbol{x}_{ij}}^\top \boldsymbol{\beta} + \boldsymbol{u}_{ij}\boldsymbol{\gamma}_i\right) \\
&= 1 - \exp\left(-\exp(\boldsymbol{a}_{\mathrm{Bs,k}}(y_{ij})^\top \boldsymbol{\vartheta} + {\boldsymbol{x}_{ij}}^\top \boldsymbol{\beta} + {\boldsymbol{u}_{ij}}^\top \boldsymbol{\gamma}_i)\right). \qquad (4.8)
\end{aligned}
$$

Note the slight difference between the model in Equation (4.8), which gives a fully parametric description of the cumulative density function of Y, and the “classical” Cox model, which leaves the baseline function, $h_Y(y)$, unspecified and estimates the parameters of interest using the partial likelihood approach.

4.5.1 Comparison with the `coxme` Package

The `R` package `coxme` by Therneau (2018) implements the Cox proportional hazards model with Gaussian random effects terms. Its approach is very similar in nature to the transformation model presented by (4.8), except the fact that it estimates the model parameters by integrating (using Laplace approximation) and maximizing the *partial likelihood*, and leaving the baseline hazards unspecified. The parameter estimates from the two implementations should be close to each other, nevertheless.

The `eortc` dataset in the `coxme` package is a simulated example based on the outcomes of breast cancer trial by the European Organization for Research and Treatment of Cancer. It contains observations from 37 enrolling centers, two treatment arms, and a total of 2323 data points. All the observations in the dataset are either right censored or exactly indicating the time of death.

We first estimate Cox proportional hazards models that include random intercepts for the various centers.

```
library("coxme")
eortc$trt <- factor(eortc$trt)
eortc$center <- factor(eortc$center)
fit_coxme1  <- coxme(Surv(y, uncens) ~ trt + (1 | center), data = eortc)
fit_tramm1 <- CoxphME(Surv(y, uncens) ~ trt + (1 | center), data = eortc,
                log_first = TRUE, order = 10)
```

Note the two extra arguments in the `Coxph` function call: `log_first = TRUE` tells `tramm` to take the logarithm of the dependent variable before applying the Bernstein polynomials. Setting this option usually results in better model fits in the case of time-to-event data. The order of the Bernstein polynomials, which are used to fit the baseline transformation, is set to a higher value with the option `order=10` to increase model flexibility.

The estimation results of the two approaches are presented in Table 4.10. As the results show, we get very similar estimates for the model parameters in this example. Since the objective function of the estimation is different in the two approaches — `coxme` uses the partial likelihood, while `tramm` maximizes the (log-)likelihood — model comparison based on the value of the objective function at its maximum is not meaningful. For this reason we omit the log-likelihood values from the table.

Table 4.10: Comparison of parameter estimates from `coxme` and `tramm` on the `eortc` dataset. The parameter σ denotes the standard deviation of the random intercept.

	`coxme`		`tramm`	
	coef	SE	coef	SE
trt = 1	0.7086	0.0642	0.7090	0.0643
$\widehat{\sigma}$	0.3292		0.3238	

An advantage of the fully parametric approach of `tramm` is that we can estimate the baseline transformations (i.e. the log-cumulative hazards) — given the vector of random effects — easily. To check whether the proportional hazards assumption holds for the treatment effect, we re-estimate the model stratifying for the treatment indicator and compare the baseline transfromations for the two strata.

```
fit_tramm1b <- CoxphME(Surv(y, uncens) | 0 + trt ~ 1 + (1 | center),
                       data = eortc, log_first = TRUE, order = 10)
```

As it can be seen in Figure 4.4, the two (conditional) log-cumulative hazards are parallel, i.e. the treatment effect is not dependent on survival time. When the mixed effects Cox model is estimated using the partial likelihood approach, as in the case of `coxme`, where the baseline hazard function is unspecified, this kind of visual comparison of the cumulative hazard functions is conceptually more complicated and currently not readily feasible in the `coxme` package.

A possible extension random effects model considered in this subsection adds nested random intercepts for the treatments within each center. This way, center-level heterogeneity of treatment effects can be introduced in the model. Nested random effects can be formulated by the operator `(1 | center/trt)` both in `coxme` and `tramm`.

```
fit_coxme2  <- coxme(Surv(y, uncens) ~ trt + (1 | center/trt), data = eortc)
fit_tramm2 <- CoxphME(Surv(y, uncens) ~ trt + (1 | center/trt), data = eortc,
              log_first = TRUE, order = 10)
```

We find slightly larger differences in the parameter estimates in the case of the extended models than previously (Table 4.11), which can be attributed to the increased model complexity.

Figure 4.5 shows the log-cumulative hazard, i.e. the baseline transformation, of the nested random effects model plotted against *log-time*. The linear relationship between these two measures suggests that the conditional distribution of the event times is very close to the Weibull distribution. Since the `eortc` dataset is a simulated dataset, Figure 4.5 indicates that the data may have been simulated from the Weibull distribution.

The Weibull regression can be written in its mixed transfromation model form as

$$\begin{aligned}\mathbb{P}(Y_{ij} \leq y_{ij} \mid \mathbf{X}_{ij} = \boldsymbol{x}_{ij}, \boldsymbol{\Gamma}_j = \boldsymbol{\gamma}_i) &= F_{MEV}\left(h_Y(y_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}\boldsymbol{\gamma}_i\right) \\ &= 1 - \exp\left(-\exp(\vartheta_1 + \vartheta_2 \log(y_{ij}) - \boldsymbol{x}_{ij}{}^\top \boldsymbol{\beta} - \boldsymbol{u}_{ij}{}^\top \boldsymbol{\gamma}_i)\right). \quad (4.9)\end{aligned}$$

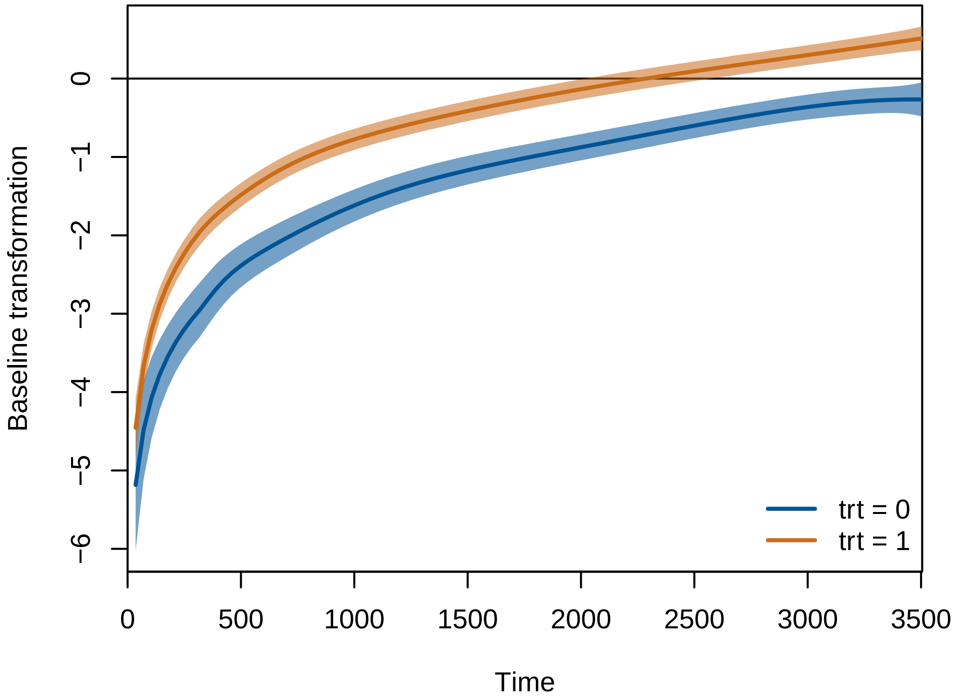

Figure 4.4: Log-cumulative baseline hazard functions in the stratified model of the `eortc` data. The solid lines denote the point estimates and the areas the 95% point-wise confidence intervals.

Table 4.11: Comparison of parameter estimates from `coxme` and `tramm` on the `eortc` dataset. The parameters σ_1 and σ_2 denote the standard deviations of the random center effect and the nested random treatment effect, respectively.

	`coxme`		`tramm`	
	coef	SE	coef	SE
trt = 1	0.7420	0.0827	0.7538	0.0852
$\widehat{\sigma}_1$	0.2627		0.2541	
$\widehat{\sigma}_2$	0.2045		0.2084	

Using the `SurvregME` function of the `tramm` package, we can fit the mixed effects Weibull regression to the `eortc` dataset, including random intercepts to account for center-level correlations in the data, and nested treatment intercepts to allow for variability in the treatment effects across enrolling centers.

```
fit_weibull <- SurvregME(Surv(y, uncens) ~ trt + (1 | center/trt), data = eortc,
                         dist = "weibull")
```

Table 4.12 compares the parameter estimates as well as the values of the likelihood function of the Weibull regression and the fully parameterized version of the Cox proportional hazards model. Note that the opposite signs of the treatment effects are due to the fact that (4.9) is parameterized with negative signs of fixed and random effects. The parameter estimates and standard errors are very similar in the two models, while the log-likelihood value is higher in the case of the Weibull regression.

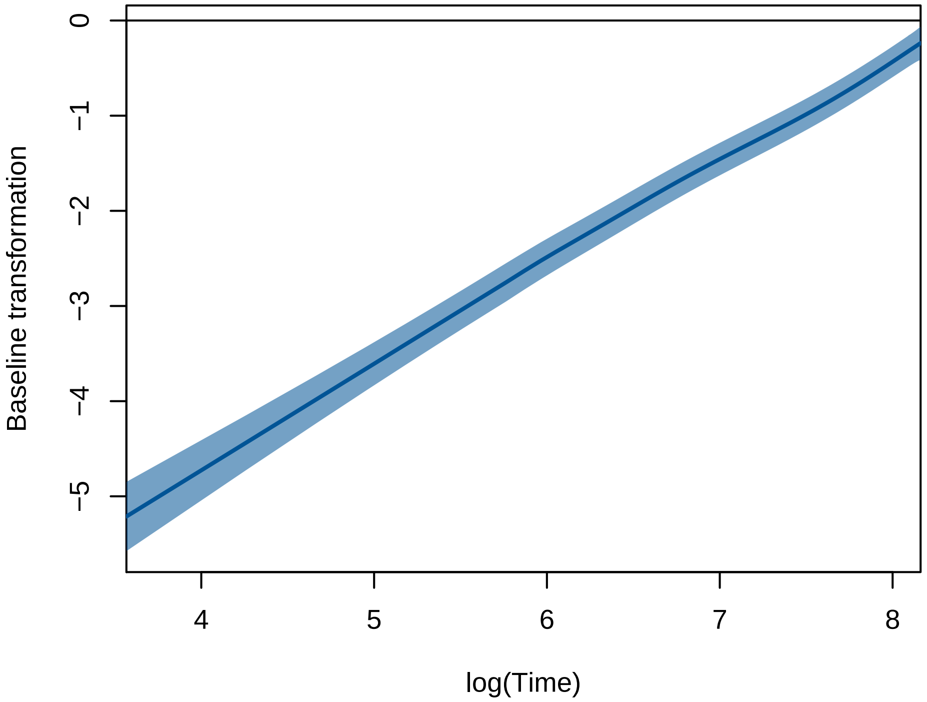

Figure 4.5: Log-cumulative baseline hazard function plotted against log-time in the nested random intercepts model of the `eortc` data. The solid line denotes the point estimates and the area the 95% point-wise confidence interval.

Table 4.12: Comparison of estimates from the fully parameterized mixed effects Cox proportional hazards model (`tramm::CoxphME`) and the mixed effects Weibull regression model (`tramm::SurvregME`) on the `eortc` dataset. The parameters σ_1 and σ_2 denote the standard deviations of the random center effect and the nested random treatment effect, respectively.

	`CoxphME`		`SurvregME`	
	coef	SE	coef	SE
trt = 1	0.7538	0.0852	-0.7531	0.0851
$\widehat{\sigma}_1$	0.2541		0.2549	
$\widehat{\sigma}_2$	0.2084		0.2079	
Log-likelihood	-13243.1237		-13032.0305	

4.5.2 Comparison with the `parfm` Package

The `parfm` package by Munda *et al.* (2012) implements parametric frailty models for survival data. It provides several parametric hazard specifications and frailty distributions, but it only allows a single grouping factor to define the frailty terms, i.e. only a random intercept model can be specified within the framework. As a result, the `parfm` package is only partially comparable with our `SurvregME` function. The mixed effects Weibull regression with normally distributed random intercepts is a specification that can be estimated with both implementations, and hence it will serve as the basis of comparison in this subsection. As a special case of the mixed effects Weibull regression, the model is given by 4.9 with ${\boldsymbol{u}_{ij}}^\top \boldsymbol{\gamma}_i = \gamma_i$.

The `kidney` dataset is an example dataset in the `survival` package. It contains 76 observations, two for each of the 38 patients with portable dialysis equipment. The outcome variable measures the recurrence time of kidney infection, and some of the observations are right censored. The age and gender of the patients as well as the disease type are also included in the dataset.

We can estimate a Weibull regression with random intercepts for patients, and age and sex as fixed effects in `parfm` and `tramm` as

```
library("parfm")
data("kidney", package = "survival")
kidney$sex <- factor(kidney$sex, levels = c(1, 2), labels = c("male", "female"))

fit_parfm <- parfm(Surv(time, status) ~ sex + age, cluster = "id",
                   data = kidney, dist = "weibull", frailty = "lognormal")
fit_tramm <- SurvregME(Surv(time, status) ~ sex + age + (1 | id),
                       data = kidney, dist = "weibull")
```

Note that the log-normal frailty term in `parfm` means an identical specification to a Weibull regression in `tramm` with normally distributed random intercepts. Since `parfm` also uses Laplace approximation to evaluate the marginal likelihood, the results from the two packages should be numerically comparable. Table 4.13 summarizes the coefficient estimates, their standard errors, the standard deviations of the random effects, and the log-likelihood values. As the results show, the two functions give very similar estimates for the Weibull mixed regression model.

Table 4.13: Comparison of the estimates from the `parfm` and `tramm` packages on the `kidney` dataset. The parameter σ stands for the standard deviation of the random effect term.

	`parfm`		`tramm`	
	coef	SE	coef	SE
sex = female	1.6264	0.4884	1.6264	0.4881
age	-0.0060	0.0114	-0.0060	0.0126
$\widehat{\sigma}$	0.7677		0.7669	
Log-likelihood	-332.8629		-332.8629	

The examples presented in this section included either exactly observed or right-censored event times. Both `coxme` and `parfm` can only deal with these two types of observations. In practice, many time-to-event datasets contain interval-censored event times, since in many cases ascertaining the status of the subjects is only possible at some fixed time points. This problem can, of course, be present in the case of other regression settings, too, but it is most pronounced in survival modeling. One of the main advantages of using the transformation model approach is that estimation for any form of (independent) censoring or truncation is available without any additional computational costs.

Chapter 5

Discussion

The model introduced in Chapter 2 builds on the transformation model approach by Hothorn *et al.* (2018), and extends the framework to incorporate mixed effects. This way the model class becomes applicable to grouped data, where the correlation between observations in the same group should be taken into account in order to do valid inference.

Chapter 3 described an approach to implement this model class in `R`. First, it summarized the main components that are used to create a package (`tramm`) for estimating transformation mixed models: The packages `tram` and `lme4` (by Hothorn (2019) and Bates *et al.* (2015), respectively) can be utilized to define stratified linear transformation models with mixed effects. The functionality of the `mlt` package by Hothorn (2018) is then applied to generate the design matrices corresponding to the baseline transformation and the fixed effects. The estimation of the resulting non-linear mixed model is done by the Template Model Builder (`TMB`) by Kristensen *et al.* (2016), which utilizes Laplace approximation to marginalize the log-likelihood over the random effects, and automatic differentiation to return numeric values of the gradients of the objective function.

In Chapter 4, with the help of several examples, we demonstrated the main features of the implementation, and compared the results to those from available alternative `R` packages. Some of these alternative implementations were directly comparable to our approach (e.g. `lme4` in the case of the normal linear mixed model, or `ordinal` in the case of proportional odds logistic mixed model), while in other cases the alternatives were very similar to our modeling approach (`coxme` and the fully parametric Cox proportional hazards mixed model, or the `ordinalCont` package and our continuous outcome logistic mixed model). In the examples, we considered several datasets and model structures, and saw that the numerical results were reasonably close to the ones calculated with the alternative packages.

As the results of the comparisons suggest, the transformation mixed model approach is a viable general framework for a range of regression models for grouped data. Moreover, compared to its alternatives, it has some additional advantages: Since the model directly parameterizes the conditional cumulative distribution of the response, censored (assuming an independent censoring mechanism) and truncated observations can be easily accommodated, without additional computational costs. Furthermore, the model is fully parametric, and as a result, inference, simulation, and prediction is relatively straightforward. Finally, as a general modeling approach for unconditional and conditional analysis, the class of transformation models can be easily extended to new problems by carefully selecting the error distribution and the baseline transformation function.

5.1 Limitations

The mixed effects model considered in this thesis is not the only conceivable extension of transformation models with random effects. Although the additive structure of the model is computationally as well as conceptually convenient, it is also restrictive. The random effects in this model only act as shifting factors on the conditional distribution of the outcome, and do not affect its higher moments. Further generalizations of the model class is left for future research.

Using the Template Model Builder for the computationally intensive calculations leads to great efficiency gains. However, one disadvantage of the approach is that `TMB` uses the Laplace approximation to marginalize over the random effects. In some cases, e.g. with discrete outcomes or small group sizes, this method can lead to numerical inaccuracies (see for example Joe, 2008). Although `TMB` offers access to some more advanced numerical integration methods, due to the loss in computational efficiency and increased complexity of the code, we have not included these in our implementation.

The `tramm` package in under ongoing development. In its current state, the package is lacking some important features for presenting the results, checking the model specifications, creating predictions, and doing simulations. Moreover, only asymptotic methods are currently available for inference.

5.2 Future Work

In this thesis, we have shown that our implementation of the transformation mixed model class gives the same estimates (i.e. the differences are within reasonably low tolerance levels) as other well established and tested packages for certain specific cases. Other examples did not allow for direct comparisons, as the alternative implementations differed slightly from the transformation model approach. As the results of Chapter 4 suggest, even in these cases, we get similar estimates. Carefully designed simulations could help to compare our implementation to these alternatives under various conditions and data structures. Moreover, simulation studies could also help to uncover efficiency gains from using the `TMB` in the case of various data structures and model complexities.

Currently, the `tramm` package lacks some important functionality that should be added to make it suitable for rigorous analyses. Simulation capabilities with efficient re-estimation methods would provide a basis for parametric bootstrap inference, similarly to the tools available in the `mlt` package by Hothorn (2018) or in `lme4` by Bates *et al.* (2015). In many cases, the goal of the statistical modeling is to calculate predictions rather than making inference based on an estimated model. Methods for point predictions and corresponding intervals would make a very useful extension of the current implementation. Moreover, additional methods for presenting the results and checking the model specifications would also be necessary to create a fully functional package.

The restricted maximum likelihood (REML) estimator provides unbiased estimates of the variance components in the linear model. With a slight modification of the current implementation we can calculate REML estimates for this subclass of mixed effects transformation models. Additionally, the concept could be extended to the general case of transformation mixed models, but the benefits of it are not clear and should be examined beforehand.

The accuracy of the Laplace approximation should be evaluated in various model settings. When necessary, other numerical techniques (e.g. adaptive Gaussian quadrature methods, importance sampling, MCMC) should also be considered in order to improve model estimations. Due to the broad range of possible model settings, the most suitable method might depend on the model complexity or the type of the response variable.

Other possible direction for extending our mixed effects model is to consider non-normal random effects distributions. In some specific cases non-normal random effects could be meaningful, such as Gamma-distributed frailty terms in proportional hazards models. Furthermore, complex dependence structures among random effects, e.g. defined by time series processes or spatial structures, play an important role in many fields of applied statistics. As a notable example, Brooks *et al.* (2017) provide an extension to generalized linear mixed models to incorporate such random effect structures in their `glmmTMB` package. Allowing for complex covariance structures would make transformation models applicable in many interesting problems.

Finally, as we discussed it before, the additive structure of the transformation mixed models considered in this thesis is rather restrictive. Extension to more general cases, where the random effects also affect the higher moments of the conditional distribution of the response variable, should be the topic of future research.

5.3 Conclusions

In this thesis, we proposed a possible extension of the transformation models, as described by Hothorn *et al.* (2018), to include mixed effects terms. With the resulting specification, the transformation model approach can be applied on grouped data, where the observations within groups are correlated. Using the Template Model Builder, we provided a feasible way of implementing this class of non-linear mixed models in `R`. The resulting code is organized into the `tramm` package, which also builds on the packages `mlt` and `tram`, as well as `lme4`. As a proof of concept, we presented a series of examples for a broad range of regression problems, where specific versions of our mixed effects model were compared to available alternative implementations. The results suggest that our approach is a viable alternative of these packages. Moreover, the transformation mixed model approach provides some additional features that are currently not available elsewhere.

Appendix A

Computer Code

A.1 `C++` Implementation of the Negative Log-likelihood

This code specifies the objective function for the `tramm` package in `C++`, which is then called by `TMB` to numerically evaluate the marginal log-likelihood, its gradient and the Hessian using Laplace approximation and automatic differentiation.

```
/*=================================================================
  Transformation model: F_Y(y|x,gamma) = F_Z(h(y|x,gamma))
  Structure of ME: h(y|x,gamma) = (a(y)', b(x)') * beta + Z * gamma
  =================================================================
 */

#include <TMB.hpp>

// Model types
enum valid_modtype {
    Lm = 0
};

// Valid error distributions
enum valid_errdist {
    Normal = 0, Logistic = 1, MinExtrVal = 2, MaxExtrVal = 3
};

// F_Z(h(y)), cens: censoring indicator, if not finite, return value
template <class Type>
Type cdf(Type hy, Type cens, Type value, int errdist) {
  Type out = value;
  if (fabs(cens) < INFINITY)  {
    switch (errdist) {
      case Normal:
        out = pnorm(hy);
        break;
      case Logistic:
        out = Type(1) / (Type(1) + exp(-hy));
        break;
      case MinExtrVal:
        out = Type(1) - exp(-exp(hy));
        break;
      case MaxExtrVal:
        out = exp(-exp(-hy));
        break;
      default:
        error("Unknown error distribution!");
    }
  }
  return out;
}

// log-density error function
template <class Type>
Type ldens(Type hy, int errdist) {
  Type out;
  switch (errdist) {
    case Normal:
      out = dnorm(hy, Type(0), Type(1), true);
      break;
    case Logistic:
      out = dlogis(hy, Type(0), Type(1), true);
      break;
    case MinExtrVal:
      out = hy - exp(hy);
      break;
    case MaxExtrVal:
      out = - hy - exp(-hy);
      break;
    default:
      error("Unknown error distribution!");
  }
  return out;
}

GVECTORIZE(ldens, V, I, N, N, N, N);

// Covariance terms of the random effects
template <class Type>
struct re_cov_term {
  vector<Type> sd;
  matrix<Type> corr;
};

// Negative log-density of the random effects
template <class Type> Type re_nldens(vector<Type> gamma,
  vector<Type> theta, int blocksize, re_cov_term<Type>& term) {
  Type ans = 0;
  if (blocksize == 1) { // diagonal cov matrix of the term
    //Type sd = theta[0];
    Type sd = exp(theta[0]); // parateterized w/ log(sd)
    ans -= dnorm(gamma, Type(0), sd, true).sum();
    //term.sd = theta;
    term.sd = exp(theta);
    matrix<Type> corr(1,1);
    term.corr = corr.setIdentity(1,1);
  } else { // correlated random effects (unstructured corr mat)
    int nblocks = gamma.size() / blocksize;
    vector<Type> sd = exp(theta.head(blocksize));
    vector<Type> corr_tr = theta.tail(theta.size() - blocksize);
    density::UNSTRUCTURED_CORR_t<Type> nldens(corr_tr);
    density::VECSCALE_t<density::UNSTRUCTURED_CORR_t<Type> >
      scnldens = density::VECSCALE(nldens, sd);
    for (int i = 0; i < nblocks; i++) {
      ans += scnldens(gamma.segment(i * blocksize, blocksize));
    }
    term.sd = sd;
    term.corr = nldens.cov();
  }
  return ans;
}

template<class Type>
Type objective_function<Type>::operator() ()
{
  DATA_INTEGER(modtype); // model type for outputs
  DATA_INTEGER(errdist); // Type of error distribution
  DATA_MATRIX(MMl); // left-hand side model matrix (a(y), b(x))
  DATA_MATRIX(MMr); // right-hand side model matrix
  DATA_MATRIX(MMe); // model matrix for exact observations
  DATA_MATRIX(MMeprime); // for constructing h(y)'
  DATA_VECTOR(offsetc); // offset for censored (same for l and r)
  DATA_VECTOR(offsete); // offset for exact
  DATA_VECTOR(weightsc);
  DATA_VECTOR(weightse);
  DATA_SPARSE_MATRIX(Zc); // design matrix of random effects -- censored
  DATA_SPARSE_MATRIX(Ze); // design matrix of random effects -- exact
  DATA_VECTOR(cl); // censoring indicators:
  DATA_VECTOR(cr); // cl == -Inf: left cens, cr == Inf: right cens
  DATA_IVECTOR(re_termsize); // number of REs corresponding one term
  DATA_IVECTOR(re_blocksize); // in the cov matrix of REs

  PARAMETER_VECTOR(beta); // fixed effects
  PARAMETER_VECTOR(gamma); // random effects
  PARAMETER_VECTOR(theta); // covariance parameters

  parallel_accumulator<Type> nll(this); // negative log-likelihood

  vector<re_cov_term<Type> > re_cov(re_termsize.size());

  // ====== Likelihood contributions of RE terms
  int cum_ts = 0;
  int cum_bs = 0;
  for (int i = 0; i < re_termsize.size(); i++) {
    int nth = re_blocksize(i) * (re_blocksize(i)+1) / 2; // cov params
    vector<Type> gamma_s = gamma.segment(cum_ts, re_termsize(i));
    vector<Type> theta_s = theta.segment(cum_bs, nth);
    nll += re_nldens(gamma_s, theta_s, re_blocksize(i), re_cov(i));
    cum_ts += re_termsize(i);
    cum_bs += nth;
  }

  // ====== Likelihood contributions of observations
  // === Censored observations
  if (MMl.rows() > 0) {
    vector<Type> hl = MMl * beta + Zc * gamma + offsetc;
    vector<Type> hr = MMr * beta + Zc * gamma + offsetc;
    for (int i = 0; i < MMl.rows(); i++) {
      nll -= log(cdf(hr(i), cr(i), Type(1), errdist) -
        cdf(hl(i), cl(i), Type(0), errdist)) * weightsc(i);
    }
  }
  // === Exact observations
  if (MMe.rows() > 0) {
    vector<Type> he = MMe * beta + Ze * gamma + offsete;
    vector<Type> hprime = MMeprime * beta;
    nll -= ((ldens(he, errdist) + log(hprime)) * weightse).sum();
  }

  // ====== Reporting SD and correlation matrices of random effects
  vector<matrix<Type> > corr_rep(re_cov.size());
  vector<vector<Type> > sd_rep(re_cov.size());
  for(int i = 0; i < re_cov.size(); i++) {
      corr_rep(i) = re_cov(i).corr;
      sd_rep(i) = re_cov(i).sd;
  }
  REPORT(corr_rep);
  REPORT(sd_rep);

  // ====== Report transformed paremeters for specific model types
  if (modtype == Lm) {
    int p = beta.size() - 1;
    Type sigma = 1 / beta[1];
    vector<Type> b(p);
    b(0) = -beta(0) * sigma;
    b.tail(p-1) = beta.tail(p-1) * sigma;
    ADREPORT(b);
    ADREPORT(sigma);
  }

  return nll;
}
```

A.2 Code for Reproducing the Red List Example

This code reproduces the model estimation of Seibold *et al.* (2015) using the functionality of the `tramm` package.

```
if (!file.exists("results/Seibold2015.rda")) {
  library("tramm")
  library("Matrix")

  ### publication: https://doi.org/10.1111/cobi.12427
  dir <- system.file("rda", package = "TH.data")
  load(file.path(dir, "beetles.rda"))

  ldata <- bmodel
  ldata$TS <- NULL

  phylo_corr <- solve(Ainv)

  ## Model specification: fixed effects only
  mfr <- RL ~ mean_body_size + mean_elev + tree + flowers + feeding +
    habitat + niche_decay + niche_diam + niche_canopy + distribution
  fit_ind <- Polr(mfr, data = ldata)
  ## load dlls for custom use of tramm functionality
  tr <- system.file("libs", package = "tramm")
  dyn.load(TMB::dynlib(paste0(tr, "/tramm")))

  ## Set up the ME model manually
  fe_data <- tramm:::get_data_tram(fit_ind)
  ## --- Constraints
  ui <- as.matrix(bdiag(fe_data$constr$ui, 1))
  ci <- c(fe_data$constr$ci, -Inf)

  ## --- Mixed-effects model specification
  mfr_me <- update(mfr, . ~ . + (1 | species))
  rt <- tramm::get_re_struct(mfr_me, ldata, na.action = na.omit, negative = TRUE)
  R <- as(t(chol(phylo_corr)), class(rt$Z))
  Zc <- rt$Z %*% R
  datalist <- list(modtype = 5L, errdist = 1L, cl = fe_data$cl, cr = fe_data$cr,
                   MMl = fe_data$MMl, MMr = fe_data$MMr,
                   MMe = fe_data$MMe, MMeprime = fe_data$MMeprime,
                   Zc = Zc,
                   Ze = Matrix(0, nrow = 0, ncol = 0, sparse = TRUE),
                   re_termsize = rt$re_termsize, re_blocksize = rt$re_blocksize,
                   offsetc = fe_data$offsetc, offsete = fe_data$offsete,
                   weightsc = fe_data$weightsc, weightse = fe_data$weightse)

  ## --- Model
  model_structure <- list(formula = mfr_me,
    restrictions = list(ui = ui, ci = ci),
    re = rt, fe = fe_data, tram_model = fit_ind)
  params <- list(beta = fe_data$beta, gamma = rep(0, nrow(ldata)), theta = 0)
  obj <- TMB::MakeADFun(data = datalist, parameters = params, random = "gamma",
                        DLL = "tramm", silent = TRUE)
  ## --- Optimization
  opt <- alabama::auglag(par = obj$par, fn = obj$fn, gr = obj$gr,
      hin = function(par) ui %*% par - ci,
      hin.jac = function(par) ui,
      control.outer = list(method = "nlminb", kkt2.check = FALSE, trace = FALSE))
  rep <- TMB::sdreport(obj)
  tmb_mod <- list(tmb_object = obj, data = datalist,
    model_structure = model_structure, opt = opt, sdrep = rep)
  class(tmb_mod) <- "tramm"
  ci_prof <- exp(TMB::tmbroot(tmb_mod$tmb_obj, length(tmb_mod$sdrep$par.fixed),
    target = 0.5 * qchisq(0.95, df = 1), trace = FALSE))
  save(tmb_mod, ci_prof, file = "results/Seibold2015.rda")
} else {
  load("results/Seibold2015.rda")
}
```

Bibliography

Bates, D., Mächler, M., Bolker, B., and Walker, S. (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, **67**, 1–48. 13, 18, 33, 34

Box, G. E. P. and Cox, D. R. (1964). An analysis of transformations. *Journal of the Royal Statistical Society. Series B (Methodological)*, **26**, 211–252. 3, 20

Brooks, M. E., Kristensen, K., van Benthem, K. J., Magnusson, A., Berg, C. W., Nielsen, A., Skaug, H. J., Maechler, M., and Bolker, B. M. (2017). glmmTMB balances speed and flexibility among packages for zero-inflated generalized linear mixed modeling. *The R Journal*, **9**, 378–400. 11, 34

Christensen, R. H. B. (2019). ordinal—regression models for ordinal data. R package version 2019.3-9. http://www.cran.r-project.org/package=ordinal/. 24

Demidenko, E. (2004). *Mixed models: Theory and applications*. Wiley Series in Probability and Statistics. Wiley-Interscience, New York, NY, USA. 4

Griewank, A. and Walther, A. (2008). *Evaluating derivatives: Principles and techniques of algorithmic differentiation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second edition. 11

Held, L. and Bové, D. S. (2014). *Applied statistical inference: Likelihood and Bayes*. Springer, Heidelberg. 18

Hothorn, T. (2018). Most likely transformations: The mlt package. *Journal of Statistical Software*, , "Accepted: 2018–03–05". 4, 9, 33, 34

Hothorn, T. (2019). Transformation models: The tram package. R package vignette version 0.2-5. https://CRAN.R-project.org/package=tram. 4, 9, 33

Hothorn, T., Kneib, T., and Bühlmann, P. (2014). Conditional transformation models. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **76**, 3–27. 3

Hothorn, T., Möst, L., and Bühlmann, P. (2018). Most likely transformations. *Scandinavian Journal of Statistics*, **45**, 110–134. i, 1, 3, 4, 5, 9, 22, 33, 35

Joe, H. (2008). Accuracy of Laplace approximation for discrete response mixed models. *Computational Statistics and Data Analysis*, **52**, 5066–5074. 7, 34

Kristensen, K., Nielsen, A., Berg, C. W., Skaug, H., and Bell, B. M. (2016). TMB: Automatic differentiation and Laplace approximation. *Journal of Statistical Software*, **70**, 1–21. i, 1, 10, 33

Lohse, T., Rohrmann, S., Faeh, D., and Hothorn, T. (2017). Continuous outcome logistic regression for analyzing body mass index distributions. *F1000Research*, **6**, 1933. 22

Manuguerra, M. and Heller, G. (2010). Ordinal regression models for continuous scales. *The International Journal of Biostatistics*, **6**, 14–14. 22

Manuguerra, M., Heller, G., and Ma, J. (2017). Semi-parametric ordinal regression models for continuous scales. In Grzegorczyk, M. and Ceoldo, G., editors, *Proceedings of the 32nd International Workshop on Statistical Modelling*, volume 1, 236–241. 22

McCulloch, C. and Searle, S. (2001). *Generalized, linear, and mixed models*. Wiley Series in Probability and Statistics. John Wiley & Sons. 5

Muff, S., Held, L., and Keller, L. F. (2016). Marginal or conditional regression models for correlated non-normal data? *Methods in Ecology and Evolution*, **7**, 1514–1524. 5

Munda, M., Rotolo, F., and Legrand, C. (2012). parfm: Parametric frailty models in R. *Journal of Statistical Software*, **51**, 1–20. 31

Pinheiro, J., Bates, D., DebRoy, S., Sarkar, D., and R Core Team (2018). *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-137. 13

Pinheiro, J. C. and Bates, D. M. (1995). Approximations to the log-likelihood function in the nonlinear mixed-effects model. *Journal of Computational and Graphical Statistics*, **4**, 12–35. 6

Pinheiro, J. C. and Chao, E. C. (2006). Efficient Laplacian and adaptive Gaussian quadrature algorithms for multilevel generalized linear mixed models. *Journal of Computational and Graphical Statistics*, **15**, 58–81. 7

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. 2, 13

Seibold, S., Brandl, R., Buse, J., Hothorn, T., Schmidl, J., Thorn, S., and Müller, J. (2015). Association of extinction risk of saproxylic beetles with ecological degradation of forests in europe. *Conservation Biology*, **29**, 382–390. 26, 27, 40

Therneau, T. M. (2018). *coxme: Mixed Effects Cox Models*. R package version 2.2-10. 28

Tutz, G. (2011). *Regression for categorical data*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press. 4, 24

Varadhan, R. (2015). *alabama: Constrained Nonlinear Optimization*. R package version 2015.3-1. 14