Journal of Statistical Software

December 2025, Volume 115, Issue 7. *doi: 10.18637/jss.v115.i07*

skewlmm: An R Package for Fitting Skewed and Heavy-Tailed Linear Mixed Models

Fernanda L. Schumacher
The Ohio State University

Larissa A. Matos
Universidade Estadual de Campinas

Victor H. Lachos
University of Connecticut

Abstract

Longitudinal data are commonly analyzed using linear mixed models, which, for mathematical convenience, usually assume that both random effect and error follow normal distributions. However, these restrictive assumptions may result in a lack of robustness against departures from the normal distribution and invalid statistical inferences. Schumacher, Lachos, and Matos (2021) developed a flexible extension of linear mixed models considering the scale mixture of skew-normal class of distributions from a frequentist point of view, accommodating skewness and heavy tails, and the robust model formulation accounts for a possible within-subject serial dependence by considering some useful dependence structures. This paper presents the R package **skewlmm**, which implements the method proposed by Schumacher *et al.* (2021) and provides a user-friendly tool to fit robust linear mixed models to longitudinal data, including model-fit tests, residual analyzes, and plot functions to support model selection and evaluation. Two data sets and a synthetic example are analyzed to illustrate the methodology and software implementation.

Keywords: longitudinal data analysis, outliers, robust models, skewness.

1. Introduction

Longitudinal data often appear in several areas, such as medicine, public health, and psychology, among others. Usually, a single measurement is collected repeatedly over time on each subject in the study, and the temporal ordering is important because measurements closer in time within a subject are more likely to be more similar than observations more distant in time (Weiss 2005).

Currently, there are two main popular functions to fit normal LMMs in R: the function `lme()` in the **nlme** package (Pinheiro, Bates, DebRoy, Sarkar, and R Core Team 2021), which sup-

ports several random effects and error level dependence structures, and the function `lmer()` in the **lme4** package (Bates, Mächler, Bolker, and Walker 2015), which is more efficient for fitting models with crossed random effects but does not support special dependence structures. Both these functions rely on the asymptotic distributions of the ML and REML estimators. Recently, a new package emerged with exact statistical results, the **glme** package (Cavus and Yazici 2025), which bases its tests and interval estimates on the generalized inference approach, as detailed in Weerahandi and Yu (2020). These inferences rely on the exact distributions of underlying statistics when the covariance structure is compound symmetric.

These functions assume normal distributions for the random terms (although the function `glmer()` in the **lme4** package fits generalized linear mixed-effect models), which is mathematically convenient but may result in a lack of robustness against departures from the normal distribution and invalid statistical inferences, especially when the data simultaneously show heavy tails and skewness (Drikvandi, Verbeke, and Molenberghs 2017).

Some proposals have been put forward in the literature to address this problem by replacing the normality assumption with a more flexible class of distributions. For instance, Pinheiro, Liu, and Wu (2001) proposed a multivariate t linear mixed model (T-LMM) and showed that it performed well in the presence of outliers. Accounting for skewness, Arellano-Valle, Bolfarine, and Lachos (2005) proposed a skew-normal linear mixed model (SN-LMM) based on the skew-normal (SN) distribution introduced by Azzalini and Dalla Valle (1996), and Ho and Lin (2010) proposed a skew-t linear mixed model (ST-LMM) based on the skew-t (ST) distribution introduced by Azzalini and Capitanio (2003). We note that in the context of Bayesian estimation, other proposals have been made in the literature (see, for example, Bandyopadhyay, Lachos, Abanto-Valle, and Ghosh 2010; Gong, Mao, Zhang, Ren, and Zuo 2023).

Regarding currently available R packages, the **lqmm** package (Geraci 2014) estimates linear quantile mixed-effect models, which allow for median-based estimation, considering Laplace random effects, but only enabling non-diagonal covariance structures for random effects under the normal assumption. Additionally, the function `heavyLme()` in the **heavy** package (Osorio 2019) fits linear mixed models under t distributions using the formulation described in Pinheiro *et al.* (2001), but it does not allow for within-subject correlation and simulation studies from Geraci and Farcomeni (2020) resulted in some unexpected estimates, indicating the possibility of bugs in the software.

Furthermore, the function `rlmer()` in the package **robustlmm** (Koller 2016) provides a robust version of the function `lmer()`, based on the random effect contamination model and the central contamination model, and allowing contamination to be detected at all levels of the data, but not considering within-subject dependence. Recently, Geraci and Farcomeni (2020) proposed a new family of linear mixed-effect models based on the generalized (symmetric) Laplace distribution and developed a maximum likelihood estimation approach based on Gaussian quadrature, which is implemented in the function `nlmm()` in the **nlmm** package (Geraci 2023). The model formulation allows the distribution of random effects and error to be different and for heteroscedasticity in the errors, but it also does not consider within-subject serial correlation.

All the packages mentioned above are restricted to symmetric distributions. Some flexible frameworks that handle multilevel models and include a variety of possible distributions, including the skew-normal distribution, are worth mentioning. The packages **gamlss**

(Stasinopoulos and Rigby 2007) and **bamlss** (Umlauf, Klein, Simon, and Zeileis 2021) fit complex generalized additive regression models for location, scale, and shape. They allow for modeling of all the parameters of the distribution as function of covariates and allowing for Gaussian random effects incorporation, while multiple distributions are available for modeling the conditional distribution of the response variable. The package **brms** (Bürkner 2017) fits Bayesian generalized linear and non-linear multilevel models using Stan and its syntax is based on the **lme4** package. These tools provide great flexibility and are useful for a wide range of applications, but given their wider focus, they lack on devoted tools for longitudinal data analysis and on flexible options for within-subject correlation.

Alternatively, the package **tramME** (Tamási and Hothorn 2021) allows for estimation and inference of mixed effects transformation models, incorporating Gaussian random effects for modeling a baseline transformation function of the response variable. Even though the availability of multiple error distributions and baseline transformations allows for a flexible model formulation, the interpretation of covariate effects is not straightforward for such a formulation. A recent proposal of Asar, Bolin, Diggle, and Wallin (2020), implemented in the package **ngme**, accommodates simultaneously heavy tails, skewness, and within-subject serial dependence considering three decoupled stochastic components following multivariate generalized hyperbolic distributions. The proposal is quite flexible, but it depends on estimating a more significant number of parameters. The R package is not yet available at CRAN, but a development version is available at `https://github.com/davidbolin/ngme2`.

On the other hand, our **skewlmm** package implements the methods proposed by Lachos, Ghosh, and Arellano-Valle (2010) and Schumacher *et al.* (2021), which considered robust parametric modeling of LMM based on a skewed and heavy-tailed class of distributions, called the scale mixture of skew-normal (SMSN), incorporating skewness at the random effects distribution and the latter allowing for within-subject serial dependence. This paper introduces the **skewlmm** package, which provides an efficient EM-type algorithm to compute maximum likelihood (ML) estimates of parameters of SMSN-LMMs, along with several tools for model selection and evaluation.

The rest of this paper is organized as follows. Section 2 introduces the proposed model and some important properties, along with the implemented dependence structure. Section 3 presents the algorithms considered in the package for maximum likelihood estimation. Section 4 shows the package's main functions and introduces the available tools for model evaluation, while Section 5 exemplifies the package's use in two real data set applications and a synthetic data setting. Finally, some final remarks are presented in Section 7.

2. Scale mixture of skew-normal linear mixed models

For completeness, we first briefly introduce the class of distributions considered throughout this work. Let $\mathbf{Y}$ be a $p\times 1$ random vector, $\boldsymbol{\mu}$ a $p\times 1$ location vector, $\boldsymbol{\Sigma}$ a $p\times p$ positive definite scale matrix, $\boldsymbol{\lambda}$ a $p\times 1$ shape parameter (which regulates the skewness), and let U be a positive random variable with a cumulative distribution function (cdf) $H(\cdot;\boldsymbol{\nu})$, where $\boldsymbol{\nu}$ is a scalar or parameter vector indexing the distribution of the mixing variable U. The multivariate SMSN class of distributions, denoted by $\text{SMSN}_p(\boldsymbol{\mu},\boldsymbol{\Sigma},\boldsymbol{\lambda};H)$, can be defined through the following density function:

$$f_{\mathbf{Y}}(\mathbf{y}) = 2\int_0^\infty \phi_p(\mathbf{y};\boldsymbol{\mu},\kappa(u)\boldsymbol{\Sigma})\Phi(\kappa(u)^{-1/2}\boldsymbol{\lambda}^\top\boldsymbol{\Sigma}^{-1/2}(\mathbf{y}-\boldsymbol{\mu}))\,dH(u;\boldsymbol{\nu}), \quad \mathbf{y}\in\mathbb{R}^p, \tag{1}$$

for some positive weight function $\kappa(u)$. Depending on the distribution of U, different distributions are attained. Letting $\kappa(u) = u^{-1}$, we consider explicitly the distributions described next, along with the distribution of the Mahalanobis distance $d = (\mathbf{y} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{y} - \boldsymbol{\mu})$.

- *The multivariate skew-normal distribution*, $\text{SN}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda})$, is obtained when $P(U = 1) = 1$. For this distribution, we have $d \sim \chi^2_p$.
- *The multivariate skew-t distribution* with ν degrees of freedom, $\text{ST}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, \nu)$ (Branco and Dey 2001; Azzalini and Genton 2008), which is derived from (1) by taking $U \sim \text{Gamma}(\nu/2, \nu/2)$, with $\nu > 0$. It can be shown that $d \sim pF(p, \nu)$, where $F(a, b)$ denotes the Snedecor's F distribution with parameters a and b.
- *The multivariate skew-slash distribution*, $\text{SSL}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, \nu)$, arises by taking $U \sim \text{Beta}(\nu, 1)$, with $u \in (0, 1)$ and $\nu > 0$. For this distribution, the cdf of the Mahalanobis distance is distributed as $P(d \leq r) = P\left(\chi^2_p \leq r\right) - \frac{2^\nu \Gamma(p/2+\nu)}{r^\nu \Gamma(p/2)} P\left(\chi^2_{p+2\nu} \leq r\right)$.
- *The multivariate skew-contaminated normal distribution*, $\text{SCN}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, \nu_1, \nu_2)$, where $\nu_1, \nu_2 \in (0, 1)$ which arises when the mixing scale factor U is a discrete random variable taking values ν_2 and 1 with probabilities ν_1 and $1 - \nu_1$, respectively. It can be clearly observed that the cdf of the Mahalanobis distance, in this case, is given by $P(d \leq r) = \nu_1 P\left(\chi^2_p \leq \nu_2 r\right) + (1 - \nu_1) P\left(\chi^2_p \leq r\right)$.

An important special case of the SMSN class is obtained when $\boldsymbol{\lambda} = \mathbf{0}$, when it reduces to the scale mixture of normal (SMN) class of distributions (denoted by $\mathbf{Y} \sim \text{SMN}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}; H)$), discussed earlier by Lange and Sinsheimer (1993). In this sense, the symmetric version of all the four distributions mentioned above can be attained by setting $\boldsymbol{\lambda} = \mathbf{0}$.

Now, suppose that a variable of interest is repeatedly measured for each of n subjects at certain occasions over time, along with possible covariates. For the ith subject, $i = 1, \ldots, n$, let $\mathbf{Y}_i$ be a $n_i \times 1$ vector of observed continuous responses. A linear mixed-effects model can be defined as

$$\mathbf{Y}_i = \mathbf{X}_i \boldsymbol{\beta} + \mathbf{Z}_i \mathbf{b}_i + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, n\,, \tag{2}$$

where $\mathbf{X}_i$ of dimension $n_i \times l$ is the design matrix corresponding to the fixed effects, $\boldsymbol{\beta}$ of dimension $l \times 1$ is a vector of population-averaged regression coefficients called fixed effects, $\mathbf{Z}_i$ of dimension $n_i \times q$ is the design matrix corresponding to the $q \times 1$ random effects vector $\mathbf{b}_i$, and $\boldsymbol{\epsilon}_i$ of dimension $n_i \times 1$ is the vector of random errors.

The SMSN-LMM can then be defined by assuming that

$$\begin{pmatrix} \mathbf{b}_i \\ \boldsymbol{\epsilon}_i \end{pmatrix} \overset{\text{ind.}}{\sim} \text{SMSN}_{q+n_i} \left(\begin{pmatrix} c\boldsymbol{\Delta} \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_i \end{pmatrix}, \begin{pmatrix} \boldsymbol{\lambda} \\ \mathbf{0} \end{pmatrix}; H \right), \tag{3}$$

for $i = 1, \ldots, n$, where $c = c(\boldsymbol{\nu}) = -\sqrt{\frac{2}{\pi}} k_1$, with $k_1 = \text{E}\{U^{-1/2}\}$, $\boldsymbol{\Delta} = \mathbf{D}^{1/2} \boldsymbol{\delta}$, $\mathbf{D} = \mathbf{D}(\boldsymbol{\alpha})$ is unstructured and depends on unknown and reduced parameter vector $\boldsymbol{\alpha}$, and we consider $\boldsymbol{\Sigma}_i = \sigma^2_e \mathbf{R}_i$, with $\mathbf{R}_i = \mathbf{R}_i(\boldsymbol{\phi})$, $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_p)^\top$, being a structured matrix. As long as $k_1 < \infty$, the chosen location parameter ensures that $\text{E}\{\mathbf{b}\} = \text{E}\{\boldsymbol{\epsilon}_i\} = \mathbf{0}$, so that $\text{E}\{\mathbf{Y}_i\} = \mathbf{X}_i \boldsymbol{\beta}$.

The model formulation implies that, marginally,

$$\mathbf{b}_i \overset{\text{iid.}}{\sim} \text{SMSN}_q(c\boldsymbol{\Delta}, \mathbf{D}, \boldsymbol{\lambda}; H) \text{ and } \boldsymbol{\epsilon}_i \overset{\text{ind.}}{\sim} \text{SMN}_{n_i}(\mathbf{0}, \sigma^2_e \mathbf{R}_i; H), \quad i = 1, \ldots, n.$$

Thus, the shape parameter $\boldsymbol{\lambda}$ incorporates asymmetry only in the distribution of the random effects (and consequently in the marginal distribution of $\mathbf{Y}$). Even though $\mathbf{b}_i$ and $\boldsymbol{\epsilon}_i$ are indexed by the same scale mixing factor U_i (and hence they are not marginally independent), conditional on U_i, we have that $\mathbf{b}_i$ and $\boldsymbol{\epsilon}_i$ are independent. Therefore, since $\text{Cov}\{\mathbf{b}_i, \boldsymbol{\epsilon}_i\} = \text{E}\{\mathbf{b}_i\boldsymbol{\epsilon}_i^\top\} = \text{E}_{U_i}\{\text{E}\{\mathbf{b}_i\boldsymbol{\epsilon}_i^\top|U_i\}\} = \mathbf{0}$, $\mathbf{b}_i$ and $\boldsymbol{\epsilon}_i$ are uncorrelated.

Finally, the model formulation given in (2) and (3) implies that, marginally,

$$\mathbf{Y}_i \overset{\text{ind.}}{\sim} \text{SMSN}_{n_i}(\mathbf{X}_i\boldsymbol{\beta} + \mathbf{Z}_i c\boldsymbol{\Delta}, \boldsymbol{\Psi}_i, \bar{\boldsymbol{\lambda}}_i; H), \tag{4}$$

where $\boldsymbol{\Psi}_i = \boldsymbol{\Sigma}_i + \mathbf{Z}_i\mathbf{D}\mathbf{Z}_i^\top$, $\bar{\boldsymbol{\lambda}}_i = \dfrac{\boldsymbol{\Psi}_i^{-1/2}\mathbf{Z}_i\mathbf{D}\boldsymbol{\zeta}}{\sqrt{1+\boldsymbol{\zeta}^\top\boldsymbol{\Lambda}_i\boldsymbol{\zeta}}}$, with $\boldsymbol{\zeta} = \mathbf{D}^{-1/2}\boldsymbol{\lambda}$ and $\boldsymbol{\Lambda}_i = (\mathbf{D}^{-1} + \mathbf{Z}_i^\top\boldsymbol{\Sigma}_i^{-1}\mathbf{Z}_i)^{-1}$.

In order to enable some flexibility when modeling the error scale matrix, we consider three possible dependence structures for $\mathbf{R}_i$: uncorrelated, AR(p), and DEC, which will be briefly introduced next.

- Uncorrelated: The most common and simplest approach is to assume that the error terms are uncorrelated (UNC). Under this assumption, we have $\mathbf{R}_i = \mathbf{I}_{n_i}$, for $i = 1, \ldots, n$.

- Autoregressive dependence of order p: Consider at first the case where a variable of interest is regularly observed over discrete time, n_i times, for each subject. Then, we propose to model $\mathbf{R}_i$ as a structured AR(p) dependence matrix (Box and Jenkins 1976). Specifically,

$$\mathbf{R}_i = \mathbf{R}_i(\boldsymbol{\phi}) = \frac{1}{1 - \phi_1\rho_1 - \ldots - \phi_p\rho_p}[\rho_{|r-s|}],$$

  where $r, s = 1, \ldots, n_i$, $i = 1, \ldots, n$, and $\rho_1, \ldots, \rho_p$ are the theoretical autocorrelations of the process, and thereby they are functions of autoregressive parameters $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_p)^\top$, and satisfy the Yule–Walker equations (Box and Jenkins 1976). To accommodate situations in which measurements are taken irregularly over discrete time, we modify $\mathbf{R}_i$ by computing it for a regular range of time and then suppressing the line and column regarding the position from the missing measurements.

- Damped exponential correlation: More generally, consider now that for each subject, a variable of interest is observed at times $\mathbf{t}_i = (t_{i1}, t_{i2}, \ldots, t_{in_i})$. Following Muñoz, Carey, Schouten, Segal, and Rosner (1992), we propose to structure $\mathbf{R}_i$ as a damped exponential correlation (DEC) matrix, as follows:

$$\mathbf{R}_i = \mathbf{R}_i(\boldsymbol{\phi}, \mathbf{t}_i) = \left[\phi_1^{|t_{ij}-t_{ik}|^{\phi_2}}\right], \quad 0 \leq \phi_1 < 1, \ \phi_2 \geq 0,$$

  where $j, k = 1, \ldots, n_i$, for $i = 1, \ldots, n$, and $\boldsymbol{\phi} = (\phi_1, \phi_2)^\top$. Note that for $\phi_2 = 1$, $\mathbf{R}_i$ reduces to the correlation matrix of a continuous-time autoregressive process of order 1 (CAR1). Hence, ϕ_2 enables attenuation or acceleration of the exponential decay from a CAR1 autocorrelation function, depending on its value.

Since the log-likelihood function for the proposed model involves complex expressions and is challenging to be optimized, ML estimation of $\boldsymbol{\theta} = (\boldsymbol{\beta}^\top, \sigma_e^2, \boldsymbol{\phi}^\top, \boldsymbol{\alpha}^\top, \boldsymbol{\lambda}^\top, \boldsymbol{\nu}^\top)^\top$ is performed using an EM-type algorithm (for details on parameter estimation, please see Schumacher *et al.* 2021), which takes advantage of a hierarchical representation of the model. Some options for parameter estimation are available in the **skewlmm** package and are introduced in the next section.

3. Algorithms for parameter estimation

The required computation time to estimate EM algorithms (or to solve fixed-point problems in general) is a frequent concern in applications, especially when dealing with complex models or big data. There are several proposals in the literature to accelerate the often-slow convergence of the EM algorithm. For example, Varadhan and Roland (2008) developed a class of iterative schemes, called squared iterative methods, that uses the novel idea of "squaring" applied to Steffensen-type methods for EM acceleration and can be implemented as an "off-the-shelf" accelerator of any EM algorithm.

Other methods that are frequently used for EM acceleration are the Anderson acceleration (AA, Anderson 1965) and further modifications to include, for example, restarts (e.g., Pratapa and Suryanarayana 2015). These methods use information gained from previous iterations, having the advantage of only using the current and past iterates of the sequence of parameter values and the corresponding EM mappings of these parameter values and only requiring modest additional storage and per-iteration computational costs (Henderson and Varadhan 2019).

Recently, Henderson and Varadhan (2019) described a new class of acceleration schemes built on the AA technique and introduced periodic restarts, a damping factor, and "epsilon-monotonicity" control, and is referred to as the DAAREM method. The introduction of a damped or a regularized version of the least-squares problem used in the AA scheme enables a compromise between the EM step and the AA step, connecting the robustness of EM to initial values with the speed of local convergence of AA or restarted AA.

The DAAREM algorithm serves as a generic "off-the-shelf" accelerator and is implemented in the R package **daarem** (Henderson and Varadhan 2020). It is used by default for estimation in the **skewlmm** package since simulation studies showed an expressive gain in time until convergence in comparison to the traditional EM algorithm (Schumacher 2021). Nevertheless, the traditional implementation is also available, and its use can be specified via the argument `control`.

Complementary, the use of parallel optimization can improve the computational time in some situations, but its performance depends on the evaluation time of the objective function in relation to the parallel overhead. An easy-to-use parallelization option is implemented in the R package **optimParallel** (Gerber and Furrer 2019), whose function `optimParallel()` provides a parallel version of the L-BFGS-B optimization method of `optim()` and is optionally used for parameter estimation in the **skewlmm** package. The numerical optimization processes are necessary to update $\boldsymbol{\phi}$ and $\boldsymbol{\nu}$ in the maximization step of the EM algorithm, and the use of parallel computation in the updates can be controlled using the argument `control`, whose use is described in the next section.

4. Implementation in R

Two main functions are available: `smsn.lmm()` and `smn.lmm()`, that fit SMSN-LMMs and SMN-LMMs, respectively. Their syntax is as follows:

```
smn.lmm(data, formFixed, groupVar, formRandom = ~1, depStruct = "UNC",
  timeVar = NULL, distr = "norm", covRandom = "pdSymm",
  pAR = 1, control = lmmControl())
smsn.lmm(data, formFixed, groupVar, formRandom = ~1, depStruct = "UNC",
  timeVar = NULL, distr = "sn", covRandom = "pdSymm",
  skewind, pAR = 1, control = lmmControl())
```

where:

- `data` is a data frame containing all variables to be used in the model.
- `formFixed` is a two-sided linear formula object describing the fixed effects part of the model.
- `groupVar` is a character containing the name of the variable which represents the subjects or groups in `data`.
- `formRandom` is a one-sided linear formula object describing the random effects part of the model (default is a random intercept model).
- `depStruct` is a character indicating which dependence structure should be used for the error: `"UNC"` for uncorrelated (default), `"ARp"` for autoregressive, `"CS"` for compound symmetry, `"DEC"` for DEC, or `"CAR1"` for continuous-time AR(1).
- `timeVar` is a character containing the name of the variable which represents the time in data (meaningless if `depStruct = "UNC"` or `depStruct = "CS"`; otherwise, if `is.null(timeVar)` the observations are considered equally spaced and ordered).
- `distr` is a character indicating which distribution should be used (default is a skew-normal distribution).
- `covRandom` is a character indicating which structure should be used for the random effects scale matrix: either `"pdSymm"` (default), for a general positive-definite matrix, or `"pdDiag"`, for a diagonal matrix.
- `skewind` is a vector of length equal to the number of random effects (q), containing 0's and 1's, indicating which elements of $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_q)$ should be estimated (default is an all-ones vector).
- `pAR` is the order of the autoregressive process that should be used (default is `1` if `depStruct = "ARp"`, meaningless otherwise).
- `control` is an object resulting from the function `lmmControl()`, containing additional options for the estimation algorithm.

4.1. Tools for model evaluation

An important step in data analysis is to evaluate the suitability of a fitted model to a real data set, and several methods can be used for this purpose. The tools available in the **skewlmm** for model evaluation are introduced next.

First, conditional and marginal residuals can be accessed via the method `residuals()`. Let $\mathbf{A}$ be an identity matrix if `type = "response"`, the variance of the response variable (for details, see Schumacher *et al.* 2021) if `type = "normalized"`, and $\mathbf{A} = \boldsymbol{\Psi}$ (as given in (4)) if `type = "modified"`. For the ith subject, marginal residuals are given by

$$\mathbf{r}_i = \widehat{\mathbf{A}}_i^{-1/2}\left(\mathbf{y}_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}}\right)$$

and conditional residuals are given by

$$\mathbf{s}_i = \widehat{\mathbf{A}}_i^{-1/2}\left(\mathbf{y}_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}} - \mathbf{Z}_i\widehat{\mathbf{b}}_i\right),$$

where $\widehat{\mathbf{b}}_i = \widehat{\mathbf{b}}_i(\widehat{\boldsymbol{\theta}}) = \mathrm{E}\{\mathbf{b}_i|\mathbf{Y}_i = \mathbf{y}_i, \widehat{\boldsymbol{\theta}}\}$ can be extracted using the function `ranef()`. Moreover, plotting an object containing a fitted model will return a fitted versus residuals plot, with the color indicating the estimated weight ($\widehat{u}_i$) of the observations.

When dealing with heavy-tailed data, the Mahalanobis distance is a convenient measure that can be used to identify potential outlying observations and to assess the validity of the underlying distributional assumption of the response variable, because if the fitted model is appropriate, the distribution of the Mahalanobis distance is known. The function `mahalDist()` returns the Mahalanobis distance from a fitted model, for which a `plot()` method is available that additionally plots a theoretical quantile of the assumed distribution.

Following Ho and Lin (2010), to assess the goodness of fit of an SMSN-LMM, one can construct a Healy-type plot (Healy 1968) by plotting the nominal probability values $1/n, 2/n, \ldots, n/n$ against the theoretical cumulative probabilities of the ordered observed Mahalanobis distances. For a fitted model, this plot is generated by the function `healy.plot()`. If the fitted model is appropriate, the plot should resemble a straight line through the origin with a unit slope. It is important to note that Haley's plot requires all subjects to have the same number of observations, but in case of unbalanced data unmeasured observations can be predicted using the method `predict()`, and an imputed data set can be provided through the argument `dataPlus`.

Additionally, based on Zeller, Labra, Lachos, and Balakrishnan (2010), the observed Mahalanobis distance can be decomposed as follows: $d_i(\widehat{\boldsymbol{\theta}}) = (\mathbf{y}_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}} - \widehat{c}\,\mathbf{Z}_i\widehat{\boldsymbol{\Delta}})^\top \widehat{\boldsymbol{\Psi}}_i^{-1}(\mathbf{y}_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}} - \widehat{c}\,\mathbf{Z}_i\widehat{\boldsymbol{\Delta}}) = \mathbf{e}_i^\top\widehat{\boldsymbol{\Sigma}}_i^{-1}\mathbf{e}_i + \left(\widehat{\boldsymbol{\mu}}_{bi} - \widehat{c}\widehat{\boldsymbol{\Delta}}\right)^\top \widehat{\mathbf{D}}^{-1}\left(\widehat{\boldsymbol{\mu}}_{bi} - \widehat{c}\widehat{\boldsymbol{\Delta}}\right) = d_{\mathbf{e}_i}(\widehat{\boldsymbol{\theta}}) + d_{\mathbf{b}_i}(\widehat{\boldsymbol{\theta}})$, where $\mathbf{e}_i = \mathbf{y}_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}} - \mathbf{Z}_i\widehat{\boldsymbol{\mu}}_{bi}$ and $\boldsymbol{\mu}_{bi} = c\boldsymbol{\Delta} + \mathbf{D}\mathbf{Z}_i^\top\boldsymbol{\Psi}_i^{-1}(\mathbf{y}_i - \mathbf{X}_i\boldsymbol{\beta} - c\mathbf{Z}_i\boldsymbol{\Delta})$. This decomposition gives insight into how the estimated random effects $\widehat{\mathbf{b}}_i$ and the estimated residuals $\mathbf{e}_i$ affect the overall distance, and it is returned by the function `mahalDist()` with the argument `decomposed = TRUE`.

Another important assumption that should be investigated is the dependence structure assumed to the within-subject errors. In the context of time series data, a commonly used tool to analyze serial correlation is the empirical autocorrelation function (ACF, Box and Jenkins 1976). In the context of mixed models, Pinheiro and Bates (2000) proposed to use the empirical autocorrelation function for the residuals of a fitted LMM. Based on this approach

and restricting to the case that data is observed at discrete times, let $\mathbf{r}_i^\top = (r_{it_1}, \ldots, r_{it_{n_i}})$ denote normalized marginal residuals. The empirical autocorrelation at lag l is computed by the function `acfresid()` and can be defined as

$$\widehat{\rho}(l) = \frac{\sum_{i=1}^{n} \sum_{\{(j,k)|t_k - t_j = l\}} r_{it_j} r_{it_k} / N(l)}{\sum_{i=1}^{n} \sum_{j=1}^{n_i} r_{it_j}^2 / N(0)}, \tag{5}$$

where $N(\cdot)$ is the number of pairs used in the respective numerator summation. If the within-subject dependence structure is correct, $\widehat{\rho}(l)$ is expected to be close to zero.

Since $\mathbf{r}_i$'s distribution is not symmetrical, the interval estimates of $\rho(\cdot)$ that are commonly used in time series models are not appropriate. Alternatively, if `calcCI = TRUE`, a Monte Carlo estimate for an uncorrelated model is computed by generating M samples from a UNC-SMSN-LMM similar to the fitted model, calculating the standardized marginal residuals (with respect to $\widehat{\boldsymbol{\theta}}$ estimated from the original data set) and $\widehat{\rho}(l)$ for each sample, and using empirical $100(\alpha/2)$th and $100(1-\alpha/2)$th percentiles as approximate interval estimates of level $1-\alpha$. If the considered dependence structure is appropriate, we expect approximately $100(1-\alpha)\%$ of the empirical autocorrelations to belong to the uncorrelated interval. The method `plot()` is available to facilitate the ACF visualization.

A similar approach is used to generate reference bounds for Healy-type plots, but in this case, the samples are generated from the same SMSN-LMM that was fitted.

Finally, the function `lr.test()` can be used to perform a likelihood-ratio test of two nested SMSN-LMMs, which might be useful to test if $\boldsymbol{\lambda} = 0$, for example, and the function `criteria()` extracts information criteria for several fitted models.

5. Data illustrations

To provide examples of the package usage in real applications, in this section we present two real data scenarios. The first one consists of balanced data with measurements equally spaced in time, for which all tools for model evaluation can be easily used. In this example, a model considering an asymmetric distribution provides a better fit for the data.

On the other hand, for the second example, measurements were taken under different time gaps for different subjects, and more care is needed in its analysis. Here, a model considering a symmetric distribution, specifically the contaminated normal distribution, yields the best results.

Additionally, aiming to illustrate the package's feature of data generation and consider the computational time for handling larger samples sizes, a third example involving synthetic data sets is presented. Taking advantage of knowing the model's generation process, we compare different methods for obtaining confidence intervals and report computational cost for model fitting.

5.1. Example with balanced data: Mice weight data

To illustrate the use of the **skewlmm** package, we now consider the `miceweight` data set, which is available at the R package **skewlmm**. This data set was derived from results of a clinical trial designed to test two diet treatments in comparison to a control group. The weight (in grams) of 52 mice was measured weekly from baseline until week 10 of treatment.

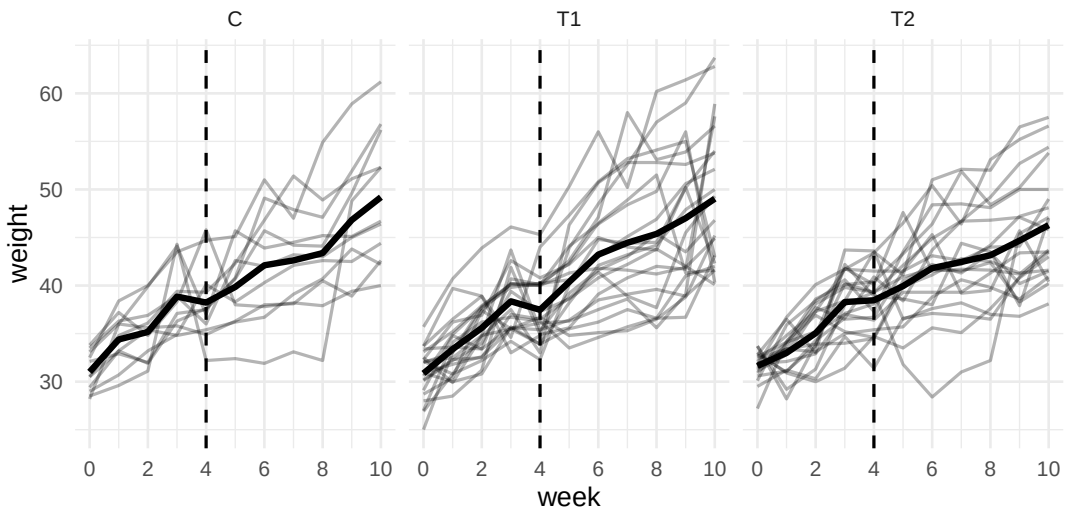

Figure 1: Mice weight data set. Trajectories of mice weight for different diets.

Minor location perturbations were performed in the publicly available version of the data set for confidentiality concerns; however, original data features such as variance and skewness were preserved.

The goal of the study was to test if either of the two special diets yield a faster weight gain. Following four initial weeks of no intervention, 21 mice randomly assigned to treatment 1 (T1) and 20 to treatment 2 (T2) started receiving special diets for weight gain, while 11 mice assigned to the control (C) group continued receiving a standard diet. Figure 1 presents individual weight trajectories over time, along with their mean profile.

After loading the **skewlmm** package and the data set,

```
R> library("skewlmm")
R> data("miceweight", package = "skewlmm")
```

we create the variables `interWeek` and `interTreat`, respectively measuring week since intervention started and indicator of time that each mouse received an intervention, as follows:

```
R> miceweight <- miceweight %>%
+   mutate(interWeek = week - 4) %>%
+   mutate(interTreat = if_else(interWeek < 0, "C", treat))
```

We aim to fit the model

$$Y_{ij} = \beta_0 + b_{0i} + (\beta_1 + b_{1i})t_{ij} + \beta_2 \mathbb{1}_{treat_i=T_1, t_{ij}\geq 0} + \beta_3 \mathbb{1}_{treat_i=T_2, t_{ij}\geq 0} + \beta_4 \mathbb{1}_{treat_i=T_1, t_{ij}\geq 0} t_{ij} + \beta_5 \mathbb{1}_{treat_i=T_2, t_{ij}\geq 0} t_{ij} + \epsilon_{ij},$$

$i = 1, \ldots, 52$, $j = 1, \ldots, 11$, where Y_{ji} is the weight for the ith mouse at the jth week, $\mathbb{1}_A$ is an indicator function (1 if A is true, 0 otherwise), and t_{ij} represents week since intervention started (`interWeek`).

First, we consider the convenient normal assumption for both random effects and errors. Figure 2 presents the estimated random effects from the fitted model, where we can see

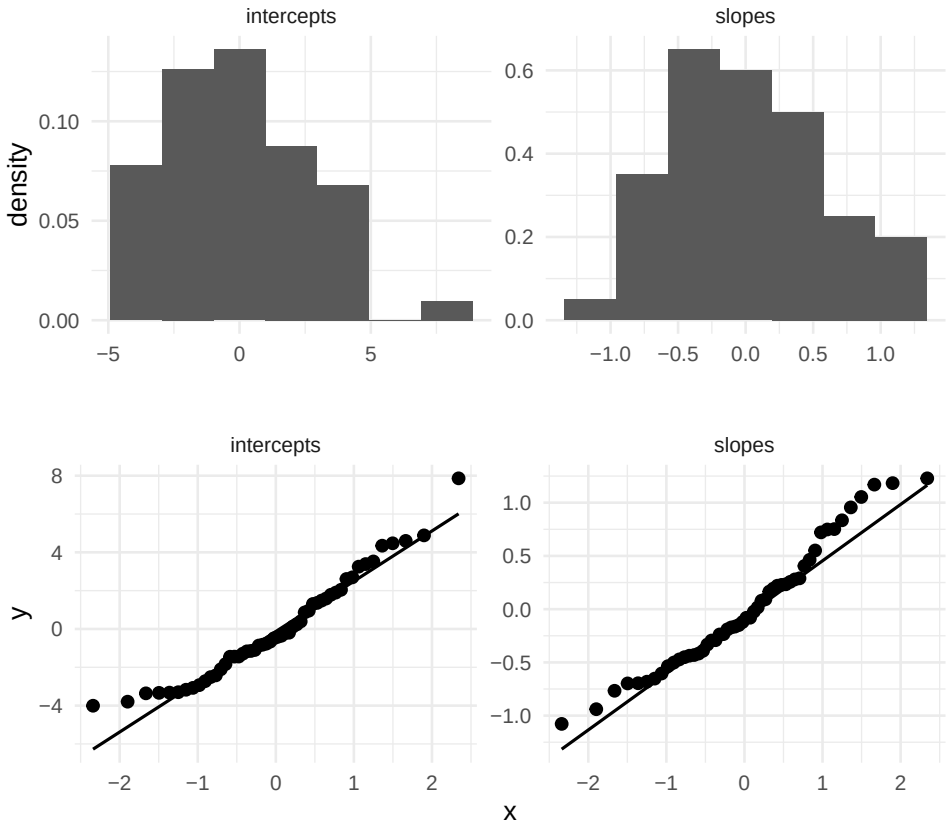

Figure 2: Mice weight data set. Histograms and quantile plots of the estimated random effects obtained from fitting an N-LMM.

that the normal assumption does not seem appropriate due to indications of both skewness and outliers. Therefore, the use of a more robust model seems advantageous and will be investigated next.

We can fit a classic model N-LMM, and, for example, an SL-LMM, and an SSL-LMM using the following code:

```
R> fit_norm <- smn.lmm(data = miceweight,
+    formFixed = weight ~ interWeek * interTreat,
+    formRandom = ~ interWeek, groupVar = "mouseID")
R> fit_sl <- update(object = fit_norm, distr = "sl")
R> fit_ssl <- smsn.lmm(data = miceweight,
+    formFixed = weight ~ interWeek * interTreat,
+    formRandom = ~ interWeek, groupVar = "mouseID", distr = "ssl")
```

The function `update()` refits the model stored in `object` by changing the extra arguments provided.

We can see some information regarding the estimated model by simply printing the fitted object:

```
R> fit_ssl
```

```
Linear mixed models with distribution ssl and dependence structure UNC
Log-likelihood value at convergence: -1468.949
Distribution ssl with nu = 1.157208
Fixed: weight ~ interWeek * interTreat
           (Intercept)              interWeek           interTreatT1
            39.8191240              2.0114239             -1.6367405
          interTreatT2 interWeek:interTreatT1 interWeek:interTreatT2
            -0.2796527             -0.3325641             -0.5836130
Random effects:
  Formula: ~ interWeek by mouseID
  Structure: General positive-definite
  Estimated variance (D):
            (Intercept) interWeek
(Intercept)    9.012879 1.7687376
interWeek      1.768738 0.4020359
  Skewness parameter: 27.51208 8.270156
Error dependence structure: UNC
  Estimate(s):
  sigma2
2.561305
Number of observations: 572
Number of groups: 52
```

To evaluate the necessity of using the skewed distribution, we can perform a likelihood ratio test for testing $H_0 : \boldsymbol{\lambda} = \mathbf{0}$ using the function `lr.test()` and the two nested fitted models:

```
R> lr.test(fit_sl, fit_ssl)

Model selection criteria:
           logLik      AIC      BIC
fit_sl  -1473.938 2969.876 3017.717
fit_ssl -1468.949 2963.899 3020.438

    Likelihood-ratio Test

chi-square statistics =  9.977532
df =  2
p-value =  0.006814066

The null hypothesis that both models represent the
data equally well is rejected at level  0.05
```

Since the p-value is quite small, we reject the null hypothesis that both skewed and symmetric models represent the data equally well. Hence, the skewed model provides a significantly improved fit to the data.

To evaluate the adequacy of the distributional assumption, we can produce Healy-type plots as follows:

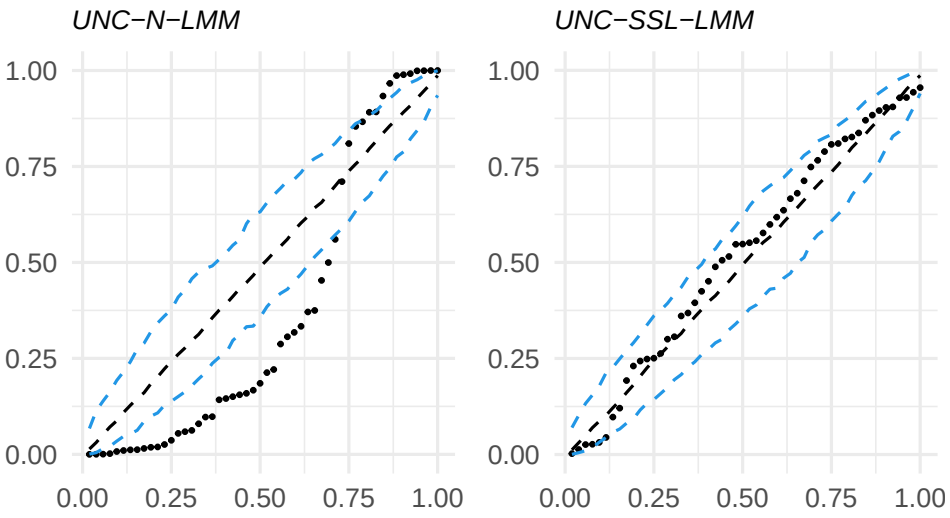

Figure 3: Mice weight data set. Healy-type plots.

```
R> grid.arrange(healy.plot(fit_norm, calcCI = TRUE),
+    healy.plot(fit_ssl, calcCI = TRUE), nrow = 1)
```

The resulting plot is presented in Figure 3, where the dashed lines represent the 2.5%, 50%, and 97.5% percentiles obtained from simulated samples, as described in the Section 4.1. The gain in considering a heavy-tailed distribution for modeling this data set can be observed.

Furthermore, to check if the uncorrelation assumption (the function's default) is appropriate, a possible approach is to refit the model considering different dependence structures and then compare AIC and BIC values to select the most suitable model. Since the data are equally spaced and sorted by time, the use of `timeVar`, in this case, is optional (if not provided, it will be automatically generated).

```
R> fit_ssl_ar1 <- update(fit_ssl, depStruct = "ARp", pAR = 1)
R> fit_ssl_ar2 <- update(fit_ssl, depStruct = "ARp", pAR = 2)
R> fit_ssl_DEC <- update(fit_ssl, depStruct = "DEC", timeVar = "interWeek")
```

By default, the functions `smsn.lmm()` and `smsn.lmm()` only use `optimParallel` to update $\widehat{\phi}$ at each iteration if the data contain more than 30 subjects. However, we can control the use of parallel computing by specifying `parallelphi` in the `lmmControl()` function passed to the argument `control`:

```
R> fit_ssl_DEC_par <- update(fit_ssl_DEC,
+    control = lmmControl(parallelphi = TRUE))
```

The parallelized version was slightly more efficient in this case, as can be seen in the following results:

```
R> fit_ssl_DEC$elapsedTime

[1] 58.79464
```

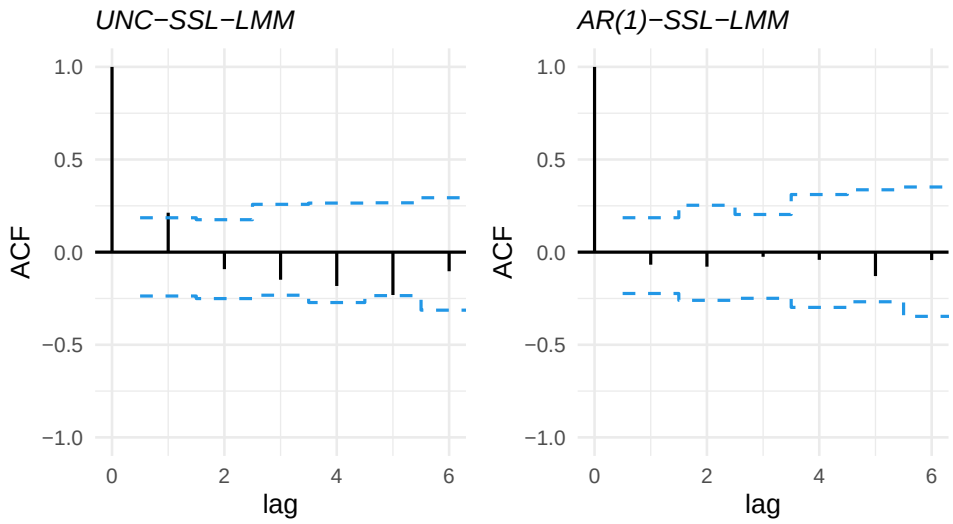

Figure 4: Mice weight data set. ACF plots.

```
R> fit_ssl_DEC_par$elapsedTime
```

```
[1] 41.90837
```

Now, comparing the information criteria from the fitted models, we see that the smaller AIC and BIC is provided by the model with AR(1) dependence.

```
R> criteria(list(UNC = fit_ssl, AR1 = fit_ssl_ar1, AR2 = fit_ssl_ar2,
+    DEC = fit_ssl_DEC))
```

```
       logLik npar      AIC      BIC
UNC -1468.949   13 2963.899 3020.438
AR1 -1438.826   14 2905.653 2966.541
AR2 -1438.907   15 2907.814 2973.051
DEC -1438.925   15 2907.849 2973.086
```

Additionally, we can compute empirical autocorrelations using the function `acfresid()` and plot the results with the following code:

```
R> grid.arrange(plot(acfresid(fit_ssl, calcCI = TRUE, maxLag = 6)),
+    plot(acfresid(fit_ssl_ar1, calcCI = TRUE, maxLag = 6)), nrow = 1)
```

From the resulting plot in Figure 4, where the dashed lines represent the 2.5% and 97.5% percentiles obtained from simulated samples, we can see some gain in considering the correlated model since the empirical autocorrelations were generally smaller. Furthermore, the residuals' ACF from the AR(1)-SL-LMM does not seem to indicate unmodeled serial dependence, and therefore we will select it for further analysis.

We can extract information about the fit using the method `summary`:

```
R> summary(fit_ssl_ar1)
```

```
Linear mixed models with distribution ssl and dependence structure ARp
Call:
smsn.lmm(data = miceweight, formFixed = weight ~ interWeek *
    interTreat, groupVar = "mouseID", formRandom = ~interWeek,
    depStruct = "ARp", distr = "ssl", pAR = 1)

Distribution ssl with nu = 1.078377

Random effects:
  Formula: ~interWeek
  Structure:
  Estimated variance (D):
            (Intercept) interWeek
(Intercept)    6.616008 1.4409527
interWeek      1.440953 0.3154857

Fixed effects: weight ~ interWeek * interTreat
with approximate confidence intervals
                           Value Std.error CI 95% lower CI 95% upper
(Intercept)            39.8628406 0.7270794   38.4377910  41.28789011
interWeek               2.0349189 0.1844456    1.6734122   2.39642556
interTreatT1           -2.1714324 1.0441153   -4.2178607  -0.12500405
interTreatT2           -0.8536847 0.9025629   -2.6226754   0.91530598
interWeek:interTreatT1 -0.1687789 0.2262999   -0.6123186   0.27476075
interWeek:interTreatT2 -0.4701132 0.2779825   -1.0149489   0.07472247

Dependence structure: ARp
  Estimate(s):
  sigma2     phi1
2.510868 0.464750

Skewness parameter estimate: 98.36299 18.61088

Model selection criteria:
    logLik      AIC      BIC
 -1438.826 2905.653 2966.541

Number of observations: 572
Number of groups: 52
```

We can analyze the observed Mahalanobis distance and its relation with the estimated weights ($\widehat{\boldsymbol{u}}$) using the following code:

```
R> grid.arrange(plot(mahalDist(fit_ssl_ar1), nlabels = 0),
+    weight_plot(fit_ssl_ar1), ncol = 2)
```

The resulting plot is shown in Figure 5, where the blue dashed line is the theoretical 99% quantile. All distances were smaller than the given quantile, and it can be seen that the

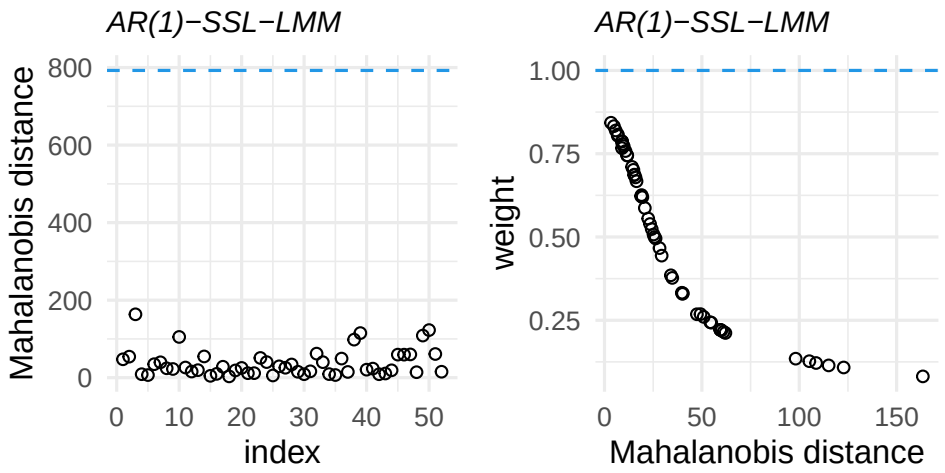

Figure 5: Mice weight data set. Mahalanobis distance versus index (left) and estimated weight versus Mahalanobis distance (right).

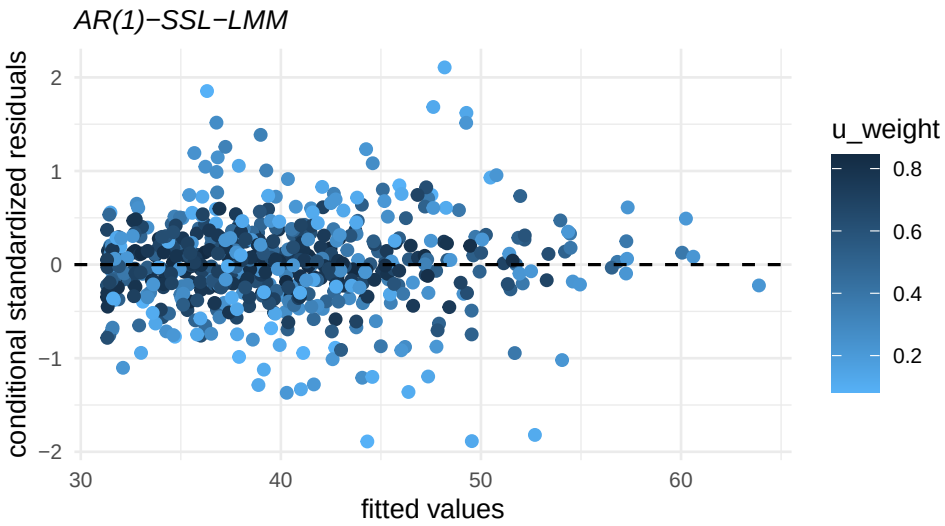

Figure 6: Mice weight data set. Plotting the fitted model object.

two subjects that presented larger Mahalanobis distance received quite small weights in the estimation processes, as opposed to the estimation under Gaussian assumptions (that gives the same weight to all subjects and all observations).

Moreover, plotting the fitted object (as follows) results in Figure 6, where it can be seen that larger residuals are associated with smaller weights and therefore have less impact on the estimation procedure, showing evidence of the model's robustness.

```
R> plot(fit_ssl_ar1, type = "normalized")
```

Finally, we can extract confidence intervals (CIs) using the function `confint`:

```
R> confint(fit_ssl_ar1) %>% round(digits = 3)
```

```
                      Estimate CI 95% lower CI 95% upper
(Intercept)             39.863       38.438       41.288
interWeek                2.035        1.673        2.396
interTreatT1            -2.171       -4.218       -0.125
interTreatT2            -0.854       -2.623        0.915
interWeek:interTreatT1  -0.169       -0.612        0.275
interWeek:interTreatT2  -0.470       -1.015        0.075
sigma2                   2.511        1.970        3.052
phiAR1                   0.465        0.311        0.619
Dsqrt11                  2.515        1.528        3.502
Dsqrt12                  0.539        0.121        0.958
Dsqrt22                  0.157       -0.903        1.217
lambda1                 98.363           NA           NA
lambda2                 18.611           NA           NA
nu1                      1.078           NA           NA
```

By default, CIs are computed based on the asymptotic normal distribution of maximum likelihood estimators. When dealing with small samples, it may be helpful to compute parametric bootstrap confidence intervals (based on B simulated samples), which can be done by specifying the `method` argument as `"bootstrap"`:

```
R> boot_ssl_ar1 <- confint(fit_ssl_ar1, method = "bootstrap", B = 100)
R> boot_ssl_ar1 %>% round(digits = 3)
```

```
                      Estimate   2.5%  97.5%
(Intercept)             39.863 38.625 41.077
interWeek                2.035  1.741  2.294
interTreatT1            -2.171 -3.195 -1.077
interTreatT2            -0.854 -1.858  0.141
interWeek:interTreatT1  -0.169 -0.551  0.206
interWeek:interTreatT2  -0.470 -0.847 -0.127
sigma2                   2.511  1.916  3.316
phiAR1                   0.465  0.336  0.554
Dsqrt11                  2.515  2.351  2.674
Dsqrt12                  0.539  0.388  0.695
Dsqrt22                  0.157  0.105  0.251
lambda1                 98.363 97.546 99.420
lambda2                 18.611  5.644 22.870
nu1                      1.078  0.818  1.805
```

We note that for this example, testing a hypothesis $H_0 : \beta_5 = 0$ (the parameter associated with the interaction of treatment 2 and time) would yield different conclusions based on the asymptotic intervals versus the bootstrapped intervals.

Some additional options are available, including some special cases of the models considered thus far and some estimation controls. For example, a reduced model that can be useful considers a diagonal matrix as the scale matrix of the random effects, which is obtained by specifying `covRandom = "pdDiag"`. Updating the fitted model (stored at `fit_sl_ar1`),

```
R> fit_ssl_ar1D <- update(fit_ssl_ar1, covRandom = "pdDiag")
```

and performing a likelihood ratio test to evaluate if the reduced model is appropriate,

```
R> lr.test(fit_ssl_ar1, fit_ssl_ar1D)

Model selection criteria:
                logLik      AIC      BIC
fit_ssl_ar1  -1438.826 2905.653 2966.541
fit_ssl_ar1D -1468.381 2962.762 3019.301

    Likelihood-ratio Test

chi-square statistics =  59.10911
df =  1
p-value =  1.491669e-14

The null hypothesis that both models represent the
data equally well is rejected at level  0.05
```

results in a p-value < 0.001, and therefore for these data, we can conclude that the more general structure is necessary (`"pdSymm"`).

For skewed models, it might be also interesting to test if a subset of the shape parameter $\boldsymbol{\lambda}$ is zero. This can be done by fitting a reduced model that forces some elements of $\boldsymbol{\lambda}$ to be zero using the argument `skewind`, that receives a vector of length q containing 0's for the elements that should be forced to zero, and 1's otherwise. Furthermore, if we would like to use an EM-type algorithm instead of DAAREM, we can pass `algorithm = "EM"` to the argument `control` using the function `lmmControl()`, as illustrated next:

```
R> fit_ssl1 <- update(fit_ssl, skewind = c(1, 0),
+    control = lmmControl(algorithm = "EM"))
```

Moreover, by default, the package uses the following initial values.

- For $\boldsymbol{\beta}, \sigma_e^2$, and $\mathbf{D}$, they are obtained using the classical LMM through the `lme()` function from **nlme** package in R.
- For $\boldsymbol{\lambda}$ (when using `smsn.lmm()`), they are chosen as $1 \times \text{sign}(\rho)$, where ρ is the sample skewness coefficient from the random effect estimated from the classical LMM.
- For $\boldsymbol{\nu}$, 10 is used for t/ST distributions, 5 for SL/SSL distributions, and $(0.05, 0.8)$ for CN/SCN distributions.
- For AR(p) and for CAR1 dependence, $\boldsymbol{\phi}$ is initialized as its estimate from fitting an AR(p)-LMM and a CAR1-LMM using `lme()` function from **nlme** package in R, respectively.
- For DEC dependence, $\boldsymbol{\phi}$ is initialized by finding the maximum marginal log-likelihood function as in (4) on a grid of $\boldsymbol{\phi}$ and for other parameters fixed.

Nevertheless, different initial values can be specified using the argument `initialValues` from the `lmmControl()` function, as follows:

```
R> fit_ssl2 <- update(fit_ssl,
+    control = lmmControl(initialValues = list(nu = 1)))
```

The `lmmControl()` function also allows control of some estimation, parallelization, and DAAREM options. For more details regarding the control options, please see `help(lmmControl)`.

5.2. Example with data measured on continuous time: Six Cities study

Now, we analyze a subsample from the Six Cities Study of Air Pollution and Health (Dockery, Berkey, Ware, Speizer, and Ferris Jr. 1983), focusing on pulmonary function (measured as FEV1) and height for girls living in Topeka, Kansas. This subsample was previously discussed in Fitzmaurice, Laird, and Ware (2012) and is available as supplementary material. The main goal of the study was to analyze lung growth as measured by changes in pulmonary function in children and adolescents, and the factors that influence lung function growth. In this subsample, most girls were enrolled between the ages of six and eight, and were follow-up once a year until graduation from high school or loss to follow-up. At each annual examination, FEV1 was measured though a spirometry test, and a respiratory health questionnaire was completed by a parent or guardian.

Since we are interesting in evaluating the pulmonary function over time, we removed from the sample 48 girls who had no follow-ups. Figure 7 presents trajectories of FEV1 over both age and height, in addition to smooth curves (blue line) obtained using generalized additive model (GAM) with cubic splines. Note that both smooth curves seem to be roughly piecewise linear, changing the slope around the age of 15 and height of 1.5.

Additionally, it is important to remark that for this data set, measures were obtained over continuous time, and therefore there it is not appropriate to plot the average for each time point, nor considering dependence structures based on discrete time, such as the AR(p).

We can read the data set and make some basic transformations using

```
R> ds <- read.dta("https://content.sph.harvard.edu/fitzmaur/ala2e/fev1.dta")
R> ds <- ds %>% filter(!(id %in% unique(id)[table(id) == 1]))
R> ds <- ds %>% transform(
+    agec = age - 12,
+    time = age - baseage,
+    fev1 = exp(logfev1),
+    htp = case_when(ht> 1.5 ~ ht - 1.5, TRUE ~ 0),
+    agep = case_when(age > 15 ~ age - 15, TRUE ~ 0))
```

Fitzmaurice *et al.* (2012) analyzed this sample using a logarithmic transformation and under normality assumption. We now revisit this problem proposing to model FEV1 directly, such that the parameter's interpretation is direct, using the R package **skewlmm**. Based on the patterns observed in Figure 7, we propose to fit the model:

$$\begin{aligned} y_{ij} = \beta_0 + \beta_1 \,\mathrm{age}_{i0} + \beta_2 \,\mathrm{age}_{ij} + \beta_3 \,\mathrm{age}_{ij} \mathbb{1}_{\mathrm{age}_{ij}>15} + \\ + \beta_4 \,\mathrm{ht}_{i0} + \beta_5 \,\mathrm{ht}_{ij} + \beta_6 \,\mathrm{ht}_{ij} \mathbb{1}_{\mathrm{ht}_{ij}>1.5} + \beta_7 \,\mathrm{age}_{ij}\mathrm{ht}_{ij} + b_{0i} + b_{1i} \,\mathrm{ht}_{ij} + \varepsilon_{ij}, \end{aligned} \tag{6}$$

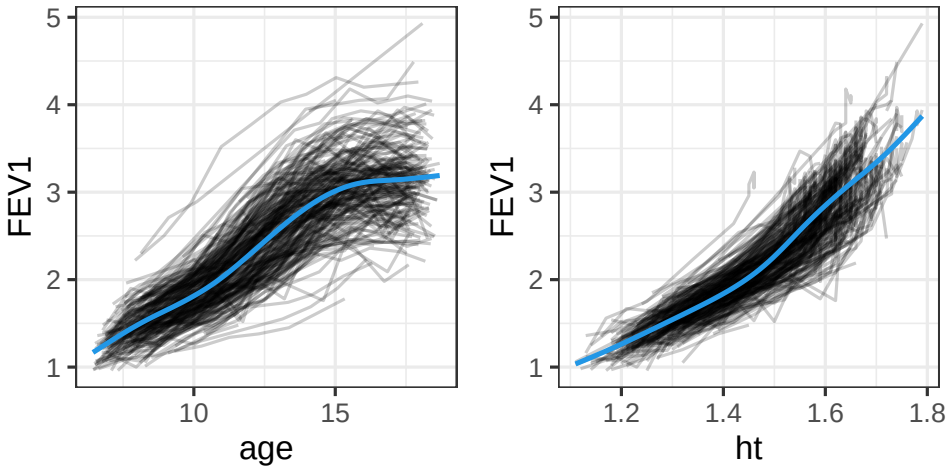

Figure 7: Six Cities study. Trajectories of FEV1 by age and height.

for $i = 1, \ldots, 252$ and $j = 1, \ldots, n_i$, where $\mathbb{1}_A$ is an indicator function (1 if A is true, 0 otherwise), age_{i0} and ht_{i0} denote age and height at baseline, respectively. We consider an interaction between age and height since, for the age range used in the study, it is not reasonable to interpret a change in age without considering a change in height.

After loading the package **skewlmm**, a normal model can be fitted using

```
R> mod_norm <- smn.lmm(
+    fev1 ~ baseage + age + agep + baseht + ht + htp + age * ht,
+    data = ds, groupVar = "id", formRandom = ~ht, distr = "norm")
R> mod_norm

Linear mixed models with distribution norm and dependence structure UNC
Log-likelihood value at convergence: 761.1776
Distribution norm
Fixed: fev1 ~ baseage + age + agep + baseht + ht + htp + age * ht
(Intercept)     baseage         age        agep      baseht          ht
-0.72482490 -0.06374243 -0.11073866 -0.08937323  1.07280577  0.66645149
        htp      age:ht
 1.62601051  0.13427577
Random effects:
  Formula: ~ ht by id
  Structure: General positive-definite
  Estimated variance (D):
            (Intercept)         ht
(Intercept)   0.5118849 -0.4382644
ht           -0.4382644  0.3809793
Error dependence structure: UNC
  Estimate(s):
    sigma2
0.01713089
```

```
Number of observations: 1946
Number of groups: 252
```

On the other hand, we can fit a more flexible model that accounts for outliers, serial correlation, and a possible skewness by considering

```
R> mod_scn <- smsn.lmm(
+    fev1 ~ baseage + age + agep + baseht + ht + htp + age * ht,
+    data = ds, groupVar = "id", formRandom = ~ ht, distr = "scn",
+    timeVar = "time", depStruct = "DEC")
R> mod_scn

Linear mixed models with distribution scn and dependence structure DEC
Log-likelihood value at convergence: 849.777
Distribution scn with nu = 0.1795557 0.3312515
Fixed: fev1 ~ baseage + age + agep + baseht + ht + htp + age * ht
(Intercept)     baseage         age        agep      baseht          ht
-0.84331885 -0.07358645 -0.10396804 -0.10018997  1.30824441  0.52561350
        htp      age:ht
 1.51464279  0.13597547
Random effects:
  Formula: ~ ht by id
  Structure: General positive-definite
  Estimated variance (D):
            (Intercept)         ht
(Intercept)   0.3303920 -0.3003916
ht           -0.3003916  0.2736832
  Skewness parameter: -0.11204 0.3416233
Error dependence structure: DEC
  Estimate(s):
    sigma2       phi1       phi2
0.01490199 0.33186733 0.75590999
Number of observations: 1946
Number of groups: 252
```

Comparing both models using an LR test, we can see that the DEC-SCN-LMM seems to provide a significantly improved fit:

```
R> lr.test(mod_norm, mod_scn)

Model selection criteria:
          logLik       AIC       BIC
mod_norm 761.178 -1498.355 -1431.473
mod_scn  849.777 -1663.554 -1563.230

    Likelihood-ratio Test
```

```
chi-square statistics =  177.1989
df =  6
p-value =  1.334695e-35

The null hypothesis that both models represent the
data equally well is rejected at level  0.05
```

To evaluate if a skewed model is necessary, we can consider its symmetric version by fitting

```
R> mod_cn <- update(mod_norm, distr = "cn", timeVar = "time",
+    depStruct = "DEC")
R> lr.test(mod_cn, mod_scn)

Model selection criteria:
         logLik       AIC       BIC
mod_cn  849.412 -1666.824 -1577.648
mod_scn 849.777 -1663.554 -1563.230

    Likelihood-ratio Test

chi-square statistics =  0.7297971
df =  2
p-value =  0.6942671

The null hypothesis that both models represent the
data equally well is not rejected at level  0.05
```

Since the gain in considering the skewed model is not substantial, for parsimony we continue the analysis using the symmetric model.

To verify if a simpler model would be enough, we consider and compare the following possibilities:

```
R> mod_normCAR1 <- update(mod_cn, distr = "norm", depStruct = "CAR1")
R> mod_normDEC <- update(mod_cn, distr = "norm")
R> mod_cnCAR1 <- update(mod_cn, depStruct = "CAR1")
R> mod_cnUNC <- update(mod_cn, depStruct = "UNC")
R> criteria(list(
+    `UNC-N-LMM`  = mod_norm,
+    `CAR-N-LMM`  = mod_normCAR1,
+    `DEC-N-LMM`  = mod_normDEC,
+    `UNC-CN-LMM` = mod_cnUNC,
+    `CAR-CN-LMM` = mod_cnCAR1,
+    `DEC-CN-LMM` = mod_cn))

             logLik npar       AIC       BIC
UNC-N-LMM  761.1776   12 -1498.355 -1431.473
CAR-N-LMM  797.1737   13 -1568.347 -1495.892
```

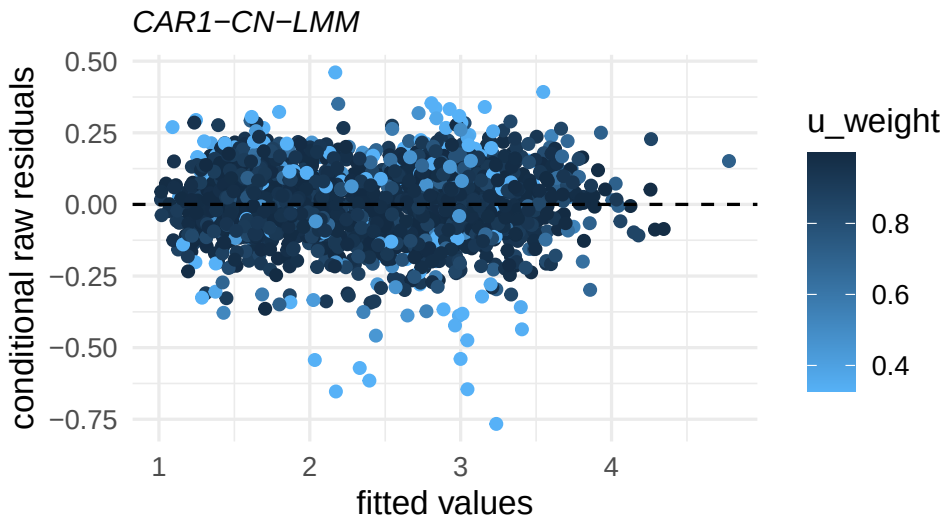

Figure 8: Six Cities study. Plotting the fitted object.

```
DEC-N-LMM  800.0750   14 -1572.150 -1494.120
UNC-CN-LMM 806.9553   14 -1585.911 -1507.881
CAR-CN-LMM 848.0047   15 -1666.009 -1582.407
DEC-CN-LMM 849.4121   16 -1666.824 -1577.648
```

From the criteria comparison, we can see that the CAR-CN-LMM provides a good trade-off between goodness of fit and the number of parameters. Therefore, we proceed with the analysis based on this model.

We can plot the model fit using

```
R> mod_cnCAR1 %>% plot()
```

And the associated Mahalanobis distance can be analyzed by running

```
R> mahalDist(mod_cnCAR1) %>% plot(nlabels = 0)
```

Finally, results can be extracted using

```
R> summary(mod_cnCAR1)

Linear mixed models with distribution cn and dependence structure CAR1
Call:
smn.lmm(data = ds, formFixed = fev1 ~ baseage + age + agep +
    baseht + ht + htp + age * ht, groupVar = "id", formRandom = ~ht,
    depStruct = "CAR1", timeVar = "time", distr = "cn")

Distribution cn with nu = 0.1744197 0.3265998

Random effects:
  Formula: ~ht
```

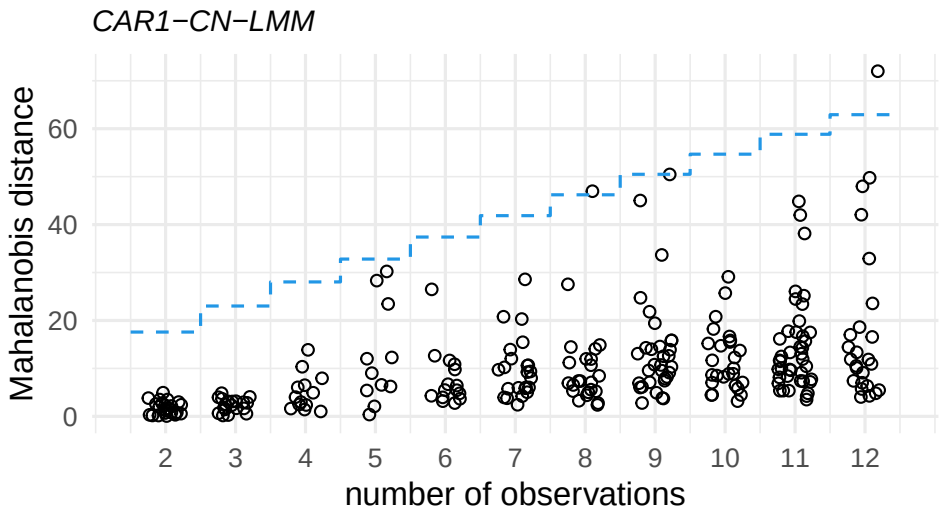

Figure 9: Six Cities study. Mahalanobis distance per number of repeated measures.

```
  Structure:
  Estimated variance (D):
            (Intercept)         ht
(Intercept)   0.3128866 -0.2850375
ht           -0.2850375  0.2600707

Fixed effects: fev1 ~ baseage + age + agep + baseht + ht + htp + age * ht
with approximate confidence intervals
                   Value  Std.error CI 95% lower CI 95% upper
(Intercept) -0.84975405 0.36661933  -1.56831474  -0.13119336
baseage     -0.07380252 0.01894638  -0.11093675  -0.03666829
age         -0.10438378 0.04216784  -0.18703124  -0.02173633
agep        -0.09893130 0.01050317  -0.11951714  -0.07834546
baseht       1.30213643 0.25894685   0.79460993   1.80966294
ht           0.54151620 0.23804335   0.07495981   1.00807258
htp          1.50424996 0.25846691   0.99766412   2.01083580
age:ht       0.13567868 0.02695395   0.08284991   0.18850746

Dependence structure: CAR1
  Estimate(s):
    sigma2       phi1
0.01451772 0.31110577

Model selection criteria:
  logLik       AIC       BIC
 848.005 -1666.009 -1582.407

Number of observations: 1946
Number of groups: 252
```

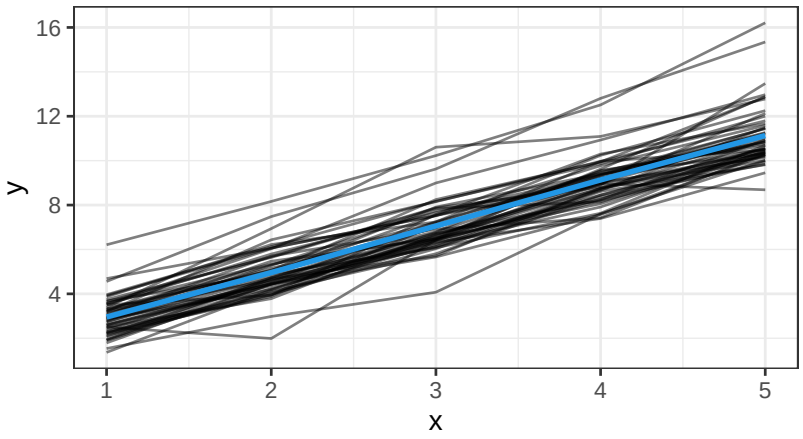

Figure 10: Synthetic data for $n = 50$ subjects. Trajectories plot with superimposed mean curve.

From $\widehat{\boldsymbol{\nu}} = (0.1744, 0.3266)$, we can interpret that the model identified about 17% of outliers, which are highlighted as lighter blue dots in Figure 8. Since the data are in their original scale, the estimated parameters can be straightforward interpreted.

5.3. Example with synthetic data

Here, we illustrate two functionalities of the package: its ability to simulate data using the function `rsmsn.lmm`, and the model performance for larger data sets. To generate data for 50 subjects with 5 time points each from a AR(1)-ST-LMM, we can use the following syntax:

```
R> nj1 <- 5
R> m <- 50
R> gendatList <- map(.x = rep(nj1, m), .f = function(nj)
+    rsmsn.lmm(time1 = 1:nj, x1 = cbind(1, 1:nj), z1 = rep(1, nj),
+      beta = c(1, 2), sigma2 = 0.25, D1 = 0.5 * diag(1), lambda = 2,
+      depStruct = "ARp", phi = 0.5, distr = "st", nu = 5))
R> gendat_50 <- bind_rows(gendatList, .id = "ind")
```

The simulated data behavior is illustrated in Figure 10.

Fitting the correct model to this synthetic data set,

```
R> fm1 <- smsn.lmm(y ~ x, data = gendat_50, groupVar = "ind",
+    depStruct = "ARp", pAR = 1, distr = "st")
```

is completed in `fm1$elapsedTime` = 39.446 seconds using a Windows 11 environment on a laptop with an 12th generation Intel Core i7 processor with 16 GB of RAM.

As a comparison, we can compute CIs using both the available methods as follows:

```
R> confint(fm1, method = "asymptotic") %>% round(digits = 3)
```

```
            Estimate CI 95% lower CI 95% upper
(Intercept)    1.028        0.672        1.384
x              2.016        1.956        2.075
sigma2         0.220        0.152        0.287
phiAR1         0.421        0.185        0.656
Dsqrt11        0.944        0.554        1.334
lambda1       12.625           NA           NA
nu1            4.001           NA           NA
```

```
R> confint(fm1, method = "bootstrap") %>% round(digits = 3)
```

```
            Estimate  2.5%  97.5%
(Intercept)    1.028 0.776  1.311
x              2.016 1.968  2.076
sigma2         0.220 0.158  0.285
phiAR1         0.421 0.184  0.596
Dsqrt11        0.944 0.631  1.289
lambda1       12.625 2.008 22.468
nu1            4.001 2.728  9.044
```

Now, to illustrate the model usability for larger samples sizes, we will consider a setting with 1,000 subjects and 10 time points each (10,000 observations total):

```
R> nj1 <- 10
R> m <- 1000
R> gendatList <- map(.x = rep(nj1, m), .f = function(nj)
+    rsmsn.lmm(time1 = 1:nj, x1 = cbind(1, 1:nj), z1 = rep(1, nj),
+      beta = c(1, 2), sigma2 = 0.25, D1 = 0.5 * diag(1), lambda = 2,
+      depStruct = "ARp", phi = 0.5, distr = "st", nu = 5))
R> gendat_1000 <- bind_rows(gendatList, .id = "ind")
```

Fitting the correct AR(1)-ST-LMM to this synthetic data set, using

```
R> fm2 <- smsn.lmm(y ~ x, data=gendat_1000, groupVar = "ind",
+    depStruct = "ARp", pAR = 1, distr = "st")
```

takes `fm2$elapsedTime` = 217.929 seconds using a Windows 11 environment on a laptop with an 12th generation Intel Core i7 processor with 16 GB of RAM. For this example, the asymptotic CIs are expected to provide reasonable estimates:

```
R> confint(fm2) %>% round(digits = 3)
```

```
            Estimate CI 95% lower CI 95% upper
(Intercept)    0.973        0.922        1.025
x              2.001        1.996        2.007
sigma2         0.243        0.230        0.256
phiAR1         0.492        0.465        0.518
Dsqrt11        0.648        0.583        0.714
lambda1        2.030           NA           NA
nu1            4.639           NA           NA
```

6. Extensions for censored data

Extending the model to accommodate censored data is crucial for dealing with studies where the outcome is subjected to limits of detection. The package **skewlmm** is being extended to allow data fit with such characteristics. Currently, implementation considering both the multivariate normal and t distribution, as proposed in Matos, Prates, Chen, and Lachos (2013), are available. In the future, we expect to extend the censored methods to accommodate other distributions from the SMN class, and further to account for skewed distributions.

In this context, the function `smn.clmm()` fits left, right, or interval-censored linear mixed models with possible within-subject dependence structures using the EM algorithm. It provides estimates and standard errors of parameters for these models.

The syntax is as follows:

```
smn.clmm(data, formFixed, groupVar, formRandom = ~1, depStruct = "UNC",
  ci, lcl, ucl, timeVar = NULL, distr = "norm", nufix = FALSE, pAR = 1,
  control = lmmControl())
```

where, in addition to the notation introduced earlier, we have:

- `ci` is a character containing the name of the censoring indicator variable in data, which should be 1 if the respective observation is censored or missing, and 0 otherwise. If missing, it is assumed that none of the observations is censored.
- `lcl` is a character containing the name of the lower censoring limit in data. If missing, it is assumed `lcl = -Inf`, i.e., no left limit.
- `ucl` is a character containing the name of the upper censoring limit in data. If missing, it is assumed `ucl = Inf`, i.e., no right limit.
- `nufix` is `TRUE` or `FALSE`, indicating if ν should be estimated for t distribution. If `nufix = TRUE`, ν must be specified through `lmmControl()`.

7. Concluding remarks

This paper discussed the estimation and evaluation of SMSN-LMM using the R package **skewlmm**, specifying the model definition, enlightening its distributional and structural options, and providing an examples of usage for real data sets, illustrating the theoretical development from Schumacher *et al.* (2021).

The **skewlmm** package aims to provide a user-friendly tool to fit robust LMM to longitudinal data, complementing standard tools as the **nlme** and **lme4** R packages by the use of a more flexible distributional assumption. Moreover, the availability of several optimization functions, such as the DAAREM algorithm and parallel optimization, is clearly an appealing strength of our new package. The implemented functions are simple to use and in accordance with traditional R packages. We hope that this package can be helpful for practitioners in several areas where LMMs are applicable.

It is worth noting that the current implementation does not support more than one clustering level – see Maullin-Sapey and Nichols (2021), and therefore, the package is not suitable for

crossed-factors LMM. Additionally, direct estimation of non-linear models or non-parametric estimation of smooth regression, such as the generalized additive models (Hastie and Tibshirani 1990),is not yet supported.

Recently, Lachos, Galea, Zeller, and Prates (2023) have proposed an interesting EM algorithm for LMM considering the family of generalized hyperbolic (GH) distributions, which is defined as the normal variance-mean mixture where the mixing distribution is the generalized inverse Gaussian (GIG) distribution and it has a convenient stochastic representation for implementation of the EM algorithm, leading to efficient ML estimation of the parameters. In the future, we plan to include the GH distribution.

Acknowlegments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001 and by the Conselho Nacional de Desenvolvimento Científico e Tecnológico – Brasil (CNPq). Larissa A. Matos acknowledges support from CNPq-Brazil (Grant 307105/2022-9) and from FAPESP-Brazil (Grant 2020/16713-0). Victor Lachos acknowledges the partial financial support from UConn – CLAS's Summer Research Funding Initiative 2023 and Research Excellence Program – UConn.

References

Anderson DG (1965). "Iterative Procedures for Nonlinear Integral Equations." *Journal of the ACM*, **12**(4), 547–560. doi:10.1145/321296.321305.

Arellano-Valle RB, Bolfarine H, Lachos VH (2005). "Skew-Normal Linear Mixed Models." *Journal of Data Science*, **3**, 415–438. doi:10.6339/jds.2005.03(4).238.

Asar Ö, Bolin D, Diggle PJ, Wallin J (2020). "Linear Mixed Effects Models for Non-Gaussian Continuous Repeated Measurement Data." *Journal of the Royal Statistical Society C*, **69**(5), 1015–1065. doi:10.1111/rssc.12405.

Azzalini A, Capitanio A (2003). "Distributions Generated by Perturbation of Symmetry with Emphasis on a Multivariate Skew t-Distribution." *Journal of the Royal Statistical Society B*, **65**(2), 367–389. doi:10.1111/1467-9868.00391.

Azzalini A, Dalla Valle A (1996). "The Multivariate Skew-Normal Distribution." *Biometrika*, **83**(4), 715–726. doi:10.1093/biomet/83.4.715.

Azzalini A, Genton M (2008). "Robust Likelihood Methods Based on the Skew-t and Related Distributions." *International Statistical Review*, **76**, 1490–1507. doi:10.1111/j.1751-5823.2007.00016.x.

Bandyopadhyay D, Lachos VH, Abanto-Valle CA, Ghosh P (2010). "Linear Mixed Models for Skew-Normal/Independent Bivariate Responses with an Application to Periodontal Disease." *Statistics in Medicine*, **29**(25), 2643–2655. doi:10.1002/sim.4031.

Bates D, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using **lme4**." *Journal of Statistical Software*, **67**(1), 1–48. doi:10.18637/jss.v067.i01.

Box GEP, Jenkins GM (1976). *Time Series Analysis: Forecasting and Control*. Holden-Day, San Francisco.

Branco MD, Dey DK (2001). "A General Class of Multivariate Skew-Elliptical Distributions." *Journal of Multivariate Analysis*, **79**, 99–113. doi:10.1006/jmva.2000.1960.

Bürkner PC (2017). "**brms**: An R Package for Bayesian Multilevel Models Using Stan." *Journal of Statistical Software*, **80**(1), 1–28. doi:10.18637/jss.v080.i01.

Cavus M, Yazici B (2025). "**glme**: An R Package for Mixed Effects Model Inference by the Generalized Approach." *Communications in Statistics – Simulation and Computation*, **54**(5), 1423–1437. doi:10.1080/03610918.2023.2286216.

Dockery DW, Berkey CS, Ware JH, Speizer FE, Ferris Jr BG (1983). "Distribution of Forced Vital Capacity and Forced Expiratory Volume in One Second in Children 6 to 11 Years of Age." *American Review of Respiratory Disease*, **128**(3), 405–412. doi:10.1164/arrd.1983.128.3.405.

Drikvandi R, Verbeke G, Molenberghs G (2017). "Diagnosing Misspecification of the Random-Effects Distribution in Mixed Models." *Biometrics*, **73**(1), 63–71. doi:10.1111/biom.12551.

Fitzmaurice GM, Laird NM, Ware JH (2012). *Applied Longitudinal Analysis*, volume 998. John Wiley & Sons.

Geraci M (2014). "Linear Quantile Mixed Models: The **lqmm** Package for Laplace Quantile Regression." *Journal of Statistical Software*, **57**(13), 1–29. doi:10.18637/jss.v057.i13.

Geraci M (2023). ***nlmm**: Generalized Laplace Mixed-Effects Models*. doi:10.32614/CRAN.package.nlmm. R package version 1.1.0.

Geraci M, Farcomeni A (2020). "A Family of Linear Mixed-Effects Models Using the Generalized Laplace Distribution." *Statistical Methods in Medical Research*, **29**(9), 2665–2682. doi:10.1177/0962280220903763.

Gerber F, Furrer R (2019). "**optimParallel**: An R Package Providing a Parallel Version of the L-BFGS-B Optimization Method." *The R Journal*, **11**(1), 352–358. doi:10.32614/rj-2019-030.

Gong M, Mao Z, Zhang D, Ren J, Zuo S (2023). "Study on Bayesian Skew-Normal Linear Mixed Model and Its Application in Fire Insurance." *Fire Technology*, **59**(5), 2455–2480. doi:10.1007/s10694-023-01436-1.

Hastie TJ, Tibshirani RJ (1990). *Generalized Additive Models*, volume 43. Chapman and Hall.

Healy MJR (1968). "Multivariate Normal Plotting." *Journal of the Royal Statistical Society C*, **17**(2), 157–161. doi:10.2307/2985678.

Henderson N, Varadhan R (2019). "Damped Anderson Acceleration with Restarts and Monotonicity Control for Accelerating EM and EM-Like Algorithms." *Journal of Computational and Graphical Statistics*, **28**(4), 834–846. doi:10.1080/10618600.2019.1594835.

Henderson N, Varadhan R (2020). ***daarem**: Damped Anderson Acceleration with Epsilon Monotonicity for Accelerating EM-Like Monotone Algorithms*. doi:10.32614/CRAN.package.daarem. R package version 0.5.

Ho HJ, Lin TI (2010). "Robust Linear Mixed Models Using the Skew t Distribution with Application to Schizophrenia Data." *Biometrical Journal*, **52**(4), 449–469. doi:10.1002/bimj.200900184.

Koller M (2016). "**robustlmm**: An R Package for Robust Estimation of Linear Mixed-Effects Models." *Journal of Statistical Software*, **75**(1), 1–24. doi:10.18637/jss.v075.i06.

Lachos VH, Galea M, Zeller C, Prates MO (2023). "Likelihood-Based Inference for Linear Mixed-Effects Models Using the Generalized Hyperbolic Distribution." *Stat*, **12**(1), e602. doi:10.1002/sta4.602.

Lachos VH, Ghosh P, Arellano-Valle RB (2010). "Likelihood Based Inference for Skew-Normal Independent Linear Mixed Models." *Statistica Sinica*, **20**, 303–322.

Lange KL, Sinsheimer JS (1993). "Normal/Independent Distributions and Their Applications in Robust Regression." *Journal of Computational and Graphical Statistics*, **2**, 175–198. doi:10.1080/10618600.1993.10474606.

Matos LA, Prates MO, Chen MH, Lachos VH (2013). "Likelihood-Based Inference for Mixed-Effects Models with Censored Response Using the Multivariate-t Distribution." *Statistica Sinica*, **23**, 1323–1345. doi:10.5705/ss.2012.043.

Maullin-Sapey T, Nichols TE (2021). "Fisher Scoring for Crossed Factor Linear Mixed Models." *Statistics and Computing*, **31**(5), 1–25. doi:10.1007/s11222-021-10026-6.

Muñoz A, Carey V, Schouten JP, Segal M, Rosner B (1992). "A Parametric Family of Correlation Structures for the Analysis of Longitudinal Data." *Biometrics*, **48**, 733–742. doi:10.2307/2532340.

Osorio F (2019). ***heavy**: Robust Estimation Using Heavy-Tailed Distributions*. doi:10.32614/CRAN.package.heavy. R package version 0.38.196.

Pinheiro J, Bates D (2000). *Mixed-Effects Models in S and S-PLUS*. Springer-Verlag, New York.

Pinheiro J, Bates D, DebRoy S, Sarkar D, R Core Team (2021). ***nlme**: Linear and Nonlinear Mixed Effects Models*. doi:10.32614/CRAN.package.nlme. R package version 3.1-152.

Pinheiro JC, Liu CH, Wu YN (2001). "Efficient Algorithms for Robust Estimation in Linear Mixed-Effects Models Using a Multivariate t-Distribution." *Journal of Computational and Graphical Statistics*, **10**, 249–276. doi:10.1198/10618600152628059.

Pratapa DP, Suryanarayana D (2015). "Restarted Pulay Mixing for Efficient and Robust Acceleration of Fixed-Point Iterations." *Chemical Physics Letters*, **635**, 69–74. doi:10.1016/j.cplett.2015.06.029.

Schumacher FL (2021). *Robust Linear Mixed Models for Longitudinal Data Using Skewed and Heavy-Tailed Distributions*. Ph.D. thesis, Universidade Estadual de Campinas.

Schumacher FL, Lachos VH, Matos LA (2021). "Scale Mixture of Skew-Normal Linear Mixed Models with Within-Subject Serial Dependence." *Statistics in Medicine*, **40**(7), 1790–1810. doi:10.1002/sim.8870.

Stasinopoulos DM, Rigby RA (2007). "Generalized Additive Models for Location Scale and Shape (GAMLSS) in R." *Journal of Statistical Software*, **23**(7), 1–46. doi:10.18637/jss.v023.i07.

Tamási B, Hothorn T (2021). "**tramME**: Mixed-Effects Transformation Models Using Template Model Builder." *The R Journal*, **13**(2), 398–418. doi:10.32614/rj-2021-075.

Umlauf N, Klein N, Simon T, Zeileis A (2021). "**bamlss**: A Lego Toolbox for Flexible Bayesian Regression (and Beyond)." *Journal of Statistical Software*, **100**(4), 1–53. doi:10.18637/jss.v100.i04.

Varadhan R, Roland C (2008). "Simple and Globally Convergent Methods for Accelerating the Convergence of Any EM Algorithm." *Scandinavian Journal of Statistics*, **35**(2), 335–353. doi:10.1111/j.1467-9469.2007.00585.x.

Weerahandi S, Yu CR (2020). "Exact Distributions of Statistics for Making Inferences on Mixed Models under the Default Covariance Structure." *Journal of Statistical Distributions and Applications*, **7**(1). doi:10.1186/s40488-020-00105-w.

Weiss RE (2005). *Modeling Longitudinal Data*, volume 1. Springer-Verlag, New York. doi:10.1007/0-387-28314-5.

Zeller CB, Labra FV, Lachos VH, Balakrishnan N (2010). "Influence Analyses of Skew-Normal/Independent Linear Mixed Models." *Computational Statistics & Data Analysis*, **54**(5), 1266–1280. doi:10.1016/j.csda.2009.11.008.

Affiliation:

Fernanda L. Schumacher
The Ohio State University
Division of Biostatistics
Columbus, OH, United States of America
E-mail: schumacher.313@osu.edu
URL: https://github.com/fernandalschumacher

Larissa A. Matos
Universidade Estadual de Campinas
Department of Statistics
Campinas, SP, Brazil
E-mail: larissam@unicamp.br
URL: https://ime.unicamp.br/~larissam/

Victor H. Lachos
University of Connecticut
Department of Statistics
Storrs, CT, United States of America
E-mail: hlachos@uconn.edu
URL: https://hlachos.stat.uconn.edu/

Journal of Statistical Software https://www.jstatsoft.org/
published by the Foundation for Open Access Statistics https://www.foastat.org/
December 2025, Volume 115, Issue 7 *Submitted:* 2023-11-01
doi:10.18637/jss.v115.i07 *Accepted:* 2025-02-07