Capabilities of R package mixAK for Clustering Based on Multivariate Continuous and Discrete Longitudinal Data

Arnošt Komárek
Faculty of Mathematics and Physics, Charles University in Prague

---

Abstract

This document supplements the paper (Komárek and Komárková 2011a) and shows the R analysis of the data from a Mayo Clinic trial on 312 patients with primary biliary cirrhosis.

Keywords: cluster analysis, generalized linear mixed model, functional data, multivariate longitudinal data, R package.

This document was built on **December 20, 2011**.

1. Introduction

Multiple outcomes, both continuous and discrete are routinely gathered on subjects in longitudinal studies. In another paper (Komárek and Komárková 2011a), we described a model-based statistical method for clustering (classification) of subjects into groups with apriori unknown characteristics on basis of repeated measurements of recorded values of longitudinal outcomes. The methodology is based on modelling the evolution of each longitudinal outcome using the classical generalized linear mixed model (GLMM) where we capture possible dependence between the values of different outcomes by specifying a joint distribution of all random effects involved in the GLMM for each response. The basis for subsequent clustering is provided by assuming a heteroscedastic mixture of multivariate normal distributions in the random effects distribution where each mixture component corresponds to one cluster in subsequent classification. Mainly for computational reasons, the inference is based on a Bayesian specification of the model and simulation based Markov chain Monte Carlo (MCMC) methodology.

To allow for practical usage of the proposed methods, we considerably extended the **R** (R Development Core Team 2011) package **mixAK** (Komárek 2009) which is available from the Comprehensive **R** Archive Network at `http://CRAN.R-project.org/package=mixAK`. The current paper builds upon previous methodological work Komárek and Komárková (2011a) and its web supplement Komárek and Komárková (2011b) and primarily supplements standard help pages of functions from the package **mixAK** related to the methods described in Komárek and Komárková (2011a,b) by a step-by-step **R** analysis of one of the examples shown therein. Consequently, only a moderate modification of the code described in this paper should suffice to analyze similar datasets.

We conclude Introduction by pointing out that the package **mixAK** exploits some routines from the following **R** contributed packages: **colorspace** (Ihaka *et al.* 2009), **lme4** (Bates *et al.* 2011), **mnormt** (Genz and Azzalini 2011), **coda** (Plummer *et al.* 2006), **snow** (Tierney *et al.* 2011), **snowfall** (Knaus 2010), which must be installed prior to or together with the package **mixAK**.

2. Data

Clustering capabilities of the **mixAK** package will be illustrated on the analysis of the data from a Mayo Clinic trial on 312 patients with primary biliary cirrhosis (PBC) conducted in 1974–1984 (Dickson *et al.* 1989). The data are available at `http://lib.stat.cmu.edu/datasets/pbcseq` and as `PBCseq` also inside the **mixAK** package.

```
R> library("mixAK")
R> data(PBCseq, package="mixAK")
```

It is a longitudinal dataset with one row per visit. There are 1 to 16 visits per patient (modus 4, median 5) and a median follow-up of 2 300 days. At each visit, measurements of several disease related markers were taken and are recorded in corresponding columns of the `data.frame` `PBCseq`. On top of that, information concerning the disease progression free survival status where disease progression is defined as either death related to PBC or liver transplantation is available. In our illustration, we will try to find groups of similar patients with respect to the longitudinal evolution of three markers:

1. logarithmic serum bilirubin (variable `lbili`);
2. platelet counts (variable `platelet`);
3. dichotomous presence of blood vessel malformations (variable `spiders`).

Due to the fact that some or all of these markers are strongly related to the disease progression we included in the analysis only those patients who were known to be alive and without liver transplantation at a pre-specified time point of 910 days of follow-up where 910 days were chosen to be able to compare our results to another analysis of the same dataset (Müller 2005). Further, only longitudinal measurements obtained by that time point will enter our clustering procedure. This mimics a situation from clinical practice when one tries to identify groups of similar individuals in a cohort of patients using all information gathered by a specific moment. Interestingly, it will be shown that found groups are indeed strongly related to the residual progression free survival time beyond 910 days and hence our clutering methodology could also be viewed as a clinical diagnostic procedure.

The following code employs the variable `alive` (number of days that patient is known to be alive) and creates a `data.frame` which contains only subset of patients known to be alive at 910 days of follow-up, longitudinal measurements by that time point and variables: `id` (identification number of patient), `day`, `months` (time from start of follow-up in days and months, respectively), `fu.days` (total number of follow-up days), `delta.ltx.death` (dichotomous variable being equal to 0 if the patient was alive and without liver transplantation at time of `fu.days` and being equal to 1 if the patient died due to PBC related complications or had to undergo liver transplantation at time of `fu.days`), `lbili`, `platelet`, `spiders` (values of longitudinal markers which we will use for clustering).

```
R> idTake <- subset(PBCseq, day == 0 & alive >= 910)[, "id"]
R> pbc01 <- subset(PBCseq, id %in% idTake & day <= 910,
+      select=c("id", "day", "month", "fu.days", "delta.ltx.death",
+               "lbili", "platelet", "spiders"))
R> rownames(pbc01) <- 1:nrow(pbc01)
R> head(pbc01)
```

```
  id day     month fu.days delta.ltx.death       lbili platelet spiders
1  2   0  0.000000    5169               0  0.09531018      221       1
2  2 182  5.979466    5169               0 -0.22314355      188       1
3  2 365 11.991786    5169               0  0.00000000      161       1
4  2 768 25.232033    5169               0  0.64185389      122       1
5  3   0  0.000000    1012               1  0.33647224      151       0
6  3 176  5.782341    1012               1  0.09531018      160       1
```

```
R> tail(pbc01)
```

```
     id day     month fu.days delta.ltx.death     lbili platelet spiders
913 311 187  6.143737    1508               0 0.4054651      382      NA
914 311 397 13.043121    1508               0 0.6418539      408       0
915 312   0  0.000000    1457               0 1.8562980      200       1
916 312 206  6.767967    1457               0 1.7047481      189       0
917 312 390 12.813142    1457               0 2.0014800      148       0
918 312 775 25.462012    1457               0 2.7911651      138       1
```

The resulting `data.frame` `pbc01` contains data from $N = 260$ patients and 1 to 5 (modus and median 4) observations of each marker per patient.

3. Descriptive analysis

Standard capabilities of a variety of **R** packages can be used to perform a descriptive analysis of the longitudinal data at hand. On top of that, two procedures are available in package **mixAK** to extract and plot longitudinal profiles of considered markers of individual patients. They include the following functions:

- `getProfiles()` which creates a list of `data.frames` (one `data.frame` per patient) with selected variables;
- `plotProfiles()` which creates a spaghetti graph with observed longitudinal profiles per patient.

We extract the longitudinal profiles of variables `lbili`, `platelet`, `spiders` from the `data.frame` `pbc01` where patients are identified by a variable `id` and the time is given by a variable `month`. Subsequently, we print the data for the first patient in the dataset.

```
R> ip <- getProfiles(t = "month",
+      y = c("lbili", "platelet", "spiders"), id = "id", data = pbc01)
R> print(ip[[1]])
```

```
     month       lbili platelet spiders
1 0.000000  0.09531018      221       1
2 5.979466 -0.22314355      188       1
```

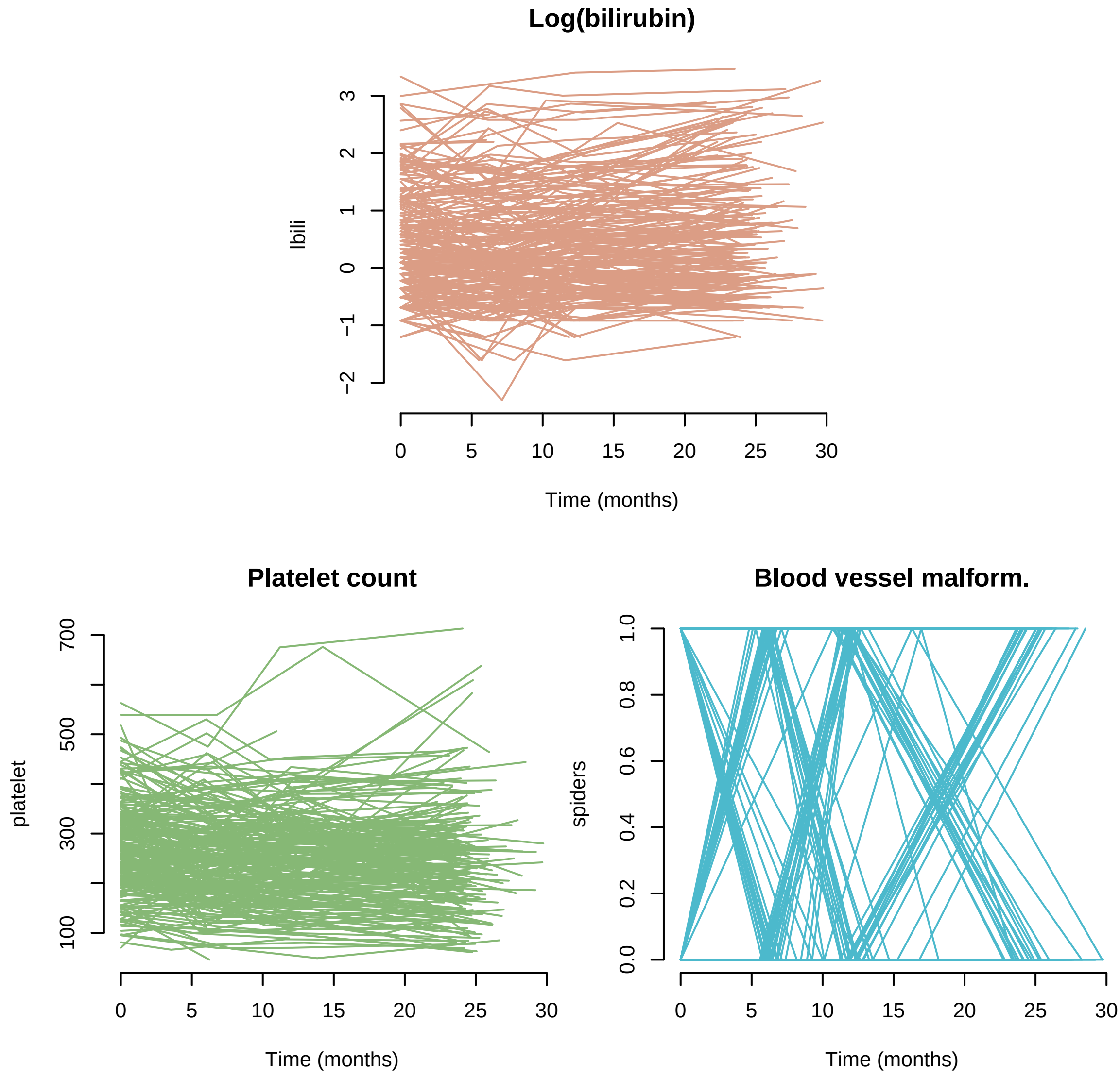

Figure 1: Observed longitudinal profiles of considered markers.

```
3 11.991786  0.00000000      161        1
4 25.232033  0.64185389      122        1
```

We use colors from the Hue-Chroma-Luminance (HCL) based palette (Zeileis *et al.* 2009) and draw gradually the spaghetti graphs of observed values of variables `lbili`, `platelet`, `spiders`, see Figure 1.

```
R> COL <- rainbow_hcl(3, start = 30, end = 210)
R> XLIM <- c(0, 910) / (365.25 / 12)
R> #
R> layout(autolayout(3))
R> plotProfiles(ip = ip, data = pbc01, var = "lbili", tvar = "month",
+     xlim = XLIM, xlab = "Time (months)", col = COL[1],
+     auto.layout = FALSE, main = "Log(bilirubin)")
R> plotProfiles(ip = ip, data = pbc01, var = "platelet", tvar = "month",
+     xlim = XLIM, xlab = "Time (months)", col = COL[2],
+     auto.layout = FALSE, main = "Platelet count")
R> plotProfiles(ip = ip, data = pbc01, var = "spiders",  tvar = "month",
+     xlim = XLIM, xlab = "Time (months)", col = COL[3],
+     auto.layout = FALSE, main = "Blood vessel malform.")
```

4. Model

It is now our intention to use all observed values of logarithmic bilirubin, platelet count and dichotomous presence of blood vessel malformations shown on Figure 1 to classify patients into a pre-specified number of groups (clusters). Model which serves as a basis for clustering procedure is described in full generality in Komárek and Komárková (2011a, Sec. 2). In this place, we concentrate on its specific form which will be used for our particular application. Let $\boldsymbol{Y}_{i,1} = (Y_{i,1,1}, \ldots, Y_{i,1,n_{i,1}})^\top$, $\boldsymbol{Y}_{i,2} = (Y_{i,2,1}, \ldots, Y_{i,2,n_{i,2}})^\top$, $\boldsymbol{Y}_{i,3} = (Y_{i,3,1}, \ldots, Y_{i,3,n_{i,3}})^\top$, $i = 1, \ldots, N$ denote random vectors leading to the longitudinal profiles of logarithmic bilirubin, platelet counts and dichotomous presence of blood vessel malformations, respectively of the ith patient. Further, let $\boldsymbol{Y}_i = (\boldsymbol{Y}_{i,1}^\top, \boldsymbol{Y}_{i,2}^\top, \boldsymbol{Y}_{i,3}^\top)^\top$ be a random vector of all longitudinal measurements on the ith patient, and finally, let $\boldsymbol{Y} = (\boldsymbol{Y}_1^\top, \ldots, \boldsymbol{Y}_N^\top)^\top$ be a random vector corresponding to observed values of all outcomes on all patients. The observed counterparts of corresponding upper case random variables and vectors will be denoted by lower case letters $y_{i,r,j}, \boldsymbol{y}_{i,r}, \boldsymbol{y}_i, \boldsymbol{y}$, $i = 1, \ldots, N$, $r = 1, 2, 3$, $j = 1, \ldots, n_{i,r}$. Note that double subscript in $n_{i,r}$ reflects the fact that not all considered longitudinal markers must be available at each patient's visit as it is, for example, the case for patient with id 311 at his/her visit at day 187 (see output from `tail(pbc01)` on page 3).

We start by modelling the evolution of each marker over time using a standard generalized linear mixed model (GLMM) where the mean values of components of $\boldsymbol{Y}_i$, $i = 1, \ldots, N$ are assumed to depend on known covariates (time $t_{i,r,j}$ from start of follow-up in months in which the value $y_{i,r,j}$ was recorded), unknown regression coefficient $\boldsymbol{\alpha} = \alpha_3$ (fixed effect) and also on patient specific random regression coefficients $\boldsymbol{b}_i = (\boldsymbol{b}_{i,1}^\top, \boldsymbol{b}_{i,2}^\top, b_{i,3}^\top)^\top$ (random effects), where $\boldsymbol{b}_{i,1} = (b_{i,1,1}, b_{i,1,2})^\top$, $\boldsymbol{b}_{i,2} = (b_{i,2,1}, b_{i,2,2})^\top$. Gaussian distribution with residual variance σ_1^2

is assumed for the logarithmic bilirubin ($Y_{i,1,j}$), Poisson distribution with the mean entering using a canonical log link function the corresponding GLMM is assumed for platelet counts ($Y_{i,2,j}$), Bernoulli distribution with the mean which enters using a canonical logit link function the corresponding GLMM is assumed for dichotomous presence of blood vessel malformations ($Y_{i,3,j}$). The mean structures of considered GLMM's are as follows

$$\left.\begin{aligned} \mathsf{E}(Y_{i,1,j}\,|\,\boldsymbol{b}_{i,1}) &= b_{i,1,1} + b_{i,1,2}\,t_{i,1,j},\\ \log\bigl\{\mathsf{E}(Y_{i,2,j}\,|\,\boldsymbol{b}_{i,2})\bigr\} &= b_{i,2,1} + b_{i,2,2}\,t_{i,2,j},\\ \mathrm{logit}\bigl\{\mathsf{P}(Y_{i,3,j}=1\,|\,b_{i,3},\,\alpha_3)\bigr\} &= b_{i,3} + \alpha_3\,t_{i,3,j}, \end{aligned}\right\} \qquad (1)$$

$i = 1,\ldots,N$, $j = 1,\ldots,n_{i,r}$, $r = 1,2,3$ and the vector of GLMM related unknown parameters is $\boldsymbol{\psi} = (\sigma_1^2,\,\alpha_3)^\top$. Further, let $\boldsymbol{B} = (\boldsymbol{b}_1,\ldots,\boldsymbol{b}_N)$ denote the matrix containing all subject specific random effect vectors in its columns.

The random effect vectors $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_N$ are assumed to be i.i.d. with a mixture density

$$p(\boldsymbol{b}_i\,|\,\boldsymbol{\theta}) = |\boldsymbol{S}|^{-1}\sum_{k=1}^{K} w_k\,\varphi\bigl(\boldsymbol{S}^{-1}(\boldsymbol{b}_i-\boldsymbol{s})\,\big|\,\boldsymbol{\mu}_k,\,\boldsymbol{D}_k\bigr) = \sum_{k=1}^{K} w_k\,\varphi(\boldsymbol{b}_i\,|\,\boldsymbol{s}+\boldsymbol{S}\boldsymbol{\mu}_k,\,\boldsymbol{S}\boldsymbol{D}_k\boldsymbol{S}^\top), \qquad (2)$$

$i = 1,\ldots,N$, where K is pre-specified number of mixture components ($K = 2$ will be used in our illustration), $\varphi(\cdot\,|\,\boldsymbol{\mu},\,\boldsymbol{D})$ is a density of the (multivariate) normal distribution with mean $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{D}$ and $\boldsymbol{\theta} = \bigl(\boldsymbol{w}^\top,\,\boldsymbol{\mu}_1^\top,\ldots,\boldsymbol{\mu}_K^\top,\,\mathsf{vec}(\boldsymbol{D}_1),\ldots,\mathsf{vec}(\boldsymbol{D}_K)\bigr)^\top$ is a vector of unknown mixture parameters with $\boldsymbol{w} = (w_1,\ldots,w_K)^\top$. Finally, $\boldsymbol{s}$ is a 5×1 fixed shift vector and $\boldsymbol{S}$ a 5×5 fixed diagonal scale matrix included in the model mainly due to possibility of improving numerical stability of the MCMC algorithm which is used for inference. Note that in subsequent clustering, each mixture component in expression (2) corresponds to one cluster and the mixture model (2) can also be specified hierarchically as

$$\left.\begin{array}{rcll} p(\boldsymbol{b}_i\,|\,\boldsymbol{\theta},\,u_i=k) & = & |\boldsymbol{S}|^{-1}\varphi\bigl(\boldsymbol{S}^{-1}(\boldsymbol{b}_i-\boldsymbol{s})\,\big|\,\boldsymbol{\mu}_k,\,\boldsymbol{D}_k\bigr), & i=1,\ldots,N,\\ \mathsf{P}(u_i=k\,|\,\boldsymbol{\theta}) & = & w_k, & i=1,\ldots,N,\ k=1,\ldots,K, \end{array}\right\} \qquad (3)$$

where $\boldsymbol{u} = (u_1,\ldots,u_N)^\top$ is a vector of latent component allocations.

Mainly for computational reasons, the model is specified also from a Bayesian point of view. The joint prior distribution for model parameters is specified in a hierarchical way using the following factorization

$$p(\boldsymbol{\psi},\,\boldsymbol{\theta},\,\boldsymbol{B},\,\boldsymbol{u}) = p(\boldsymbol{B}\,|\,\boldsymbol{\theta},\,\boldsymbol{u})\,\times\,p(\boldsymbol{u}\,|\,\boldsymbol{\theta})\,\times\,p(\boldsymbol{\theta})\,\times\,p(\boldsymbol{\psi}), \qquad (4)$$

where $p(\boldsymbol{B}\,|\,\boldsymbol{\theta},\,\boldsymbol{u})\,\times\,p(\boldsymbol{u}\,|\,\boldsymbol{\theta}) = \prod_{i=1}^{N}\Bigl\{|\boldsymbol{S}|^{-1}\varphi\bigl(\boldsymbol{S}^{-1}(\boldsymbol{b}_i-\boldsymbol{s})\,\big|\,\boldsymbol{\mu}_{u_i},\,\boldsymbol{D}_{u_i}\bigr)\,\times\,w_{u_i}\Bigr\}$ follows from hierarchically specified mixture model (3). Further, specification of the mixture related part $p(\boldsymbol{\theta})$ pursues the classical proposal of Richardson and Green (1997), and specification of the GLMM related part $p(\boldsymbol{\psi})$ follows classically used priors in this context, see Komárek and Komárková (2011b, Sec. A) for more details and also discussion on possible choices for additional hyperparameters to achieve weakly informative prior distribution.

5. Posterior Markov chain Monte Carlo simulation

The clustering and in general the whole inference is based on a sample from the posterior distribution $p(\boldsymbol{\psi},\, \boldsymbol{\theta},\, \boldsymbol{B},\, \boldsymbol{u} \,|\, \boldsymbol{y})$ which is derived using the Bayes theorem from the prior distribution (4) and the likelihood based on the multivariate GLMM model (1). We obtain the posterior sample

$$\mathcal{S}_M = \Bigl\{\bigl(\boldsymbol{\psi}^{(m)},\, \boldsymbol{\theta}^{(m)},\, \boldsymbol{B}^{(m)},\, \boldsymbol{u}^{(m)}\bigr) : m = 1, \ldots, M\Bigr\} \tag{5}$$

using the Markov chain Monte Carlo methodology, see Komárek and Komárková (2011b, Sec. B) for details. Further, it is necessary to point out that the posterior distribution is invariant towards $K!$ possible label switching of mixture components which is a crucial difficulty as our main goal is clustering which requires unique identification of mixture components. One possibility on how to resolve this complication is to use a suitable relabelling algorithm (see Stephens 2000) which we adapted to be used in a context of our model.

Package **mixAK** contains the following routines which are primarily related to sampling from the posterior distribution and whose use will be illustrated in this Section.

- `GLMM_MCMC()` is the main function which runs the MCMC algorithm, stores the sample $\mathcal{S}_M$ from the posterior distribution of model parameters and calculates basic posterior summary statistics. It returns a `list` which contains some information concerning the model and specific choices of hyperparameters of the prior distribution, the sample $\mathcal{S}_M$, and several other quantities calculated from these sampled values of model parameters. The class of the resulting object is set to `GLMM_MCMC`. Several methods which we shall introduce in a sequel are available for objects of this class to handle or visualize the results.
- `NMixRelabel()` is a generic function with a method for objects of class `GLMM_MCMC` which applies the relabelling algorithm and returns appropriately modified input object.

5.1. Running MCMC simulation

We run the MCMC algorithm for 1 000 burn-in and 10 000 subsequent iterations with 1:100 thinning to obtain a sample $\mathcal{S}_{10\,000}$ from the posterior distribution based on model (1). To calculate the penalized expected deviance (PED) which may be used for model comparison [SECTION ON IT TO BE ADDED], two MCMC chains are generated, by default sequentially. On multicore processors this task might be parallelized by setting the `parallel` argument to `TRUE`. Indicated computational time was achieved on Intel Core 2 Duo 3 GHz CPU with 3.25 GB RAM running on Linux Debian OS.

```
R> set.seed(20042007)
R> mod <- GLMM_MCMC(y = pbc01[, c("lbili", "platelet", "spiders")],
+       dist = c("gaussian", "poisson(log)", "binomial(logit)"),
+       id = pbc01[, "id"],
+       x = list(lbili    = "empty",
+                platelet = "empty",
+                spiders  = pbc01[, "month"]),
+       z = list(lbili    = pbc01[, "month"],
+                platelet = pbc01[, "month"],
+                 spiders   = "empty"),
+        random.intercept = rep(TRUE, 3),
+        prior.b = list(Kmax = 2),
+        nMCMC = c(burn = 1000, keep = 10000, thin = 100, info = 1000),
+        PED = TRUE, parallel = FALSE)


Chain number 1
==============
MCMC sampling started on Fri Dec 16 14:47:24 2011.
Burn-in iteration 1000
Iteration 11000
MCMC sampling finished on Fri Dec 16 15:59:15 2011.

Chain number 2
==============
MCMC sampling started on Fri Dec 16 15:59:16 2011.
Burn-in iteration 1000
Iteration 11000
MCMC sampling finished on Fri Dec 16 17:13:34 2011.

Computation of penalized expected deviance started on Fri Dec 16 17:13:53 2011.
Computation of penalized expected deviance finished on Fri Dec 16 17:15:48 2011.
```

Note that the `data.frame` specified in a `y` argument may contain missing values which is primarily useful to indicate that a value of a particular marker was not obtained at specific visit. The mean structure of a multivariate GLMM (1) is indicated by arguments `x` (fixed effects except intercept), `z` (random effects except intercept) and `random.intercept`. Note that we assume a hierarchically centered GLMM (Gelfand *et al.* 1995) where the population effect of covariates included in the definition of random effects is given by the mean of these random effects. Hence the marker specific parts of the `list`s in `x` and `z` arguments may not contain the same variables. The keyword `"empty"` is used to indicate that there are no fixed or random effects, respectively in a model for a specific marker. Finally, intercept is always included in the model and the fact whether it is random or fixed in indicated by argument `random.intercept`. The only obligatory part of the prior distribution which has to be specified by the user is the number of mixture components given as a `Kmax` component of the list in the `prior.b` argument. All other values of prior hyperparameters are automatically selected to achieve weakly informative prior distribution using the guidelines described in Komárek and Komárková (2011b, Sec. A). Furthermore, reasonable values of the shift vector $\boldsymbol{s}$, the scale matrix $\boldsymbol{S}$ and initial values of model parameters to start the MCMC were automatically chosen by the `GLMM_MCMC` routine as well. To achieve this, GLMM from each row of expression (1) with a classical one-component normal distribution assumed for random effects is separately estimated using the method of maximum-likelihood (ML) by the mean of `lmer` or `glmer` function from the **R** package **lme4**. In a sequel, let $\boldsymbol{\alpha}^0_{ML}$, $\boldsymbol{\beta}^0_{ML} = \bigl(\beta^0_{ML,1}, \dots, \beta^0_{ML,5}\bigr)^\top$, $\boldsymbol{d}^0_{ML} = \bigl(d_{ML,1}, \dots, d_{ML,5}\bigr)^\top$, $\sigma^0_{ML,1}$ be vectors of `lmer`/`glmer` ML estimates of fixed effects, means of random effects, standard deviations of random effects, and residual standard deviation (from a Gaussian model for logarithmic bilirubin), respectively. Further, let $\boldsymbol{B}^0_{ML}$ be a matrix with `lmer`/`glmer` based empirical Bayes estimates of random effects.

In majority of the reminder of this Section, we examine components of the resulting object related to the prior distribution and initial values for the MCMC simulation. Further, we show how to specify user-defined values of prior hyperparameters, shift vector $\boldsymbol{s}$, scale matrix $\boldsymbol{S}$ and initial values for all model parameters and random hyperparameters. The object `mod` is a `list` with two main components `mod[[1]]` and `mod[[2]]` holding the sampled chains, their initial values, posterior summary statistics based on that chain etc. and some additional components derived from both sampled chains. Objects `mod[[1]]` and `mod[[2]]` are again `list`s with informations pertaining to the first and second sampled chain, respectively.

5.2. Shift vector and scale matrix

First, we examine particular values of the shift vector $\boldsymbol{s}$ and a diagonal of the scale matrix $\boldsymbol{S}$ which, by default, are given as $\boldsymbol{s} = \boldsymbol{\beta}^0_{ML}$, $\boldsymbol{S} = \mathsf{diag}(\boldsymbol{d}^0_{ML})$ (these are the same for chain 1 and 2).

```
R> print(mod[[1]]$scale.b)


$shift
[1]  0.315158108  0.007654708  5.526209768 -0.006634000 -2.749539170

$scale
(Intercept)          z1 (Intercept)          z1 (Intercept) 
 0.86449212  0.02007624  0.34860152  0.01565365  3.22849129 
```

```
R> print(mod[[2]]$scale.b)


$shift
[1]  0.315158108  0.007654708  5.526209768 -0.006634000 -2.749539170

$scale
(Intercept)          z1 (Intercept)          z1 (Intercept) 
 0.86449212  0.02007624  0.34860152  0.01565365  3.22849129 
```

That is,

$$\boldsymbol{s} = \bigl(s_1, \ldots, s_5\bigr)^\top \doteq \bigl(0.315,\, 0.00765,\, 5.53,\, -0.00663,\, -2.75\bigr)^\top,$$
$$\boldsymbol{S} = \mathsf{diag}\bigl(S_1, \ldots, S_5\bigr) \doteq \mathsf{diag}\bigl(0.864,\, 0.02008,\, 0.35,\, 0.01565,\, 3.23\bigr).$$

The user is able to set his/her own values of the shift vector and a scale matrix by using the `scale.b` argument in the `GLMM_MCMC` function call. For example, setting $\boldsymbol{s}$ to vector of zeros and $\boldsymbol{S}$ to a unit matrix is achieved by

```
scale.b = list(shift = rep(0, 5),
               scale = rep(1, 5))
```

5.3. Prior distribution for mixture related parameters

We continue in exploration of the resulting object `mod` by checking the particular values of the hyperparameters of the prior distribution for mixture related parameter vector $\boldsymbol{\theta}$ which are stored in the `prior.b` component.

```
R> print(mod[[1]]$prior.b)
```

```
$Kmax                                   $priorK
[1] 2                                   [1] "fixed"

$priormuQ                               $lambda
[1] "independentC"                      [1] 0

$delta                                  $xi
[1] 1                                      m1 m2 m3 m4 m5
                                        j1  0  0  0  0  0
                                        j2  0  0  0  0  0


$ce                                     $D
c1 c2                                        m1 m2 m3 m4 m5
 0  0                                   j1.1 36  0  0  0  0
                                        j1.2  0 36  0  0  0
                                        j1.3  0  0 36  0  0
                                        j1.4  0  0  0 36  0
                                        j1.5  0  0  0  0 36
                                        j2.1 36  0  0  0  0
                                        j2.2  0 36  0  0  0
                                        j2.3  0  0 36  0  0
                                        j2.4  0  0  0 36  0
                                        j2.5  0  0  0  0 36

$zeta                                   $g
[1] 6                                   [1] 0.2 0.2 0.2 0.2 0.2

$h
(Intercept)          z1 (Intercept)          z1 (Intercept)
  0.2777778   0.2777778   0.2777778   0.2777778   0.2777778
```

The same would be seen if we print this for the second sampled chain.

```
R> print(mod[[2]]$prior.b)
```

Some parts of this list, namely `priorK`, `lambda`, and `ce` are redundant in this situation and are included in the object only for compatibility with other related functions from the **mixAK** package. Component `priormuQ` informs us that a semiconjugate independent Normal and Wishart prior, see Komárek and Komárková (2011b, Subsec. A.3) is assumed for mixture means and inverted covariance matrices. Alternatively, a natural-conjugate Normal-Wishart prior (see Komárek 2009, Sec. 2.2) can be considered. Component `delta` indicates a value of the parameter δ in the Dirichlet prior assumed for mixture weights. That is, in our application, apriori

$$(w_1,\, w_2) \sim \mathcal{D}(1, 1).$$

Further, the mixture means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are apriori normally distributed, i.e., $\boldsymbol{\mu}_k \sim \mathcal{N}(\boldsymbol{\xi}_b,\, \boldsymbol{C}_b)$, $k = 1, 2$. The value of $\boldsymbol{\xi}_b$ (repeated $K = 2$-times) is shown in the rows of matrix `xi`, and the value of $\boldsymbol{C}_b$ (again repeated $K = 2$-times), is shown in the blocks of matrix `D`, i.e.,

$$\boldsymbol{\mu}_k \sim \mathcal{N}\bigl((0, \dots, 0)^\top,\, \mathsf{diag}(36, \dots, 36)\bigr), \qquad k = 1, 2.$$

Finally, the inverted mixture covariance matrices $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ are apriori Wishart distributed, i.e., $\boldsymbol{D}_k \sim \mathcal{W}(\zeta_b,\, \Xi_b)$, where $\Xi_b = \mathsf{diag}(\gamma_{b,1}, \dots, \gamma_{b,5})$ and $\gamma_{b,l}^{-1} \sim \mathcal{G}(g_{b,l},\, h_{b,l})$, $l = 1, \dots, 5$. The value of ζ_b is reflected by the value of `zeta`, the values of $g_{b,l}$ and $h_{b,l}$ are shown by components `g` and `h`, respectively. That is, for our application,

$$\begin{aligned} \boldsymbol{D}_k^{-1} &\sim \mathcal{W}\bigl(6,\, \mathsf{diag}(\gamma_{b,1}, \dots, \gamma_{b,5})\bigr), \qquad k = 1, 2, \\ \gamma_{b,l}^{-1} &\sim \mathcal{G}(0.2,\, 0.2777778), \qquad l = 1, \dots, 5. \end{aligned}$$

The user gets a full control over the choice of the hyperparameters by replacing the `prior.b = list(Kmax = 2)` statement in the `GLMM_MCMC` function call by

```
prior.b = list(Kmax = 2, priormuQ = "independentC",
               delta = 1, xi = rep(0, 5), D = diag(rep(36, 5)),
               zeta = 6, g = rep(0.2, 5), h = rep(0.2777778, 5))
```

5.4. Prior distribution for fixed effects

Model (1) contains a single fixed effect $\boldsymbol{\alpha} = \alpha_3$ which is the slope from the logit model for presence of blood vessel malformations. A normal prior distribution $\mathcal{N}(\boldsymbol{\xi}_\alpha,\, \boldsymbol{C}_\alpha)$ with a diagonal covariance matrix $\boldsymbol{C}_\alpha$ is assumed for (generally a vector) $\boldsymbol{\alpha}$. We check particular values of $\boldsymbol{\xi}_\alpha$, $\boldsymbol{C}_\alpha$.

```
R> print(mod[[1]]$prior.alpha)
```

```
$mean                $var
alpha1.mean          alpha1.var
          0               10000
```

That is, apriori $\alpha_3 \sim \mathcal{N}(0,\, 10\,000)$. Note that `alpha1` in the output indicates that it corresponds to the first component of in general a vector of fixed effects $\boldsymbol{\alpha}$. Nevertheless, for notational clarity, subscript 3 was used for α in model (1) to indicate that it pertains to the third marker. The values of the prior mean $\boldsymbol{\xi}_\alpha$ and a diagonal of the prior covariance matrix $\boldsymbol{C}_\alpha$ might be set by inclusion of the `prior.alpha` argument in the `GLMM_MCMC` function call. For example, the $\mathcal{N}(0,\, 10\,000)$ prior for α_3 is also achieved by using

```
prior.alpha = list(mean = 0, var = 10000)
```

The same information can be seen if we print this for the second sampled chain.

```
R> print(mod[[2]]$prior.alpha)
```

5.5. Prior distribution for dispersion parameters

Finally, model (1) contains a single dispersion parameter which is the residual variance σ_1^2 from the Gaussian model for logarithmic bilirubin ($Y_{i,1,j}$). It is apriori assumed that $\sigma_1^{-2} \sim \mathcal{G}(\zeta_{\phi,1}/2,\, \gamma_{\phi,1}^{-1}/2)$ where $\gamma_{\phi,1}^{-1}$ is random with a $\mathcal{G}(g_{\phi,1},\, h_{\phi,1})$ hyperprior. We examine particular values of $\zeta_{\phi,1}$, $g_{\phi,1}$, $h_{\phi,1}$ selected by the `GLMM_MCMC` routine using the guidelines given in Komárek and Komárková (2011b, Sec. A).

```
R> print(mod[[1]]$prior.eps)
```

```
$zeta      $g        $h
zeta1       g1              h1
    2      0.2        2.755851
```

The same information can be seen if we print this for the second sampled chain.

```
R> print(mod[[2]]$prior.eps)
```

That is, $\zeta_{\phi,1} = 2$, $g_{\phi,1} = 0.2$, $h_{\phi,1} = 2.755851$. The same values can be explicitely chosen by the user by adding the argument `prior.eps` in the `GLMM_MCMC` function call as

```
prior.eps = list(zeta = 2, g = 0.2, h = 2.755851)
```

5.6. Initial values

To start the MCMC simulation, initial values for the model parameters $\boldsymbol{\theta}$, $\boldsymbol{\psi}$, for latent quantities which are a matrix of random effects $\boldsymbol{B}$ and a vector of component allocation $\boldsymbol{u}$ and also for random hyperparameters $\gamma_{b,1}, \ldots, \gamma_{b,5}$ and $\gamma_{\phi,1}$ must be given. Reasonable initial values were automatically selected by the function `GLMM_MCMC` and are stored as `init.b`, `init.alpha` and `init.eps` components of the resulting object `mod`. The component `init.b` contains the initial values for mixture related parameters $\boldsymbol{\theta}$ and also for random effects $\boldsymbol{B}$ and component allocations $\boldsymbol{u}$ (the output was shortened).

```
R> print(mod[[1]]$init.b)
```

```
$b
             b1           b2       b3            b4         b5
1  -0.045833040  1.739426e-02 5.378129 -2.262848e-02  1.8360865
2   0.241232664  1.119751e-02 5.050482 -1.639513e-02 -0.4899194
```

```
3    0.489765757  1.828415e-02 5.365383  4.223745e-03  1.8387236
4    0.881059232  2.078636e-02 4.885222 -1.673188e-02 -0.5046516
5   -0.210395893  4.070156e-03 5.695454 -3.190762e-04 -3.7510091
...

$K         $w           $mu                    $Sigma
[1] 2       w1  w2         m1 m2 m3 m4 m5           m1 m2 m3 m4 m5
           0.5 0.5      j1 -1 -1 -1 -1 -1      j1.1  1  0  0  0  0
                        j2  1  1  1  1  1      j1.2  0  1  0  0  0
                                               j1.3  0  0  1  0  0
                                               j1.4  0  0  0  1  0
                                               j1.5  0  0  0  0  1
                                               j2.1  1  0  0  0  0
                                               j2.2  0  1  0  0  0
                                               j2.3  0  0  1  0  0
                                               j2.4  0  0  0  1  0
                                               j2.5  0  0  0  0  1

$Li
Li1.1.1 Li1.2.1 Li1.3.1 Li1.4.1 Li1.5.1 Li1.2.2 Li1.3.2 Li1.4.2 Li1.5.2
      1       0       0       0       0       1       0       0       0
Li1.3.3 Li1.4.3 Li1.5.3 Li1.4.4 Li1.5.4 Li1.5.5 Li2.1.1 Li2.2.1 Li2.3.1
      1       0       0       1       0       1       1       0       0
Li2.4.1 Li2.5.1 Li2.2.2 Li2.3.2 Li2.4.2 Li2.5.2 Li2.3.3 Li2.4.3 Li2.5.3
      0       0       1       0       0       0       1       0       0
Li2.4.4 Li2.5.4 Li2.5.5
      1       0       1

$gammaInv
gammaInv1 gammaInv2 gammaInv3 gammaInv4 gammaInv5
        6         6         6         6         6

$r
  r1   r2   r3   r4   r5   r6   r7   r8   r9  r10  r11  r12  r13  r14  r15
   2    1    1    1    1    1    1    1    1    1    1    2    1    1    1
  ...
```

Different initial values for the second sampled chain can be seen by invoking

```
R> print(mod[[2]]$init.b)
```

First, `mod[[*]]$init.b$b` is a matrix $\boldsymbol{B}^{\top}$ with initial values of individual random effects in rows, by default, equal to empirical Bayes estimates $\boldsymbol{B}_{ML}^{0\top}$ obtained from initial `lmer`/`glmer` fits of GLMM's (1) for chain 1 and [ADD EXPLANATION FOR CHAIN 2]. Second, `mod[[*]]$init.b$K` stores the information on the number of mixture components. Third, `mod[[*]]$init.b$w` gives the initial values of mixture weight which are, by default, equal all to $1/K$ for chain 1 and [ADD EXPLANATION FOR CHAIN 2]. The initial value for mixture means are shown in rows of a matrix `mod[[*]]$init.b$mu`. By default, for chain 1, initials for components of $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$ are chosen equidistantly on intervals starting at $\frac{\beta^0_{ML,l}-s_l}{S_l} - 3\frac{d^0_{ML,l}}{S_l} + \delta_l$ and ending at $\frac{\beta^0_{ML,l}-s_l}{S_l} + 3\frac{d^0_{ML,l}}{S_l} - \delta_l$, where $\delta_l = \frac{6d^0_{ML,l}}{(K+1)\,S_l}$, l=1,…,5. [ADD EXPLANATION FOR CHAIN 2]. The initial values for mixture covariance matrices $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ are shown as blocks of `mod[[*]]$init.b$Sigma`. By default, for chain 1, initial values are $\boldsymbol{D}_k = \mathsf{diag}\bigl((d^0_{ML,1}/S_1)^2, \ldots, (d^0_{ML,5}/S_5)^2\bigr)$, k=1,2 which in our case leads to $\boldsymbol{D}_1 = \boldsymbol{D}_2 = I_5$. Further, `mod$init.b$Li` shows lower triangles of Cholesky factors of inverted initial covariance matrices $\boldsymbol{D}_1^{-1}$, $\boldsymbol{D}_2^{-1}$ stacked in a vector. [ADD EXPLANATION FOR CHAIN 2]. Furthermore, `mod[[*]]$init.b$gammaInv` are initial values for hyperparameters $\gamma_{b,1}^{-1}, \ldots, \gamma_{b,5}^{-1}$, by default equal for chain 1 to $\zeta_b\,\bigl(d^0_{ML,l}/S_l\bigr)^2$, $l = 1, \ldots, 5$. [ADD EXPLANATION FOR CHAIN 2]. Finally, `mod[[*]]$init.b$r` is a vector of initial values of component allocations $\boldsymbol{u}$. For each subject i, $i = 1, \ldots, N$, the initial value of u_i for chain 1 is by default equal to $g(i)$ for which $p\bigl(\boldsymbol{b}_i \,\big|\, \boldsymbol{\theta},\, u_i = g(i)\bigr)$, see expression (3), is maximal at initial values of remaining parameters. [ADD EXPLANATION FOR CHAIN 2].

Further, `mod[[*]]$init.alpha` contains intial values for a vector of fixed effects $\boldsymbol{\alpha}$, by default equal to $\boldsymbol{\alpha}^0_{ML}$ from the initial `lmer`/`glmer` fit for chain 1. [ADD EXPLANATION FOR CHAIN 2].

```
R> print(mod[[1]]$init.alpha)
```

```
    alpha1
0.02560626
```

That is, in our application, the inital value for chain 1 was $\boldsymbol{\alpha} = \alpha_3 = 0.02560626$.

Finally, `mod[[*]]$init.eps` keeps the initial values for parameters related to the GLMM dispersion parameters. The initial value of the residual standard deviation σ_1 from the model for logarithmic bilirubin is for chain 1 by default equal to $\sigma^0_{ML,1}$, the initial value of random hyperparameter $\gamma_{\phi,1}^{-1}$ is for chain 1 by default equal to $\zeta_{\phi,1}\,(\sigma^0_{ML,1})^2$. [ADD EXPLANATION FOR CHAIN 2].

```
R> print(mod[[1]]$init.eps)
```

```
$sigma                     $gammaInv
   sigma1                  gammaInv1
0.3174833                  0.2015913
```

In this case, initially for chain 1, $\sigma_1 = 0.3174833$, $\gamma_{\phi,1}^{-1} = 0.2015913$.

The user is able to define his/her own initial values for all or a subset of model parameters by defining the `lists` with the same structure as `mod[[1]]$init.b`, `mod[[2]]$init.b`, `mod[[1]]$init.alpha`, `mod[[2]]$init.alpha`, `mod[[1]]$init.eps`, `mod[[2]]$init.eps`, and using these `lists` as additional arguments `init.b`, `init2.b`[1], `init.alpha`, `init2.alpha`, `init.eps`, `init2.eps` in the call to `GLMM_MCMC` function. At the same time, the user does

---

[1] Only either `Sigma` or `Li` component is sufficient to specify the initial values for mixture covariance matrices.

not have to specify the initial values for all sets of parameters as missing components are initialized using the default procedures described above.

Additionally, the objects `mod[[1]]` and `mod[[2]]` contains components `state.first.b`, `state.last.b`, `state.first.alpha`, `state.last.alpha`, `state.first.eps`, `state.last.eps` which have the same structure as corresponding `init.*` components.[2] The `state.first.*` components contain the values of the chain at the first saved (after burn-in) MCMC iteration, the `state.last.*` components contain the last saved values of the chain. All of them can be directly supplied as corresponding `init.*` arguments to the `GLMM_MCMC` function when one wishes to re-start the MCMC simulation either from the end of the original burn-in period or from the end of currently finished MCMC simulation. To illustrate this and also explicit specification of hyperparameters of the prior distribution, shift vector $\boldsymbol{s}$ and scale matrix $\boldsymbol{S}$, we generate a sample of 1 000 values, again with 1:100 thinning, starting from the last MCMC iteration saved in the `mod`.

```
R> set.seed(20042007)
R> mod02 <- GLMM_MCMC(y = pbc01[, c("lbili", "platelet", "spiders")],
+      dist = c("gaussian", "poisson(log)", "binomial(logit)"),
+      id = pbc01[, "id"],
+      x = list(lbili    = "empty",
+               platelet = "empty",
+               spiders  = pbc01[, "month"]),
+      z = list(lbili    = pbc01[, "month"],
+               platelet = pbc01[, "month"],
+               spiders  = "empty"),
+      random.intercept = rep(TRUE, 3),
+      scale.b = list(shift = c(0.315158108, 0.007654708, 5.526209768,
+                               -0.006634000, -2.749539170),
+                     scale = c(0.86449212, 0.02007624, 0.34860152,
+                               0.01565365, 3.22849129)),
+      prior.b = list(Kmax = 2, priormuQ = "independentC",
+                     delta = 1, xi = rep(0, 5), D = diag(rep(36, 5)),
+                     zeta = 6, g = rep(0.2, 5), h = rep(0.2777778, 5)),
+      prior.alpha = list(mean = 0, var = 10000),
+      prior.eps = list(zeta = 2, g = 0.2, h = 2.755851),
+      init.b   = mod[[1]]$state.last.b,
+      init2.b = mod[[2]]$state.last.b,
+      init.alpha  = mod[[1]]$state.last.alpha,
+      init2.alpha = mod[[2]]$state.last.alpha,
+      init.eps = mod[[1]]$state.last.eps,
+      init2.eps = mod[[2]]$state.last.eps,
+      nMCMC = c(burn = 0, keep = 1000, thin = 100, info = 1000),
+      PED = TRUE, parallel = FALSE)
```

5.7. Posterior samples

---

[2] The `state.first.b` and `state.last.b` components contain additionally element `Q` which are lower triangles of inverted mixture covariance matrices stacked in a vector.

Posterior samples of model parameters $\boldsymbol{\theta}$, $\boldsymbol{\psi}$, random hyperparameters and some additional derived quantities are kept in the resulting object `mod` as a series of components. All of them are matrices where each row corresponds to one MCMC iteration and each column to one parameter or element of a vector parameter. With respect to mixture related parameters, we find the component `w_b` (mixture weights $\boldsymbol{w}$), for which we print the first three sampled values of chain 1.

```
R> print(mod[[1]]$w_b[1:3,])
```

```
            w1        w2
[1,] 0.5861780 0.4138220
[2,] 0.5760271 0.4239729
[3,] 0.6140991 0.3859009
```

Similarly, we find the component `mu_b` with sampled mixture means $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ (the first three sampled values of chain 1 are printed).

```
R> print(mod[[1]]$mu_b[1:3,])
```

```
         mu.1.1      mu.1.2     mu.1.3      mu.1.4     mu.1.5    mu.2.1
[1,] -0.5757105 -0.14819857 0.10030886  0.01287247 -0.6078800 0.9933894
[2,] -0.6963823 -0.14882040 0.10367778  0.12907884 -0.5953020 1.1181582
[3,] -0.5663048 -0.09290386 0.05253764 -0.06194500 -0.3588196 1.1579885
        mu.2.2     mu.2.3      mu.2.4    mu.2.5
[1,] 0.2363280 -0.1436500 -0.27251015 0.6956592
[2,] 0.1045462 -0.1662364 -0.05900691 0.5829094
[3,] 0.1956312 -0.1845504  0.08468051 0.4105315
```

The first five columns contain the values of $\boldsymbol{\mu}_1$, the remaining five columns contain the values of $\boldsymbol{\mu}_2$. Further, the component `Sigma_b` contains sampled values of lower triangles of mixture covariance matrices $\boldsymbol{D}_1, \boldsymbol{D}_2$, see the first three sampled values of chain 1:

```
R> print(mod[[1]]$Sigma_b[1:3,])
```

```
     Sigma1.1.1  Sigma1.2.1  Sigma1.3.1   Sigma1.4.1 Sigma1.5.1 Sigma1.2.2
[1,]  0.1483652  0.02725310 -0.12066884  0.006712001 0.09787152  0.1401401
[2,]  0.1897234 -0.02665165 -0.08574339 -0.085695122 0.01485711  0.1675582
[3,]  0.3113878  0.06903808 -0.34222189 -0.100087885 0.40650164  0.1548528
      Sigma1.3.2  Sigma1.4.2  Sigma1.5.2 Sigma1.3.3  Sigma1.4.3
[1,] -0.03533349 -0.10356217 0.009842313  0.8480862  0.02543108
[2,]  0.05805863 -0.03786570 0.178521511  0.7892866 -0.21511899
[3,] -0.07138611 -0.05850689 0.194395314  1.3587504  0.16404291
       Sigma1.5.3 Sigma1.4.4  Sigma1.5.4 Sigma1.5.5 Sigma2.1.1 Sigma2.2.1
[1,]  0.009172017  0.4433576  0.25815239   1.493001  0.6246352 -0.3877369
[2,]  0.284246759  0.7382875 -0.10912226   1.169762  0.7320425 -0.3345317
```

```
[3,] -0.618696188  0.6534640 -0.06728697   1.774370  0.8968992 -0.2011429
     Sigma2.3.1  Sigma2.4.1 Sigma2.5.1 Sigma2.2.2  Sigma2.3.2 Sigma2.4.2
[1,]  0.2444709  0.00820094 0.03761508   3.103192  0.16108994  0.7715097
[2,]  0.2603831 -0.38925799 0.08142149   2.137363  0.02972364  0.5766868
[3,] -0.1523893 -0.51109311 0.10041998   2.328876 -0.14525281  0.1847445
      Sigma2.5.2 Sigma2.3.3  Sigma2.4.3  Sigma2.5.3 Sigma2.4.4  Sigma2.5.4
[1,]  0.15420589   1.631642  0.02649452  0.07681108   2.458052 -0.09860401
[2,] -0.05666061   1.339500 -0.28396256 -0.10200859   2.477405 -0.12539592
[3,]  0.29375832   1.101824 -0.31911426 -0.19380199   2.358011 -0.11308685
     Sigma2.5.5
[1,]  0.7350301
[2,]  0.4446280
[3,]  0.3946266
```

The first 15 columns comprise the lower triangles of sampled matrices $\boldsymbol{D}_1$, the remaining 15 columns comprise the lower triangles of sampled matrices $\boldsymbol{D}_2$. Similarly, components `Q_b` and `Li_b` contain sampled values of lower triangles of inverted mixture covariance matrices $\boldsymbol{D}_1^{-1}, \boldsymbol{D}_2^{-1}$ and their Cholesky decompositions, respectively.

All above mentioned mixture related parameters are saved as sampled, i.e., without applying any relabelling algorithm. Nevertheless, during the MCMC simulation provided by the `GLMM_MCMC` function, initial relabelling based on ordering of the first margins of the mixture means was applied and the resulting relabelling is reflected in components `order_b` and `rank_b` which are also present in the objects `mod[[1]]` and `mod[[2]]`. Both of them are $M \times K$ matrices. Let $o_{m,k}$, $m = 1, \ldots, M$, $k = 1, \ldots, K$ be the elements of matrix `order_b` and $r_{m,k}$ $m = 1, \ldots, M$, $k = 1, \ldots, K$ the elements of matrix `rank_b`. After relabelling, component number κ, $\kappa \in \{1, \ldots, K\}$ at iteration m is given by the $o_{m,\kappa}$th component in the original sample, or vice versa, component number ι in the original sample equals to the $r_{m,\iota}$th component in the relabeled sample.

Further, the objecs `mod[[1]]` and `mod[[2]]` contain the following components holding the sampled values of model parameters: `alpha` (fixed effects $\boldsymbol{\alpha} = \alpha_3$), `sigma_eps` (residual standard deviation σ_1). Further, samples for random hyperparameters are saved in components `gammaInv_b` ($\gamma_{b,1}, \ldots, \gamma_{b,5}$) and `gammaInv_eps` ($\gamma_{\phi,1}$).

Additionally, samples for some derived model quantities are stored in the objects `mod[[1]]` and `mod[[2]]`. First, we check the component `mixture_b` (only first three rows from chain 1 are printed).

```
R> print(mod[[1]]$mixture_b[1:3,])
```

```
   b.Mean.1    b.Mean.2 b.Mean.3     b.Mean.4  b.Mean.5    b.SD.1
1 0.3787999 0.007874080 5.525984 -0.008281157 -2.970519 0.8393546
2 0.3782087 0.006823555 5.522459 -0.005861718 -3.058739 0.9563330
3 0.4008306 0.008024956 5.512630 -0.006717936 -2.949468 0.9634055
    b.Corr.2.1  b.Corr.3.1  b.Corr.4.1 b.Corr.5.1     b.SD.2  b.Corr.3.2
1  0.001635467 -0.05901592 -0.09161347  0.4646133 0.02377304  0.01798102
2 -0.040225348 -0.05194083 -0.22092440  0.4661432 0.02025990  0.02850830
3  0.073691987 -0.29097023 -0.15546775  0.4600577 0.02021169 -0.10221657
```

```
  b.Corr.4.2 b.Corr.5.2    b.SD.3  b.Corr.4.3  b.Corr.5.3      b.SD.4
1 0.17200261  0.1279722 0.3797639  0.03446715 -0.02908926 0.01782619
2 0.17166193  0.1371638 0.3555658 -0.18662511  0.03830124 0.01907091
3 0.03929654  0.2410830 0.3933058 -0.02367496 -0.37540904 0.01795976
   b.Corr.5.4   b.SD.5
1  0.01411600 4.072918
2 -0.12741539 3.538619
3 -0.04316814 3.795634
```

It is a sample for overall means (columns `b.Mean.*`)

$$\boldsymbol{\beta} = (\beta_{1,1},\,\beta_{1,2},\,\beta_{2,1},\,\beta_{2,2},\,\beta_3)^\top = \mathsf{E}\bigl(\boldsymbol{b}\,\big|\,\boldsymbol{\theta}\bigr) = \boldsymbol{s} + \boldsymbol{S}\sum_{k=1}^{K} w_k\boldsymbol{\mu}_k \tag{6}$$

of random effects, see also Komárek and Komárková (2011a, Eq. (3)) and standard deviations (columns `b.SD.*`) and correlations coefficients (columns `b.Corr.*.*`) derived from the overall covariance matrix

$$\boldsymbol{D} = \mathsf{VAR}\bigl(\boldsymbol{b}\,\big|\,\boldsymbol{\theta}\bigr) = \boldsymbol{S}\Biggl[\sum_{k=1}^{K} w_k\Bigl\{\boldsymbol{D}_k \,+\, \bigl(\boldsymbol{\mu}_k - \sum_{j=1}^{K} w_j\boldsymbol{\mu}_j\bigr)\bigl(\boldsymbol{\mu}_k - \sum_{j=1}^{K} w_j\boldsymbol{\mu}_j\bigr)'\Bigr\}\Biggr]\boldsymbol{S}' \tag{7}$$

of random effects, see also Komárek and Komárková (2011a, Eq. (4)). Further, we print the first ten values of the component `Deviance`.

```
R> print(mod[[1]]$Deviance[1:10])

 [1] 14081.78 14092.04 14096.07 14080.28 14092.15 14080.67 14096.09
 [8] 14088.80 14103.54 14099.09
```

This is a sample of observed data deviances, that is sample for $D(\boldsymbol{\psi},\,\boldsymbol{\theta}) = -2\log\bigl\{L(\boldsymbol{\psi},\,\boldsymbol{\theta})\bigr\}$ where $L(\boldsymbol{\psi},\,\boldsymbol{\theta}) = p(\boldsymbol{y}\,|\,\boldsymbol{\psi},\,\boldsymbol{\theta})$ is the observed data (frequentist) likelihood of the GLMM (1) with the normal mixture (2) in the random effects distribution. As it is explained in full generality in Komárek and Komárková (2011a, Sec. 2.5),

$$L(\boldsymbol{\psi},\,\boldsymbol{\theta}) = \prod_{i=1}^{N}\Bigl\{\sum_{k=1}^{K} w_k\,L_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta})\Bigr\}, \tag{8}$$

with

$$L_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta}) = \int\Bigl\{\prod_{r=1}^{3}\prod_{j=1}^{n_{i,r}} p(y_{i,r,j}\,|\,\boldsymbol{\psi},\,\boldsymbol{b}_{i,r})\Bigr\}p(\boldsymbol{b}_i\,|\,\boldsymbol{\theta},\,u_i = k)d\boldsymbol{b}_i, \qquad i = 1,\ldots,N,\ k = 1,\ldots,K, \tag{9}$$

where

$$p(y_{i,1,j}\,|\,\boldsymbol{\psi},\,\boldsymbol{b}_{i,1}) = \frac{1}{\sigma_1\sqrt{2\pi}}\exp\Bigl\{-\frac{(y_{i,1,j}-\lambda_{i,1,j})^2}{2\sigma_1^2}\Bigr\}, \qquad \lambda_{i,1,j} = b_{i,1,1} + b_{i,1,2}t_{i,1,j}, \tag{10}$$

$$p(y_{i,2,j}\,|\,\boldsymbol{\psi},\,\boldsymbol{b}_{i,2}) = \frac{\lambda_{i,2,j}^{y_{i,2,j}}\exp(-\lambda_{i,2,j})}{y_{i,2,j}!}, \qquad \lambda_{i,2,j} = \exp(b_{i,2,1} + b_{i,2,2}t_{i,2,j}), \tag{11}$$

$$p(y_{i,3,j}\,|\,\boldsymbol{\psi},\,\boldsymbol{b}_{i,3}) = \lambda_{i,3,j}\,(1-\lambda_{i,3,j}), \qquad \lambda_{i,3,j} = \frac{\exp(\alpha_3 + b_{i,3}t_{i,3,j})}{1+\exp(\alpha_3 + b_{i,3}t_{i,2,j})} \tag{12}$$

are likelihood contributions given by the Gaussian, Poisson with a log-link and Bernoulli with a logit-link generalized linear models for the three markers, respectively. Note that the integral in (9) cannot be evaluated analytically and function `GLMM_MCMC` uses Laplace approximation to calculate numerically its value. Similarly, component `Cond.Deviance` is a vector of sampled values of the deviances based on the conditional (given random effects and component allocations) likelihood

$$L^{Bayes}(\boldsymbol{\psi},\,\boldsymbol{\theta},\,\boldsymbol{B},\,\boldsymbol{u}) = p(\boldsymbol{y}\,|\,\boldsymbol{\psi},\,\boldsymbol{\theta},\,\boldsymbol{B},\,\boldsymbol{u}) = p(\boldsymbol{y}\,|\,\boldsymbol{\psi},\,\boldsymbol{B}) = \prod_{i=1}^{N}\prod_{r=1}^{3}\prod_{j=1}^{n_{i,r}} p(y_{i,r,j}\,|\,\boldsymbol{\psi},\,\boldsymbol{b}_{i,r}). \qquad (13)$$

Note that $\boldsymbol{\beta}$, $\boldsymbol{D}$ as well as the observed data and conditional deviances are invariant towards label switching.

5.8. Re-labelling of the posterior sample

As was pointed out at the beginning of this Section, application of a suitable relabelling algorithm is necessary for successful use of the results for clustering purposes. Simple relabelling based on ordering of components of mixture means (as currently reflected in the `order_b` and `rank_b` components of the object `mod`) is often unsatisfactory (Stephens 2000) and utilization of a more sophisticated procedure is recommended. Hence we apply the Stephens' relabelling algorithm on the chains stored in `mod`. At the same time, by setting `keep.comp.prob = TRUE` we generate posterior samples for individual probabilities

$$p_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta}) = \mathsf{P}(u_i = k\,|\,\boldsymbol{\psi},\,\boldsymbol{\theta},\,\boldsymbol{y}), \qquad i = 1,\ldots,N,\; k = 1,\ldots,K, \qquad (14)$$

that subject i belongs to the kth group (where k refers to a new labeling of the components), see Komárek and Komárková (2011a, Sec. 2.5) for theoretical details.

```
R> mod[[1]]  <- NMixRelabel(mod[[1]], type = "stephens", keep.comp.prob = TRUE)
R> mod[[2]]  <- NMixRelabel(mod[[2]], type = "stephens", keep.comp.prob = TRUE)
```

Objects `mod[[1]]` and `mod[[2]]` have the same structure as before with exception that all results which are not invariant towards label switching (components `order_b`, `rank_b` and also not yet mentioned components `poster.mean.w_b`, `poster.mean.mu_b`, `poster.mean.Sigma_b`, `poster.mean.Q_b`, `poster.mean.Li_b`, `poster.comp.prob1`, `poster.comp.prob2`, `poster.comp.prob3`, `comp.prob2`, `comp.prob3`, `quant.comp.prob2`),`quant.comp.prob3`) were (re-)calculated to reflect the new labelling of the mixture components.

6. Basic convergence diagnostics

Classical tools for convergence diagnostics, e.g., **R** package **coda** can be used to evaluate the convergence of the performed MCMC simulation by applying the appropriate methods to posterior samples described in Section 5.7 or their derivatives. On top of that, package **mixAK** offers the following routine which simplifies production of some of the output useful for convergence diagnostics.

- `tracePlots()` is a generic function with a method for objects of class `GLMM_MCMC` which produces traceplots of selected model parameters.

In this Section, we illustrate the use of this and also some of **coda** routines on exploration of the properties of generated Markov chain stored in the object `mod`.

6.1. Traceplots

A basic tool to reveal convergence problems are traceplots of important model parameters or other quantities derived from these parameters. We start by drawing a traceplot of the observed data deviances $D(\boldsymbol{\psi}, \boldsymbol{\theta})$, see Figure 2.

```
R> tracePlots(mod[[1]], param = "Deviance")
```

Similarly, the traceplots of the conditional deviances, fixed effect α_3 and the residual standard deviations σ_1 from the Gaussian model for logarithmic bilirubin, respectively, are obtained using the following commands (output not shown).

```
R> tracePlots(mod[[1]], param = "Cond.Deviance")
R> tracePlots(mod[[1]], param = "alpha")
R> tracePlots(mod[[1]], param = "sigma_eps")
```

Further, we can draw traceplots for the overall means $\boldsymbol{\beta}$ (Eq. (6)) of random effects (output not shown).

```
R> COL <- rep(rainbow_hcl(3, start = 30, end = 210), c(2, 2, 1))
R> tracePlots(mod[[1]], param = "Eb", col = COL)
```

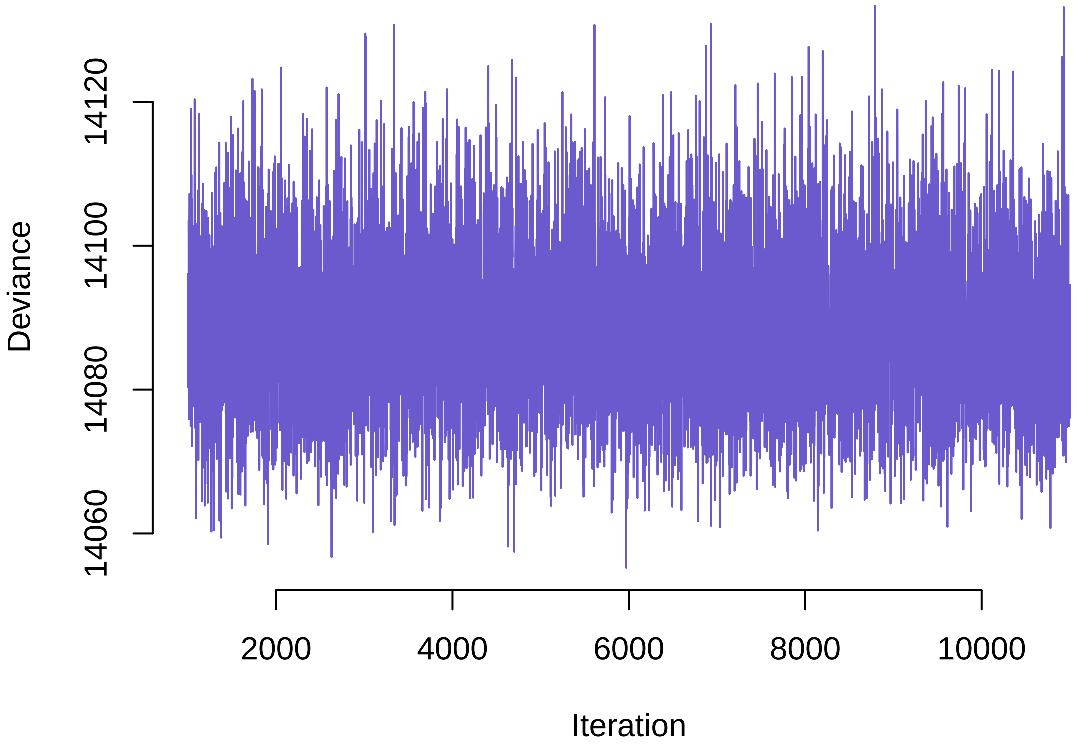

Figure 2: Traceplot of the observed data deviance $D(\boldsymbol{\psi}, \boldsymbol{\theta})$.

Traceplots of standard deviations and correlation coefficients derived from the overall covariance matrix $\boldsymbol{D}$ (Eq. (7)) are drawn using the commands (output not shown).

```
R> tracePlots(mod[[1]], param = "SDb")
R> tracePlots(mod[[1]], param = "Corb")
```

Additionally, traceplots of sampled (as such before applying any re-labelling algorithm) mixture weights w_1, w_2, components of mixture means $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ and standard deviations from mixture covariance matrices $\boldsymbol{D}_1, \boldsymbol{D}_2$ can be drawn using the following sequence of commands (output not shown).

```
R> tracePlots(mod[[1]], param = "w_b")
R> tracePlots(mod[[1]], param = "mu_b")
R> tracePlots(mod[[1]], param = "sd_b")
```

Traceplots of these quantities after current re-labelling (reflected by `order_b` and `rank_b` components of the object `mod`) are drawn by setting the `relabel` argument to `TRUE` (output not shown).

```
R> tracePlots(mod[[1]], param = "w_b", relabel = TRUE)
R> tracePlots(mod[[1]], param = "mu_b", relabel = TRUE)
R> tracePlots(mod[[1]], param = "sd_b", relabel = TRUE)
```

Finally, we can draw the traceplots of the hyperparameters $\gamma_{b,1}, \ldots, \gamma_{b,5}, \gamma_{\phi,1}$.

```
R> tracePlots(mod[[1]], param = "gammaInv_b")
R> tracePlots(mod[[1]], param = "gammaInv_eps")
```

6.2. Autocorrelations

Another tool which is often used to explore the properties of the generated Markov chain are plots of estimated autocorrelation functions based on sampled values. We draw it for the observed data deviance using the `autocorr.plot` function from the **coda** package, see Figure 3.

```
R> autocorr.plot(mod[[1]]$Deviance, lag.max = 20, col = "blue4",
+                auto.layout = FALSE, lwd = 2)
```

Similarly estimated autocorrelation functions for the conditional deviance, fixed effect α_3 and the residual standard deviations σ_1 from the Gaussian model for logarithmic bilirubin, respectively, are drawn using the following command (output not shown).

```
R> autocorr.plot(mod[[1]]$Cond.Deviance, col = "blue4", auto.layout = FALSE, lwd = 2)
R> autocorr.plot(mod[[1]]$alpha, col = "blue4", auto.layout = FALSE, lwd = 2)
R> autocorr.plot(mod[[1]]$sigma_eps, col = "blue4", auto.layout = FALSE, lwd = 2)
```

Further, we draw autocorrelation plots for the overall means $\boldsymbol{\beta}$ (Eq. (6)) of random effects, see Figure 4,

```
R> layout(autolayout(5))
R> name.Eb <- paste("b.Mean.", 1:5, sep = "")
R> autocorr.plot(mod[[1]]$mixture_b[, name.Eb], lag.max = 20, col = "blue4",
+                 auto.layout = FALSE, lwd = 2)
```

Autocorrelation plots for standard deviations and correlation coefficients derived from the overall covariance matrix $\boldsymbol{D}$ (Eq. (7)) are drawn using the commands (output not shown).

```
R> layout(autolayout(5))
R> name.SDb <- paste("b.SD.", 1:5, sep = "")
R> autocorr.plot(mod[[1]]$mixture_b[, name.SDb], lag.max = 20, col = "blue4",
+                 auto.layout = FALSE, lwd = 2)
R> #
R> layout(autolayout(10))
R> name.Corb <- paste("b.Corr.", c(2:5, 3:5, 4:5, 5), ".", rep(1:4, 4:1), sep = "")
R> autocorr.plot(mod[[1]]$mixture_b[, name.Corb], lag.max = 20, col = "blue4",
+                 auto.layout = FALSE, lwd = 2)
```

Similarly, autocorrelation plots for other model parameters or their derivatives might be created and examined.

7. Posterior summary for classical GLMM parameters

To summarize the estimated models, we calculate posterior summary statistics and credible intervals for model parameters which are classically estimated in the context of generalized mixed models. These are: fixed effects $\boldsymbol{\alpha} = \alpha_3$ and means $\boldsymbol{\beta}$ of random effects which together

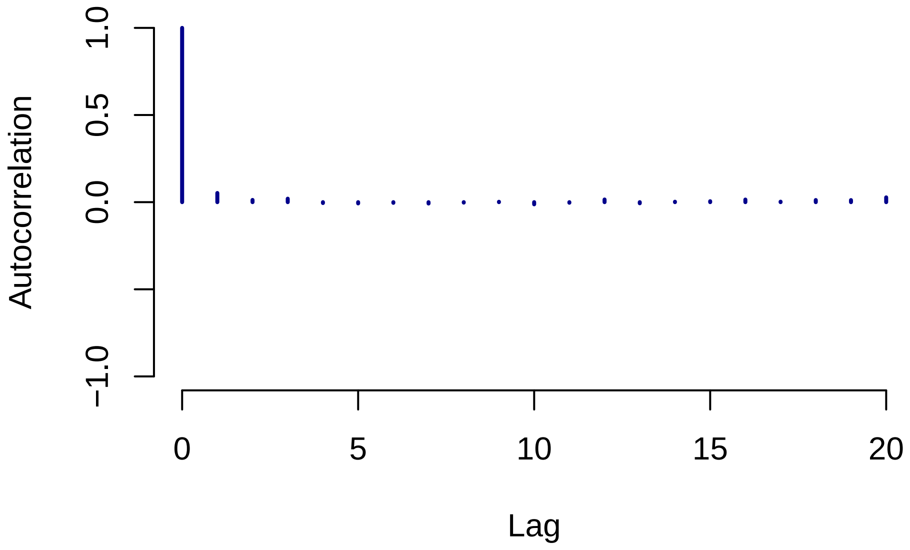

Figure 3: Estimated autocorrelation based on sampled values of the observed data deviance $D(\boldsymbol{\psi}, \boldsymbol{\theta})$.

quantify the population effect of included covariates on the response variables. Additionally, following parameters are usually of interest: residual standard deviation σ_1 from a Gaussian model for logarithmic bilirubin which quantifies the within-patient variability of log-bilirubin measurement, overall standard deviations and covariances or correlations for random effects (derived from the overall covariance matrix $\boldsymbol{D}$). Finally, as basis for possible model comparison, we calculate posterior summary statistics and credible intervals for observed data deviance $D(\boldsymbol{\psi}, \boldsymbol{\theta})$. Note that all mentioned quantities are invariant towards label switching. Apart from standard capabilities for calculation of posterior summary statistics provided by the **R** package **coda**, we introduce the following routines from the package **mixAK**.

- `print()` method for objects of class `GLMM_MCMC` and `GLMM_MCMClist` which outputs posterior summary statistics in a synoptic form for important model parameters;
- `fitted()` method for objects of class `GLMM_MCMC` which takes suitable posterior summary statistics (mean, median, ...) of fixed effects $\boldsymbol{\alpha}$ and means of random effects $\boldsymbol{\beta}$ and calculates estimated longitudinal profiles of the response variables for required values of covariates.

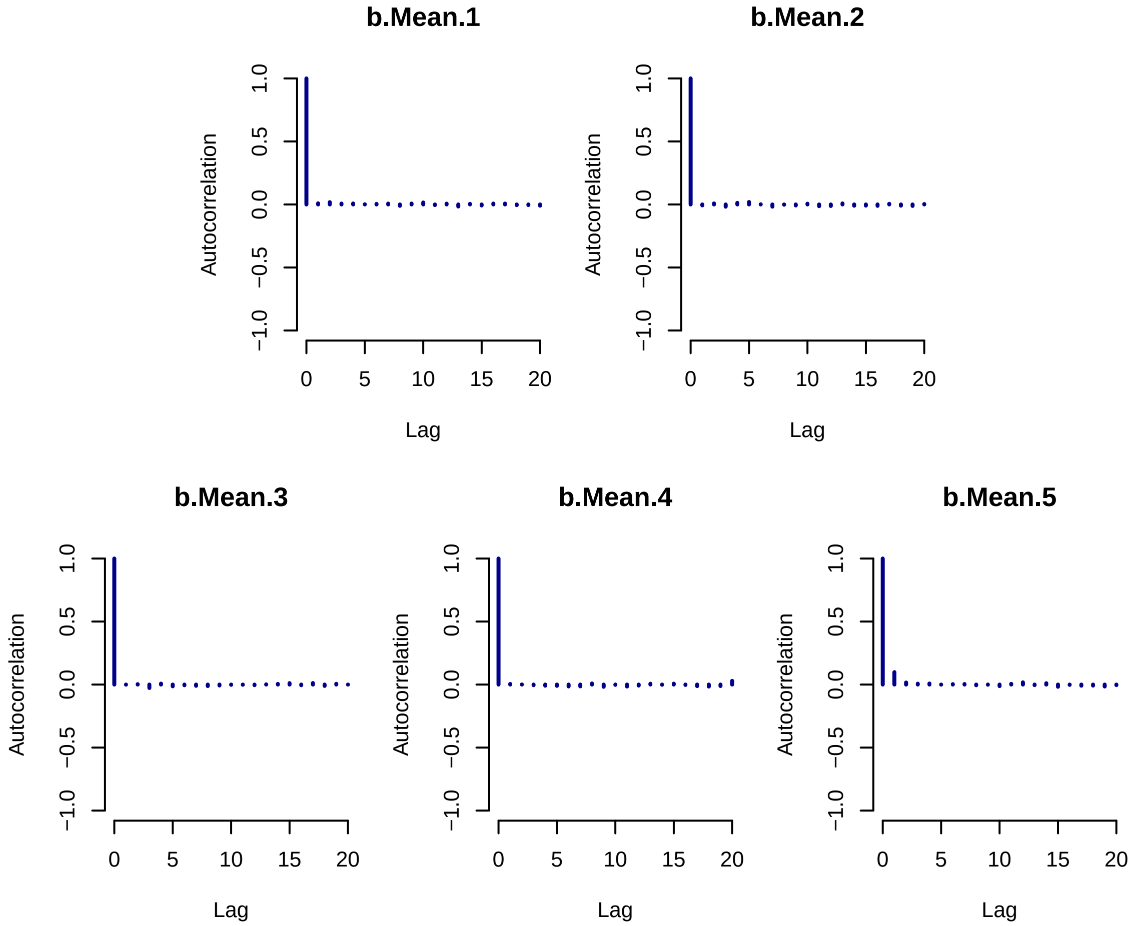

Figure 4: Estimated autocorrelation based on sampled values of the overall means $\boldsymbol{\beta}$ of random effects.

7.1. Basic posterior summary statistics

Posterior summary statistics for above mentioned quantities has in fact already been calculated by the function `GLMM_MCMC` and are stored as components `summ.Deviance` (posterior summary statistics for observed data deviance $D(\boldsymbol{\psi}, \boldsymbol{\theta})$), `summ.alpha` (posterior summary statistics for the fixed effect α_3), `summ.sigma_eps` (posterior summary statistics for the reisual standard deviation σ_1), `summ.b.Mean` (posterior summary statistics for the overall means $\boldsymbol{\beta}$ of random effects), `summ.b.SDCorr` (posterior summary statistics for standard deviations and correlation coefficients derived from the overall covariance matrix $\boldsymbol{D}$ of random effects) of the objects `mod[[1]]` and `mod[[2]]`. To inspect their values in a synoptic form, we simply print this object.

```
R> print(mod)


     Generalized linear mixed model for 3 responses estimated using MCMC
     ====================================================================

Penalized expected deviance:
----------------------------
   D.expect      p(opt)         PED     wp(opt)        wPED
14088.17659    74.86202 14163.03861    74.82185 14162.99844

Deviance posterior summary statistics:
-----------------------------------------------
            Mean Std.Dev.     Min.     2.5%  1st Qu.   Median  3rd Qu.
Chain 1 14088.24 10.38671 14055.23 14069.74 14080.93 14087.57 14094.68
Chain 2 14088.13 10.43965 14053.97 14069.43 14080.64 14087.56 14094.84
           97.5%     Max.
Chain 1 14110.43 14133.33
Chain 2 14110.11 14133.84

Posterior summary statistics for fixed effects:
-----------------------------------------------
              Mean   Std.Dev.        Min.        2.5%    1st Qu.
Chain 1 0.02802721 0.01297156 -0.01948352 0.002757211 0.01913345
Chain 2 0.02804690 0.01272354 -0.02387802 0.003474781 0.01939963
            Median    3rd Qu.      97.5%       Max.
Chain 1 0.02793498 0.03676435 0.05353450 0.07345849
Chain 2 0.02785970 0.03645750 0.05328103 0.08187126

Distribution of random effects is a normal mixture with 2 components
---------------------------------------------------------------------
Posterior summary statistics for moments of mixture for random effects:
-----------------------------------------------------------------------
Means:
         b.Mean.1(Chain 1) b.Mean.1(Chain 2) b.Mean.2(Chain 1)
Mean            0.31489792        0.31483865      0.0078279034
```

```
Std.Dev.        0.05687024        0.05597357      0.0018522622
Min.            0.13094232        0.13605722      0.0002825657
2.5%            0.20447151        0.20836454      0.0041760466
1st Qu.         0.27678609        0.27615643      0.0065920026
Median          0.31362019        0.31412212      0.0078315437
3rd Qu.         0.35270103        0.35253912      0.0090696223
97.5%           0.42976080        0.42490433      0.0114618416
Max.            0.53598711        0.55464820      0.0146877128
         b.Mean.2(Chain 2) b.Mean.3(Chain 1) b.Mean.3(Chain 2)
Mean           0.007857248        5.52671528        5.52641864
Std.Dev.       0.001857585        0.02248193        0.02228574
Min.           0.001119513        5.43271613        5.44289074
2.5%           0.004196234        5.48262040        5.48276209
1st Qu.        0.006596830        5.51157562        5.51142614
Median         0.007858602        5.52691500        5.52686759
3rd Qu.        0.009117601        5.54215278        5.54134516
97.5%          0.011439556        5.56970657        5.56945832
Max.           0.015453538        5.60588162        5.61606899
         b.Mean.4(Chain 1) b.Mean.4(Chain 2) b.Mean.5(Chain 1)
Mean          -0.006743481      -0.006725579        -2.9099164
Std.Dev.       0.001142145       0.001148100         0.4837242
Min.          -0.011276667      -0.011261919        -6.2960729
2.5%          -0.009007707      -0.009001230        -3.9862596
1st Qu.       -0.007501477      -0.007493934        -3.1936515
Median        -0.006741291      -0.006691344        -2.8669268
3rd Qu.       -0.005978643      -0.005964303        -2.5827107
97.5%         -0.004530282      -0.004545705        -2.0797337
Max.          -0.001866809      -0.002575189        -1.3219925
         b.Mean.5(Chain 2)
Mean            -2.9128661
Std.Dev.         0.4818729
Min.            -5.4510756
2.5%            -3.9726817
1st Qu.         -3.2082227
Median          -2.8699729
3rd Qu.         -2.5764604
97.5%           -2.0773479
Max.            -1.2870739

Standard deviations and correlations:
         b.SD.1(Chain 1) b.SD.1(Chain 2) b.Corr.2.1(Chain 1)
Mean          0.87285780      0.87328133          0.05320370
Std.Dev.      0.04207569      0.04141777          0.10426096
Min.          0.72686885      0.74465734         -0.30851066
2.5%          0.79516600      0.79464956         -0.14807928
1st Qu.       0.84411543      0.84525032         -0.01795145
Median        0.87179539      0.87249338          0.05239554
```

```
3rd Qu.        0.89979655      0.90106743          0.12417160
97.5%          0.96023795      0.95723099          0.25812419
Max.           1.07996690      1.05838571          0.41603786
         b.Corr.2.1(Chain 2) b.Corr.3.1(Chain 1) b.Corr.3.1(Chain 2)
Mean              0.05207174        -0.142838516        -0.143313561
Std.Dev.          0.10467047         0.067977177         0.067105878
Min.             -0.31695608        -0.373075788        -0.397533788
2.5%             -0.15125358        -0.272947476        -0.271622624
1st Qu.          -0.01949973        -0.189190541        -0.188843819
Median            0.05151273        -0.144149854        -0.144063051
3rd Qu.           0.12383354        -0.097379269        -0.099920253
97.5%             0.25427501        -0.006152941        -0.006905992
Max.              0.41597032         0.129552173         0.121025440
         b.Corr.4.1(Chain 1) b.Corr.4.1(Chain 2) b.Corr.5.1(Chain 1)
Mean             -0.13708932         -0.13550821          0.46312431
Std.Dev.          0.07782893          0.07942440          0.05798145
Min.             -0.41670243         -0.43247042          0.22292257
2.5%             -0.28651724         -0.28865178          0.34389954
1st Qu.          -0.19086906         -0.18967973          0.42553874
Median           -0.13693761         -0.13650056          0.46588228
3rd Qu.          -0.08438715         -0.08198412          0.50315361
97.5%             0.01888570          0.02257741          0.56872715
Max.              0.15615509          0.15883911          0.64496926
         b.Corr.5.1(Chain 2) b.SD.2(Chain 1) b.SD.2(Chain 2)
Mean              0.46478317     0.020918387     0.020956291
Std.Dev.          0.05744724     0.002127909     0.002112694
Min.              0.21339113     0.014037551     0.014387420
2.5%              0.34362569     0.017009248     0.017051074
1st Qu.           0.42784281     0.019456613     0.019474136
Median            0.46710445     0.020856295     0.020888393
3rd Qu.           0.50490995     0.022291240     0.022296540
97.5%             0.56803538     0.025329177     0.025347837
Max.              0.66261318     0.030568269     0.031046699
         b.Corr.3.2(Chain 1) b.Corr.3.2(Chain 2) b.Corr.4.2(Chain 1)
Mean             -0.04280737         -0.04383702          0.14932874
Std.Dev.          0.09500748          0.09572310          0.11382894
Min.             -0.41578682         -0.38643160         -0.28403120
2.5%             -0.22758520         -0.22794970         -0.07374712
1st Qu.          -0.10822543         -0.10901274          0.07318174
Median           -0.04253305         -0.04392804          0.15021131
3rd Qu.           0.02120721          0.02102476          0.22732288
97.5%             0.14187156          0.14431148          0.36519397
Max.              0.30962567          0.29076441          0.52143640
         b.Corr.4.2(Chain 2) b.Corr.5.2(Chain 1) b.Corr.5.2(Chain 2)
Mean              0.15207423          0.15149850          0.14930534
Std.Dev.          0.11343550          0.09068398          0.09083737
Min.             -0.28631731         -0.16628504         -0.20083219
```

```
2.5%             -0.07606826         -0.02781551         -0.03339873
1st Qu.           0.07617147          0.09038612          0.08838740
Median            0.15235753          0.15299775          0.15083670
3rd Qu.           0.23010557          0.21292217          0.21103394
97.5%             0.36649227          0.32897319          0.32306742
Max.              0.49808310          0.47947288          0.46728347
         b.SD.3(Chain 1) b.SD.3(Chain 2) b.Corr.4.3(Chain 1)
Mean          0.35583754      0.35598259         -0.02753763
Std.Dev.      0.01722565      0.01739245          0.07568116
Min.          0.30063618      0.30032625         -0.29115409
2.5%          0.32473220      0.32368052         -0.17488882
1st Qu.       0.34379368      0.34398055         -0.07874264
Median        0.35517847      0.35520398         -0.02791579
3rd Qu.       0.36716734      0.36727140          0.02416662
97.5%         0.39158665      0.39176453          0.11912364
Max.          0.43988038      0.42895442          0.29629638
         b.Corr.4.3(Chain 2) b.Corr.5.3(Chain 1) b.Corr.5.3(Chain 2)
Mean             -0.02696777        -0.154670092        -0.157034629
Std.Dev.          0.07616620         0.076003624         0.076484634
Min.             -0.30434988        -0.430529201        -0.437465243
2.5%             -0.17474226        -0.303829014        -0.303246351
1st Qu.          -0.07871611        -0.206177049        -0.208922411
Median           -0.02730403        -0.155820096        -0.158272914
3rd Qu.           0.02538218        -0.104245026        -0.106170909
97.5%             0.12131703        -0.003745955        -0.003836278
Max.              0.24275115         0.165235821         0.209343426
         b.SD.4(Chain 1) b.SD.4(Chain 2) b.Corr.5.4(Chain 1)
Mean         0.016856072     0.016843050         -0.03654043
Std.Dev.     0.001241295     0.001216541          0.07436306
Min.         0.013394196     0.012920786         -0.33892160
2.5%         0.014648156     0.014673664         -0.18178889
1st Qu.      0.015995317     0.015990534         -0.08670039
Median       0.016760537     0.016755136         -0.03758897
3rd Qu.      0.017623768     0.017612455          0.01402791
97.5%        0.019499373     0.019442777          0.10962380
Max.         0.022903996     0.022245987          0.22707359
         b.Corr.5.4(Chain 2) b.SD.5(Chain 1) b.SD.5(Chain 2)
Mean             -0.03533042       3.8518958       3.8565687
Std.Dev.          0.07469032       0.6244691       0.6218358
Min.             -0.30518029       1.9793088       2.1842750
2.5%             -0.18116576       2.8424281       2.8317176
1st Qu.          -0.08555959       3.4169164       3.4169778
Median           -0.03567030       3.7785541       3.7859030
3rd Qu.           0.01496213       4.2097690       4.2207021
97.5%             0.11170572       5.2937428       5.2740559
Max.              0.24529186       7.9353658       7.1513505
```

```
Posterior summary statistics for standard deviations
of residuals of continuous responses:
----------------------------------------------------
              Mean   Std.Dev.      Min.      2.5%   1st Qu.    Median
Chain 1 0.3143806 0.01010008 0.2787472 0.2951569 0.3074424 0.3142047
Chain 2 0.3141542 0.01013398 0.2792843 0.2948743 0.3071814 0.3139059
           3rd Qu.     97.5%      Max.
Chain 1 0.3211925 0.3347104 0.3570582
Chain 2 0.3207388 0.3349294 0.3540355
```

For each parameter, we have posterior mean, posterior standard deviation and posterior 0%, 2.5%, 25%, 50%, 75%, 97.5% and 100% quantiles (separately for chain 1 and 2). Among other things, we directly see 95% equal-tail credible intervals.

7.2. Extended posterior summary statistics

To get the posterior summary including estimates of Monte Carlo errors in the estimation of the posterior means, one may use the `summary` procedure from the **coda** package applied directly to matrices of sampled values. In the following code, we create a matrix containing sampled values of means $\boldsymbol{\beta}$ of random effects and a fixed effect α_3 and calculate the posterior summary using the **coda** package.

```
R> Regr <- cbind(mod[[1]]$mixture_b[, name.Eb],
+                mod[[1]]$alpha)
R> colnames(Regr) <- paste(rep(c("lbili", "platelet", "spiders"), each = 2),
+                          ":", rep(c("Intcpt", "Slope"), 3), sep="")
R> Regr <- mcmc(Regr)
R> summary(Regr)
```

```
Iterations = 1:10000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

                     Mean       SD  Naive SE Time-series SE
lbili:Intcpt     0.314898 0.056870 5.687e-04      5.174e-04
lbili:Slope      0.007828 0.001852 1.852e-05      1.820e-05
platelet:Intcpt  5.526715 0.022482 2.248e-04      1.983e-04
platelet:Slope  -0.006743 0.001142 1.142e-05      1.037e-05
spiders:Intcpt  -2.909916 0.483724 4.837e-03      4.694e-03
spiders:Slope    0.028027 0.012972 1.297e-04      1.168e-04

2. Quantiles for each variable:
```

```
                      2.5%       25%       50%       75%    97.5%
lbili:Intcpt      0.204472  0.276786  0.313620  0.352701  0.42976
lbili:Slope       0.004176  0.006592  0.007832  0.009070  0.01146
platelet:Intcpt   5.482620  5.511576  5.526915  5.542153  5.56971
platelet:Slope   -0.009008 -0.007501 -0.006741 -0.005979 -0.00453
spiders:Intcpt   -3.986260 -3.193651 -2.866927 -2.582711 -2.07973
spiders:Slope     0.002757  0.019133  0.027935  0.036764  0.05353
```

Above results were also reported in Komárek and Komárková (2011b, Table C.1). Similarly, we can calculate posterior summary statistics for the residual standard deviation σ_1, standard deviations and correlations derived from the overall covariance matrix $\boldsymbol{D}$ of random effects to get results reported in Komárek and Komárková (2011b, Tables C.1 and C.2), output not shown here.

```
R> sigma1 <- mcmc(mod[[1]]$sigma_eps)
R> summary(sigma1)
R> #
R> SDb <- mcmc(mod[[1]]$mixture_b[, name.SDb])
R> summary(SDb)
R> #
R> Corb <- mcmc(mod[[1]]$mixture_b[, name.Corb])
R> summary(Corb)
```

7.3. Highest posterior density intervals

Alternative to equal-tail credible intervals are the highest posterior density (HPD) credible intervals reported in Komárek and Komárková (2011b, Table C.1). These can be calculated by the mean of the `HPDinterval` function from the **coda** package applied to above created objects `Regr`, `sigma1`, `SDb`, `Corb`. Output is shown only for the means of random effects $\boldsymbol{\beta}$ and the fixed effect α_3 stored in the object `Regr`.

```
R> HPDinterval(Regr)
```

```
                       lower        upper
lbili:Intcpt      0.206373622  0.430927560
lbili:Slope       0.004282489  0.011531590
platelet:Intcpt   5.482363696  5.569401950
platelet:Slope   -0.009019218 -0.004555172
spiders:Intcpt   -3.854407935 -1.976333749
spiders:Slope     0.002593894  0.053156233
attr(,"Probability")
[1] 0.95
```

```
R> HPDinterval(sigma1)
R> HPDinterval(SDb)
R> HPDinterval(Corb)
```

7.4. Estimated posterior densities

Further, we estimate posterior densities for GLMM model parameters using the `densplot` function from the **coda** package. First, we draw estimated posterior densities for the overall means $\boldsymbol{\beta}$ of random effects and the fixed effect α_3, all stored in the object `Regr`, see Figure 5.

```
R> COL <- rep(rainbow_hcl(3, start = 30, end = 210), each = 2)
R> par(mfcol = c(2, 3))
R> for (i in 1:6){
+    densplot(Regr[, i], show.obs = FALSE, col = COL[i], lwd = 2)
+    title(main = colnames(Regr)[i])
```

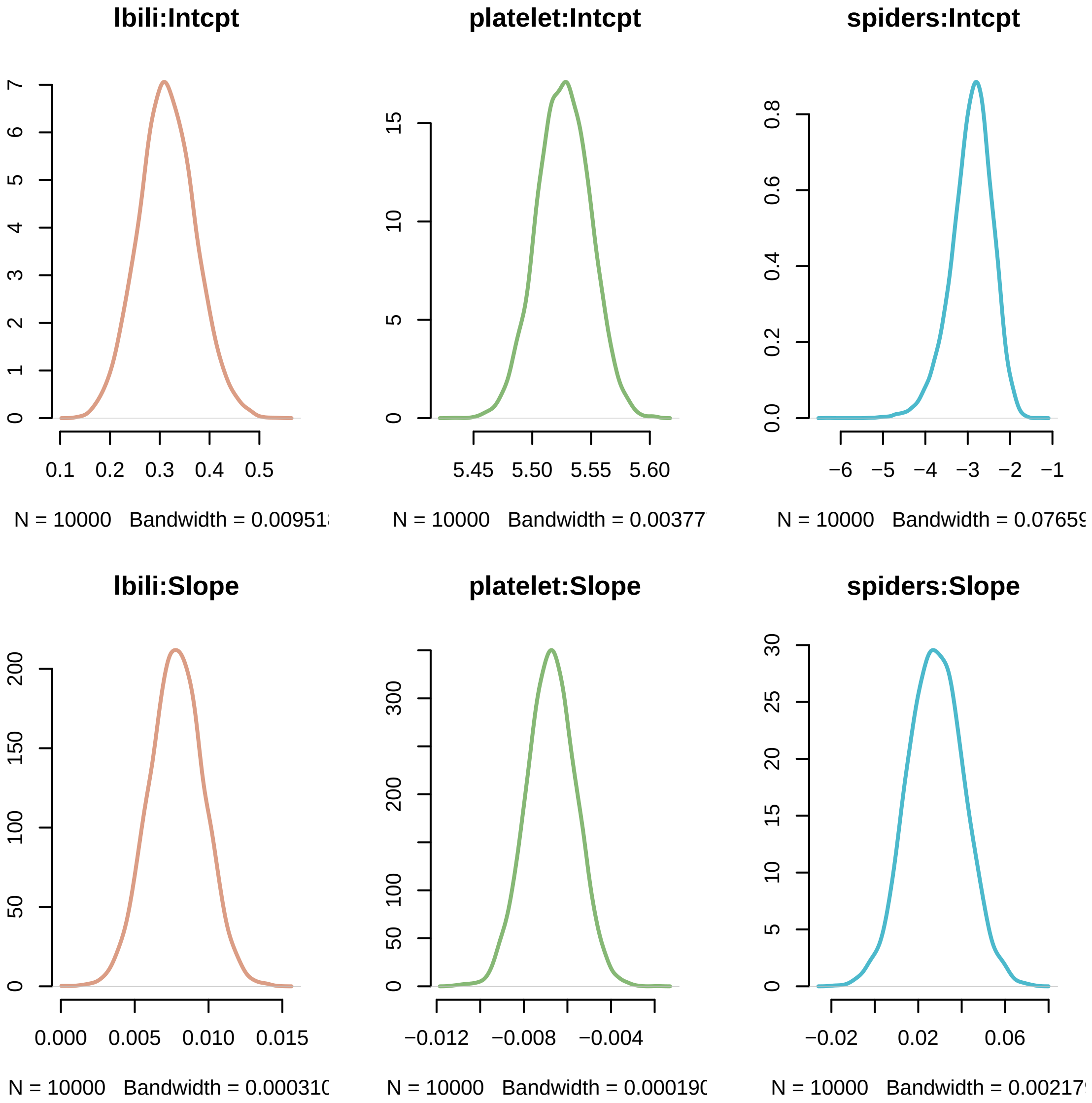

Figure 5: Estimated posterior densities of the overall means $\boldsymbol{\beta}$ of random effects and the fixed effect α_3.

```
+ }
```

Similarly, estimated posterior densities for other GLMM related parameters can be drawn.

7.5. Estimated longitudinal profiles

We use suitable posterior summary statistics $\widehat{\boldsymbol{\alpha}} = \widehat{\alpha}_3$ for fixed effects $\boldsymbol{\alpha}$ and analogous posterior summary statistics $\widehat{\boldsymbol{\beta}} = \bigl(\widehat{\beta}_{1,1}, \widehat{\beta}_{1,2}, \widehat{\beta}_{2,1}, \widehat{\beta}_{2,2}, \widehat{\beta}_3\bigr)^\top$ for the overall means $\boldsymbol{\beta}$ of random effects and calculate estimated longitudinal profiles of analyzed markers. The following code takes $t_{pred,1} = 0, t_{pred,2} = 0.3, \ldots, t_{pred,101} = 30$ and calculates the values of

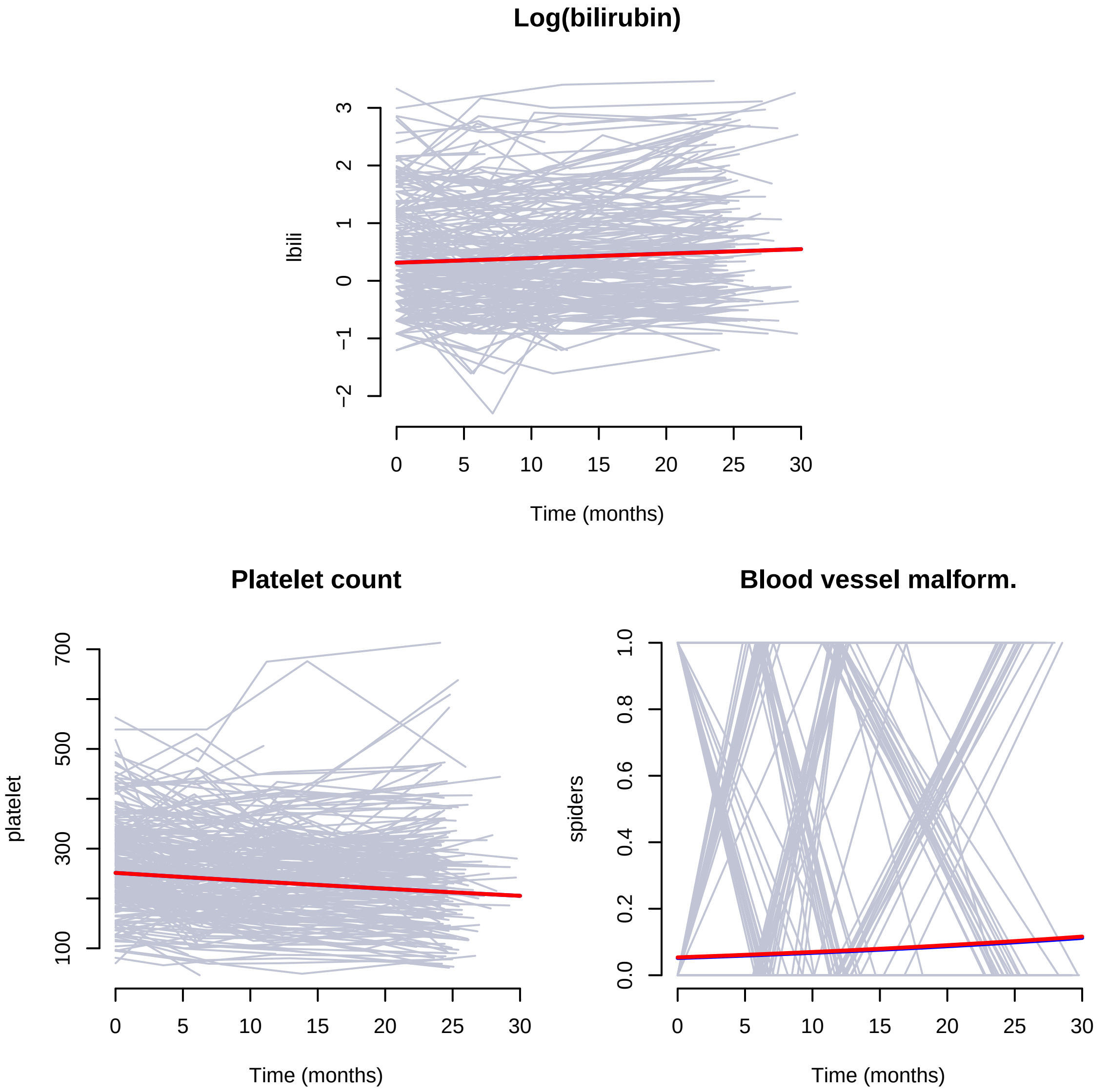

Figure 6: Observed longitudinal profiles of considered markers together with estimated longitudinal profiles based on posterior means (in blue) and posterior medians (in red) of fixed effects $\boldsymbol{\alpha}$ and means of random effects $\boldsymbol{\beta}$.

1. $\widehat{\beta}_{1,1} + \widehat{\beta}_{1,2} t_{pred,j}$, $j = 1, \ldots, 101$ (estimated longitudinal profile for logarithmic bilirubin);
2. $\exp\bigl(\widehat{\beta}_{2,1} + \widehat{\beta}_{2,2} t_{pred,j}\bigr)$, $j = 1, \ldots, 101$ (estimated longitudinal profile for platelet count);
3. $\text{logit}^{-1}\bigl(\widehat{\beta}_3 + \widehat{\alpha}_3 t_{pred,j}\bigr)$, $j = 1, \ldots, 101$ (estimated longitudinal profile for presence of blood vessel malformations).

Object `fitMean` is based on posterior means in place of $\widehat{\boldsymbol{\alpha}}$ and $\widehat{\boldsymbol{\beta}}$, object `fitMean` is based on posterior medians in place of $\widehat{\boldsymbol{\alpha}}$ and $\widehat{\boldsymbol{\beta}}$.

```
R> tpred <- seq(0, 30, by = 0.3)
R> fitMean <- fitted(mod[[1]], x = list("empty", "empty", tpred),
+                              z = list(tpred, tpred, "empty"),
+                              statistic = "mean", overall = TRUE)
R> fitMed <- fitted(mod[[1]], x = list("empty", "empty", tpred),
+                             z = list(tpred, tpred, "empty"),
+                             statistic = "median", overall = TRUE)
```

Both `fitMean` and `fitMed` are `list`s of length three (number of longitudinal markers). Each `list` component is a 101 × 1 matrix with calculated estimated longitudinal profiles for one marker. For instance, first 10 values of the estimated longitudinal profile for platelet count based on posterior means of regression coefficients are as follows.

```
R> print(fitMean[[2]][1:10,])
```

```
 [1] 251.3170 250.8091 250.3023 249.7964 249.2916 248.7877 248.2849
 [8] 247.7831 247.2824 246.7826
```

Subsequently, we plot observed longitudinal profiles (see the code on page 5) together with estimated longitudinal profiles, see Figure 6.

```
R> COL <- sequential_hcl(12, power = 2.2)[7]
R> COLmean <- "blue"
R> COLmed <- "red"
R> XLIM <- c(0, 910) / (365.25 / 12)
R> layout(autolayout(3))
R> #
R> plotProfiles(ip = ip, data = pbc01, var = "lbili", tvar = "month",
+     xlim = XLIM, xlab = "Time (months)", col = COL,
+     auto.layout = FALSE, main = "Log(bilirubin)")
R> lines(tpred, fitMean[[1]][,1], col = COLmean, lwd = 2)
R> lines(tpred, fitMed[[1]][,1], col = COLmed, lwd = 2)
R> #
R> plotProfiles(ip = ip, data = pbc01, var = "platelet", tvar = "month",
+     xlim = XLIM, xlab = "Time (months)", col = COL,
+     auto.layout = FALSE, main = "Platelet count")
R> lines(tpred, fitMean[[2]][,1], col = COLmean, lwd = 2)
```

```
R> lines(tpred, fitMed[[2]][,1], col = COLmed, lwd = 2)
R> #
R> plotProfiles(ip = ip, data = pbc01, var = "spiders",  tvar = "month",
+     xlim = XLIM, xlab = "Time (months)", col = COL,
+     auto.layout = FALSE, main = "Blood vessel malform.")
R> lines(tpred, fitMean[[3]][,1], col = COLmean, lwd = 2)
R> lines(tpred, fitMed[[3]][,1], col = COLmed, lwd = 2)
```

7.6. Posterior means of individual random effects

For situations when one is interested in inference on individuals, suitable estimates of individual values of random effects might be needed. Within Bayesian framework, posterior means of individidual random effects, i.e., values of $\mathsf{E}(\boldsymbol{b}_i\,|\,\boldsymbol{y})$, $i = 1, \ldots, N$ may serve this purpose. When using the **mixAK** package, MCMC based estimates $\widehat{\boldsymbol{b}}_i = M^{-1}\sum_{m=1}^{M}\boldsymbol{b}_i^{(m)}$ of $\mathsf{E}(\boldsymbol{b}_i\,|\,\boldsymbol{y})$, $i = 1, \ldots, N$, are available in the `poster.mean.profile` of the object `mod` obtained by running the `GLMM_MCMC` routine. It is a matrix with N rows and values of $\widehat{\boldsymbol{b}}_i$ in its initial columns. We extract them and print first 10 values.

```
R> bhat <- mod[[1]]$poster.mean.profile[, 1:mod[[1]]$dimb]
R> print(bhat[1:10,])
```

```
            b1          b2       b3           b4         b5
1   0.01941317 0.013573928 5.370993 -0.022027877  2.4033087
2   0.25963476 0.009724842 5.045187 -0.016060771 -0.5045080
3   0.50083022 0.015408244 5.365412  0.004046026  2.7069998
4   0.87940689 0.021633816 4.881798 -0.016878592 -0.2418984
5  -0.25366522 0.004090812 5.706338 -0.001027675 -5.7065666
6  -0.07411346 0.003195730 5.349503 -0.006966620 -5.8347617
7  -0.68313577 0.006880460 5.900706  0.001604459 -6.4214860
8   1.32774590 0.042383755 5.554436 -0.001197036 -1.3364364
9   0.26097685 0.010561914 5.611316 -0.003852797 -1.8181445
10 -0.27814326 0.007823518 5.524950 -0.004782997 -0.7026804
```

8. Residual plots

Model checking with respect to the assumed relationship (1) between the mean response and covariates is classically performed by a suitable residual analysis. To allow this, the resulting object `mod` contains a component called `poster.mean.y` which is a `list` (in our case of length 3) with a `data.frame` for each response marker. We print the first three rows of each of them.

```
R> print(mod[[1]]$poster.mean.y$lbili[1:3,])
```

```
  id    observed     fitted      stres eta.fixed eta.random
1  1  0.09531018 0.01941317  0.2438135         0 0.01941317
2  1 -0.22314355 0.10057802 -1.0288467         0 0.10057802
3  1  0.00000000 0.18218883 -0.5784067         0 0.18218883
```

```
R> print(mod[[1]]$poster.mean.y$platelet[1:3,])
```

```
  id observed   fitted      stres eta.fixed eta.random
1  1      221 215.3776  0.4040622         0   5.370993
2  1      188 188.6808 -0.0388953         0   5.239278
3  1      161 165.2801 -0.3228092         0   5.106840
```

```
R> print(mod[[1]]$poster.mean.y$spiders[1:3,])
```

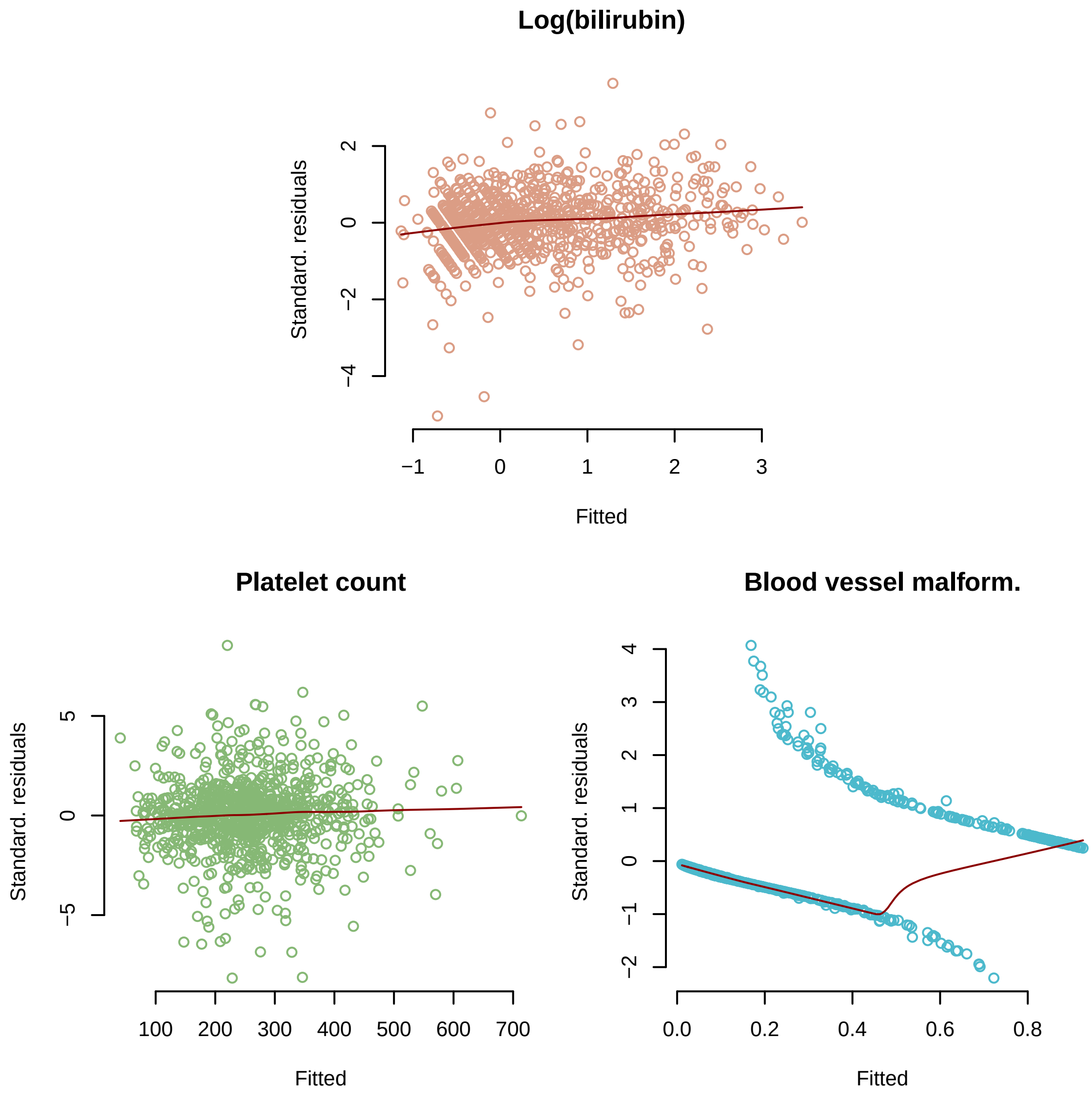

Figure 7: Plots of posterior means of standardized residuals against posterior means of fitted values.

```
  id observed    fitted     stres eta.fixed eta.random
1  1        1 0.8403518 0.4119686 0.0000000   2.403309
2  1        1 0.8580191 0.3783572 0.1675878   2.403309
3  1        1 0.8740891 0.3478410 0.3360963   2.403309
```

Each `data.frame` contains identification of the subject (column `id`, taken from the `id` argument of the original `GLMM_MCMC` function call), observed response values (column `observed`), posterior means of model based fitted values (column `fitted`) which are based on sampled values of $\lambda_{i,1,j}$, $\lambda_{i,2,j}$, $\lambda_{i,3,j}$, respectively (see Eq. (10)–(12)) Further, we have a column `stres` containing posterior means of model based standardized residuals $e_{i,r,j}$ where

$$e_{i,1,j} = \frac{y_{i,1,j} - \lambda_{i,1,j}}{\sigma_1}, \qquad e_{i,2,j} = \frac{y_{i,2,j} - \lambda_{i,2,j}}{\sqrt{\lambda_{i,2,j}}}, \qquad e_{i,3,j} = \frac{y_{i,3,j} - \lambda_{i,3,j}}{\sqrt{\lambda_{i,3,j}(1 - \lambda_{i,3,j})}},$$

$i = 1, \ldots, N$, $j = 1, \ldots, n_{i,r}$, $r = 1, 2, 3$. Finally, columns `eta.fixed` and `eta.random` contain posterior means of fixed effects and random effects related parts of the linear predictors (right-hand sides of expressions in Eq. (1)). We use these objects to draw basic residual plots of posterior means of standardized residuals against posterior means of fitted values accompanied by the lowess (Cleveland 1979) line, see Figure 7.

```
R> COL <- rainbow_hcl(3, start = 30, end = 210)
R> MAIN <- c("Log(bilirubin)", "Platelet count", "Blood vessel malform.")
R> layout(autolayout(3))
R> for (i in 1:3){
+    plot(mod[[1]]$poster.mean.y[[i]]$fitted, mod[[1]]$poster.mean.y[[i]]$stres,
+         xlab = "Fitted", ylab = "Standard. residuals", col = COL[i])
+    lines(lowess(mod[[1]]$poster.mean.y[[i]]$fitted, mod[[1]]$poster.mean.y[[i]]$stres),
+           col = "red4")
+    title(main = MAIN[i])
+  }
```

9. Estimated distribution of random effects

Before we proceed to the main objective of our approach which is clustering, we explore how much the actual distribution (2) of random effects which serves as a basis for clustering differs from conventional one-component normal distribution and how much the data support our assumption of having at least two groups of patients. Within the Bayesian framework, basically two approaches exist to estimate the density of random effects. The first one leads to the plug-in estimate of the random effects density given by

$$\widetilde{p}(\boldsymbol{b}) = p\bigl(\boldsymbol{b} \,\big|\, \widehat{\boldsymbol{\theta}}\bigr), \tag{15}$$

where $\widehat{\boldsymbol{\theta}}$ is a suitable estimate, e.g., the posterior mean, of mixture parameters. Nevertheless, it is clear that $\widetilde{p}(\boldsymbol{b})$ is not invariant towards label switching and hence it is not always a good estimate of the random effect density. Alternatively, we may consider posterior predictive density

$$p_{PP}(\boldsymbol{b}) = \int p(\boldsymbol{b} \,|\, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \,|\boldsymbol{y})\, d\boldsymbol{\theta},$$

and its MCMC estimated counterpart

$$\widehat{p}(\boldsymbol{b}) = \frac{1}{M} \sum_{m=1}^{M} p(\boldsymbol{b} \,|\, \boldsymbol{\theta}^{(m)}),$$

which are both invariant against label switching. Its disadvantage when clustering is of interest, though, is the fact that we cannot extract weights, means or covariance matrices that specify the mixture components since the random effect density is now considered as one functional parameter which is directly estimated. In any case, agreement between $\widetilde{p}(\boldsymbol{b})$ and $\widehat{p}(\boldsymbol{b})$ indicates a successful relabelling of the Markov chain and into some extent guarantees meaningfulness of subsequent clustering procedure.

In package **mixAK**, several routines are available to calculate plug-in and posterior predictive densities of random effects and to calculate related quantities. First,

- `NMixSummComp()` is a generic function with a method for objects of class `GLMM_MCMC` which prints posterior means of mixture weights $\boldsymbol{w}$, shifted and scaled mixture means $\boldsymbol{s} + \boldsymbol{S}\boldsymbol{\mu}_k$, $k = 1, \ldots, K$ and scaled mixture covariance matrices $\boldsymbol{S}\boldsymbol{D}_k$, $k = 1, \ldots, K$ calculated under relabelling reflected in the `order_b` and `rank_b` components of this object.

Second, a series of generic functions with a method for objects of class `GLMM_MCMC` is implemented to calculate the values of the plug-in or posterior predictive estimate of the random effect density or related cdf evaluated in a specific grid of $\boldsymbol{b}$ values. For plug-in estimates we have routines

- `NMixPlugDensMarg()` which calculates estimates of all univariate marginal densities;
- `NMixPlugDensJoint2()` which calculate estimates of all joint bivariate marginal densities;
- `NMixPlugCondDensMarg()` which calculates estimates of all univariate conditional densities given a specified margin;
- `NMixPlugCondDensJoint2()` which calculates estimates of all bivariate conditional densities given a specified margin.

For posterior predictive estimates the following functions are available.

- `NMixPredDensMarg()` which calculates estimates of all univariate marginal densities;
- `NMixPredDensJoint2()` which calculate estimates of all joint bivariate marginal densities;
- `NMixPredCDFMarg()` which calculates estimates of all univariate marginal cdf's;
- `NMixPredCondDensMarg()` which calculates estimates of all univariate conditional densities given a specified margin;
- `NMixPredCondDensJoint2()` which calculates estimates of all bivariate conditional densities given a specified margin.

For results created by each of above mentioned functions a specific `plot` method exists to visualize calculated density of cdf.

9.1. Posterior means of mixture parameters

First, we check posterior means of mixture parameters calculated after relabelling which is reflected in the `order_b` and `rank_b` components of the object `mod`. The posterior means of mixture weights $\boldsymbol{w}$, mixture means $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ and mixture covariance matrices $\boldsymbol{D}_1, \boldsymbol{D}_2$ are stored as components `poster.mean.w_b`, `poster.mean.mu_b`, `poster.mean.Sigma_b`, respectively and we can directly print them out.

```
R> print(mod[[1]]$poster.mean.w_b)
```

```
       w1        w2 
0.5976874 0.4023126 
```

```
R> print(mod[[1]]$poster.mean.mu_b)
```

```
           m1         m2         m3          m4         m5
j1 -0.6059869 -0.1571325  0.1441811  0.06146376 -0.4890238
j2  0.9105904  0.2568817 -0.1947171 -0.10519656  0.5930209
```

Posterior mean of $\boldsymbol{\mu}_1$ is found in the first row of the matrix above, posterior mean of $\boldsymbol{\mu}_2$ in the second row of the matrix above.

```
R> print(mod[[1]]$poster.mean.Sigma_b)
```

```
$j1
            [,1]        [,2]        [,3]        [,4]        [,5]
[1,]  0.24459564  0.00644807 -0.12355838 -0.02873796  0.20088303
[2,]  0.00644807  0.17360912  0.01483741 -0.06008096  0.05197403
[3,] -0.12355838  0.01483741  0.78491511 -0.02316504 -0.04598325
[4,] -0.02873796 -0.06008096 -0.02316504  0.45322903  0.02377058
[5,]  0.20088303  0.05197403 -0.04598325  0.02377058  1.55189552

$j2
           [,1]        [,2]        [,3]        [,4]        [,5]
[1,]  0.8057164 -0.25230233  0.12184208 -0.18518042  0.11486954
[2,] -0.2523023  2.36936449 -0.05926528  0.56948008  0.13339433
[3,]  0.1218421 -0.05926528  1.30335249 -0.07748660 -0.16362231
[4,] -0.1851804  0.56948008 -0.07748660  2.20520552 -0.04737602
[5,]  0.1148695  0.13339433 -0.16362231 -0.04737602  0.56102968
```

Similarly, there are additional components `poster.mean.Q_b` and `poster.mean.Li_b` attached to the objects `mod[[1]]` and `mod[[2]]` which provide posterior means of inverted mixture covariance matrices $\boldsymbol{D}_1^{-1}, \boldsymbol{D}_2^{-1}$ and posterior means of their Cholesky decompositions, respectively.

With respect to interpretation, posterior summary statistics for shifted and scaled mixture means, that is, for $\boldsymbol{s} + \boldsymbol{S}\boldsymbol{\mu}_k$, $k = 1, \dots, K$ and for scaled mixture covariance matrices, that is, for $\boldsymbol{S}\boldsymbol{D}_k$, $k = 1, \dots, K$, are more important as they directly correspond to the data at hand. To get these, we may use the `NMixSummComp` function which additionally prints standard deviations and correlation matrices calculated from the posterior means of scaled mixture covariance matrices $\boldsymbol{S}\boldsymbol{D}_k$, $k = 1, \dots, K$.

```
R> NMixSummComp(mod[[1]])
```

```

Component 1
    Weight:  0.5976874
    Mean:    -0.2087128 0.004500078 5.576472 -0.005671868 -4.328348

    Covariance matrix:
              m1            m2            m3            m4           m5
m1  0.1827977252  1.119111e-04 -0.0372359555 -3.888952e-04  0.560665583
m2  0.0001119111  6.997409e-05  0.0001038412 -1.888143e-05  0.003368747
m3 -0.0372359555  1.038412e-04  0.0953852528 -1.264089e-04 -0.051752172
m4 -0.0003888952 -1.888143e-05 -0.0001264089  1.110577e-04  0.001201310
m5  0.5606655826  3.368747e-03 -0.0517521718  1.201310e-03 16.175649169
    Standard deviations:  0.4275485 0.008365051 0.308845 0.01053839 4.021896

    Correlation matrix:
            m1          m2          m3          m4          m5
m1  1.00000000  0.03129097 -0.28199181 -0.08631231  0.32605258
m2  0.03129097  1.00000000  0.04019393 -0.21418634  0.10013108
m3 -0.28199181  0.04019393  1.00000000 -0.03883853 -0.04166363
m4 -0.08631231 -0.21418634 -0.03883853  1.00000000  0.02834325
m5  0.32605258  0.10013108 -0.04166363  0.02834325  1.00000000

---------------------------------------------

Component 2
    Weight:  0.4023126
    Mean:    1.102356 0.01281193 5.458331 -0.00828071 -0.8349765

    Covariance matrix:
             m1            m2            m3            m4          m5
m1  0.602149424 -0.0043788961  0.0367187258 -0.0025059457  0.320601486
m2 -0.004378896  0.0009549851 -0.0004147744  0.0001789684  0.008646082
m3  0.036718726 -0.0004147744  0.1583873282 -0.0004228355 -0.184149866
m4 -0.002505946  0.0001789684 -0.0004228355  0.0005403563 -0.002394274
```

```
m5  0.320601486  0.0086460821 -0.1841498657 -0.0023942737  5.847699884
    Standard deviations:  0.7759829 0.03090283 0.3979791 0.02324556 2.418202

    Correlation matrix:
           m1          m2          m3          m4          m5
m1  1.0000000 -0.18260565  0.11889819 -0.13892469  0.17085231
m2 -0.1826057  1.00000000 -0.03372511  0.24913690  0.11569871
m3  0.1188982 -0.03372511  1.00000000 -0.04570578 -0.19134568
m4 -0.1389247  0.24913690 -0.04570578  1.00000000 -0.04259329
m5  0.1708523  0.11569871 -0.19134568 -0.04259329  1.00000000

---------------------------------------------
```

Above calculated results are also shown in Komárek and Komárková (2011b, Table C.3).

9.2. Univariate marginal densities

Second, we calculate and plot the univariate marginal densities of random effects derived from both the plug-in estimate $\widetilde{p}(\boldsymbol{b})$ and the estimated posterior predictive density $\widehat{p}(\boldsymbol{b})$. We calculate the values of margins of $\widetilde{p}(\boldsymbol{b})$ and $\widehat{p}(\boldsymbol{b})$ evaluated in automatically selected grid (calculated using the information on posterior means of $\boldsymbol{\beta}$, expression (6) and components of $\boldsymbol{D}$, expression (7)) of $\boldsymbol{b}$ values and store them in objects `plugdm` and `pdm`, respectively.

```
R> plugdm <- list()
R> plugdm[[1]] <- NMixPlugDensMarg(mod[[1]])
R> plugdm[[2]] <- NMixPlugDensMarg(mod[[2]])
R> pdm <- list()
R> pdm[[1]] <- NMixPredDensMarg(mod[[1]])
R> pdm[[2]] <- NMixPredDensMarg(mod[[2]])
```

We can use the `plot` method and create basic plots of calculated estimated marginal densities, separately for chain 1 and 2 (output not shown).

```
R> plot(plugdm[[1]])
R> plot(plugdm[[2]])
R> plot(pdm[[1]])
R> plot(pdm[[2]])
```

Main calculated results are stored as `x` and `dens` elements of objects `plugdm[[*]]` and `pdm[[*]]`, respectively and we may use them to create our own plots. Figure 8 shows that the plug-in estimates closely correspond to the estimated posterior predictive densities of individual margins of random effects which indicates meaningfulness of posterior means of mixture weights, means and components of covariance matrices shown in Section 9.1. Additionally, it is clear that one-component normal distribution would be unsatisfactory to model the distribution of random effects in this application.

```
R> layout(autolayout(mod[[1]]$dimb))
R> blab <- c("Intcpt (lbili)", "Slope (lbili)", "Intcpt (platelet)",
+            "Slope (platelet)", "Intcpt (spiders)")
R> for (i in 1:mod[[1]]$dimb){
+    plot(plugdm[[1]]$x[[i]], plugdm[[1]]$dens[[i]], type = "l",
+         xlab = "b", ylab = "Density", col = "red", main = blab[i],
+         ylim = c(0, max(pdm[[1]]$dens[[i]])))
+    lines(pdm[[1]]$x[[i]], pdm[[1]]$dens[[i]], col = "darkblue")
+  }
```

The values in which the densities are evaluated and subsequently plotted can be changed by the user by specifying the `grid` argument in `NMixPlugDensMarg` and `NMixPredDensMarg` functions.

```
R> bgrid <- list(b1 = seq(-2.73, 3.37, length = 50),
+                b2 = seq(-0.07, 0.08, length = 50),
+                b3 = seq(4.28, 6.77, length = 50),
+                b4 = seq(-0.06, 0.05, length = 50),
```

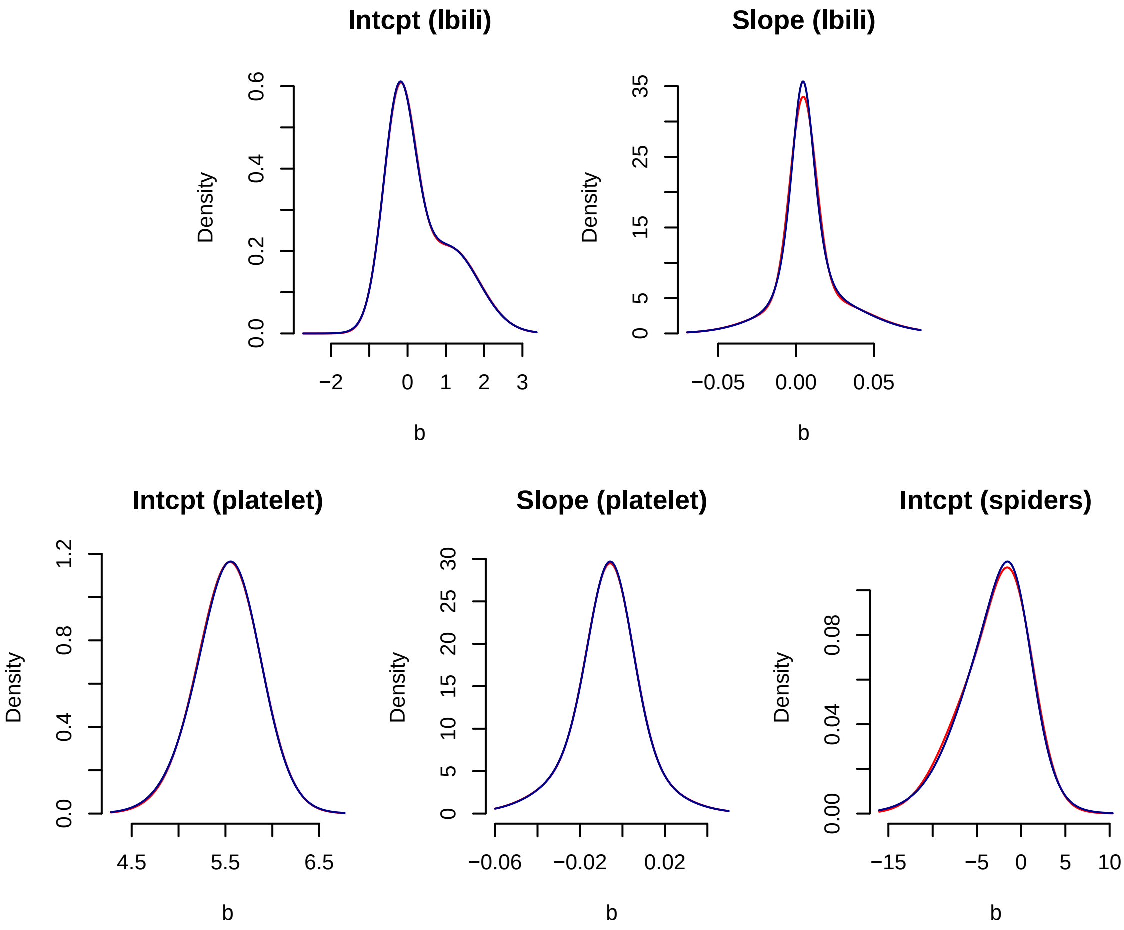

Figure 8: Estimated univariate marginal posterior predictive densities of random effects (in blue) and corresponding plug-in estimates (in red).

```
+                b5 = seq(-16.05, 10.34, length = 50))
R> plugdm <- list()
R> plugdm[[1]] <- NMixPlugDensMarg(mod[[1]], grid = bgrid)
R> plugdm[[2]] <- NMixPlugDensMarg(mod[[2]], grid = bgrid)
R> pdm <- list()
R> pdm[[1]] <- NMixPredDensMarg(mod[[1]], grid = bgrid)
R> pdm[[2]] <- NMixPredDensMarg(mod[[2]], grid = bgrid)
```

9.3. Joint bivariate densities

Similarly to estimates of univariate marginal densities of random effects, analogous estimates of pairwise joint bivariate marginal densities can easily be calculated using the functions `NMixPlugDensJoint2` and `NMixPredDensJoint2`, respectively. We calculate their values evaluated in automatically created grid of $\boldsymbol{b}$ values and store them in objects `plugdj01` and `pdj01`, respectively. Subsequently, we produce their basic image plots (output not shown).

```
R> plugdj <- list()
R> plugdj[[1]] <- NMixPlugDensJoint2(mod[[1]])
R> plugdj[[2]] <- NMixPlugDensJoint2(mod[[2]])
R> pdj <- list()
R> pdj[[1]] <- NMixPredDensJoint2(mod[[1]])
R> pdj[[2]] <- NMixPredDensJoint2(mod[[2]])
R> plot(plugdj[[1]])
R> plot(plugdj[[2]])
R> plot(pdj[[1]])
R> plot(pdj[[2]])
```

Figure 9 shows image and contour plots of estimated pairwise joint bivariate marginal posterior predictive densities of random effects which was created by using results stored as `x` and `dens` components of the object `pdj01`.

```
R> layout(matrix(c(1:9, 0, 10, 0), ncol = 3, byrow = TRUE))
R> for (i in 1:(mod[[1]]$dimb - 1)){
+   for (j in (i+1):mod[[1]]$dimb){
+     image(pdj[[1]]$x[[i]], pdj[[1]]$x[[j]], pdj[[1]]$dens[[paste(i, "-", j, sep = "")]]
+        col = rev(heat_hcl(33, c = c(80, 30), l = c(30, 90), power = c(1/5, 1.3))),
+        xlab = blab[i], ylab = blab[j])
+     contour(pdj[[1]]$x[[i]], pdj[[1]]$x[[j]], pdj[[1]]$dens[[paste(i, "-", j, sep = "")
+          col = "brown", add = TRUE)
+   }
+ }
```

Even for estimates of joint bivariate marginal densities of random effects, the user may specify the grid of $\boldsymbol{b}$ values in which the densities are evaluated.

```
R> pdj[[1]]    <- NMixPredDensJoint2(mod[[1]], grid = bgrid)
R> pdj[[2]]    <- NMixPredDensJoint2(mod[[2]], grid = bgrid)
R> plugdj[[1]] <- NMixPlugDensJoint2(mod[[1]], grid = bgrid)
R> plugdj[[2]] <- NMixPlugDensJoint2(mod[[2]], grid = bgrid)
```

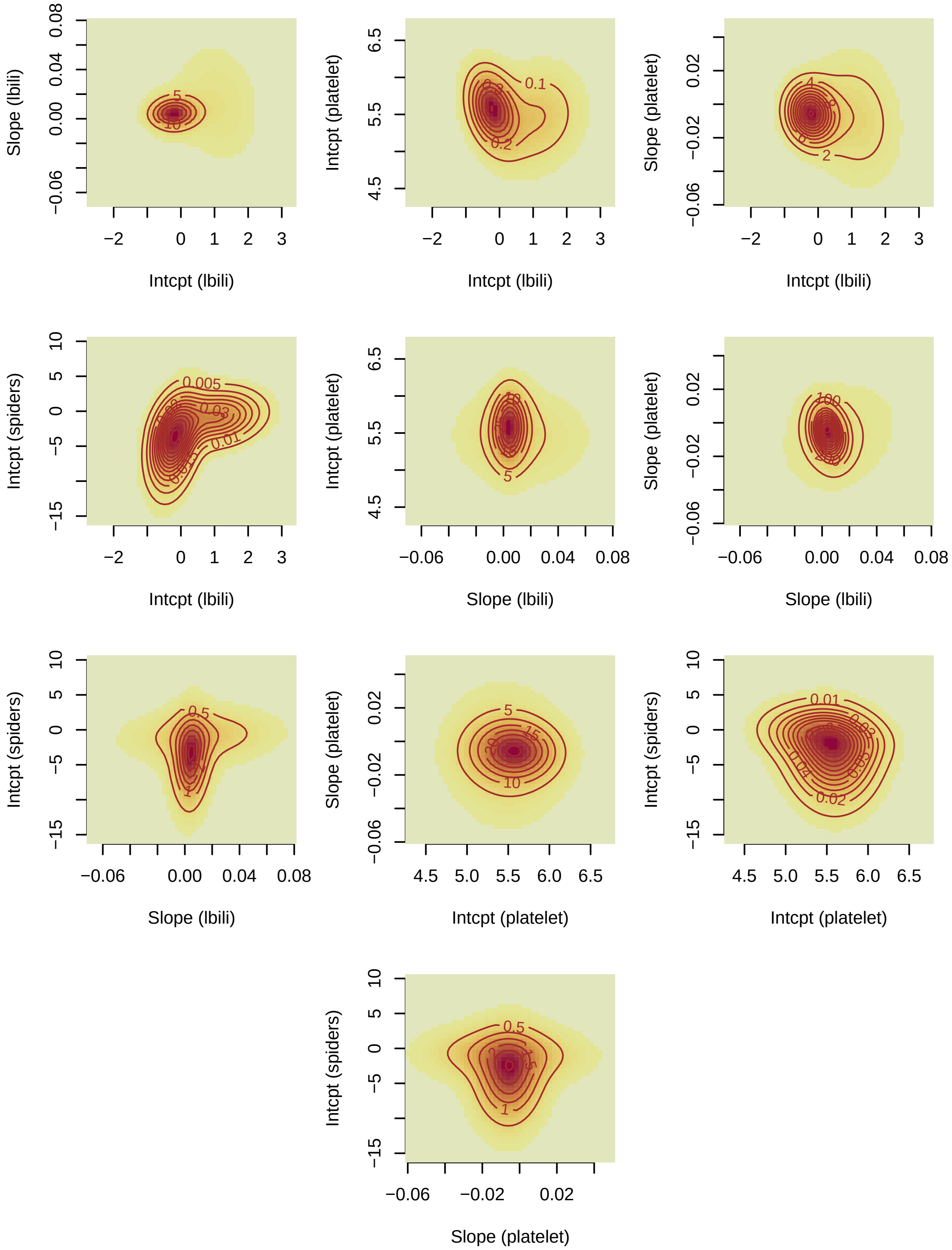

Figure 9: Estimated pairwise joint bivariate marginal posterior predictive densities of random effects.

10. Clustering

10.1. Posterior distribution of component probabilities

It is explained in Komárek and Komárková (2011a, Sec. 2.5) that clustering, i.e., classification into groups represented by mixture components, is based on posterior distribution of component probabilities

$$p_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta}) = \mathsf{P}(u_i = k\,|\,\boldsymbol{\psi},\,\boldsymbol{\theta},\,\boldsymbol{y}) = \mathsf{P}(u_i = k\,|\,\boldsymbol{\psi},\,\boldsymbol{\theta},\,\boldsymbol{y}_i) = \frac{L_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta})\,w_k}{\sum_{l=1}^{K} L_{i,l}(\boldsymbol{\psi},\,\boldsymbol{\theta})\,w_l}, \qquad i = 1,\dots,N,\ k = 1,\dots,K, \tag{16}$$

where $L_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta})$ is given by (9). In particular, we are interested in corresponding posterior means

$$\pi_{i,k} = \mathsf{P}(u_i = k\,|\,\boldsymbol{y}) = \int p_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta})\,p(\boldsymbol{\psi},\,\boldsymbol{\theta}\,|\,\boldsymbol{y})\,d(\boldsymbol{\psi},\boldsymbol{\theta}), \qquad i = 1,\dots,N, k = 1,\dots,K, \tag{17}$$

estimated from the MCMC simulation as

$$\widehat{\pi}_{i,k} = M^{-1}\sum_{m=1}^{M} p_{i,k}\bigl(\boldsymbol{\psi}^{(m)},\,\boldsymbol{\theta}^{(m)}\bigr), \qquad i = 1,\dots,N,\ k = 1,\dots,K, \tag{18}$$

or alternatively in posterior medians estimated from the MCMC simulation as

$$\widetilde{\pi}_{i,k} = \underset{m=1,\dots,M}{\text{sample median}}\, p_{i,k}\bigl(\boldsymbol{\psi}^{(m)},\,\boldsymbol{\theta}^{(m)}\bigr), \qquad i = 1,\dots,N,\ k = 1,\dots,K. \tag{19}$$

Additionally, it is useful to quantify the variability of the posterior distribution of each $p_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta})$ which can be done by reporting corresponding credible intervals.

Upon running `NMixRelabel` routine (see Section 5.8), objects `mod[[1]]` and `mod[[2]]` contain components `poster.comp.prob3`, `quant.comp.prob3` and `comp.prob3` which serve as basis for reporting or calculation of posterior means, posterior medians or credible intervals for component probabilities $p_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta})$, $i = 1,\dots,N, k = 1,\dots,K$. Note that the labels of components pertain to the result of a particular relabelling routine invoked by the `NMixRelabel` routine and also correspond to posterior means of mixture parameters provided by the `NMixSummComp` routine (see Section 9.1).

First, `poster.comp.prob3` is $N \times K$ matrix which contains the values of estimated posterior means $\widehat{\pi}_{i,k}$, $i = 1,\dots,N, k = 1,\dots,K$.

```
R> print(mod[[1]]$poster.comp.prob3[1:5,])
```

```
          [,1]       [,2]
[1,] 0.7170372 0.28296276
[2,] 0.5803136 0.41968644
[3,] 0.5305126 0.46948738
[4,] 0.1325851 0.86741491
[5,] 0.9883736 0.01162644
```

Second, `quant.comp.prob3` is a `list` with $N \times K$ matrices as its components.

```
R> names(mod[[1]]$quant.comp.prob3)
```

```
[1] "2.5%"  "50%"   "97.5%"
```

```
R> print(mod[[1]]$quant.comp.prob3[["50%"]][1:5,])
```

```
          [,1]        [,2]
[1,] 0.8266722 0.173327823
[2,] 0.6231642 0.376835802
[3,] 0.5674317 0.432568268
[4,] 0.0428354 0.957164597
[5,] 0.9902083 0.009791726
```

Each matrix provides p100% posterior quantiles of $p_{i,k}(\boldsymbol{\psi},\, \boldsymbol{\theta})$ for selected values of p. By default, 2.5%, 50%, and 97.5% are provided. Quantiles for other values of p can be calculated by running the `NMixRelabel` function with appropriately modified value of its `prob` argument. Due to the fact that the posterior distribution of each $p_{i,k}(\boldsymbol{\psi},\, \boldsymbol{\theta})$ is usually skewed (see Figure 10), Komárek and Komárková (2011a, Sec. 2.5) preferred to use posterior medians ($\widetilde{\pi}_{i,k}$, `mod$quant.comp.prob3[["50%"]]`) rather than posterior means ($\widehat{\pi}_{i,k}$, `mod$poster.comp.prob3`) for classification. Note that objects `mod$quant.comp.prob3[["2.5%"]]` and `mod$quant.comp.prob3[["97.5%"]]` directly provide lower and upper limits for 95% equal-tail credible intervals for $p_{i,k}(\boldsymbol{\psi},\, \boldsymbol{\theta})$. The whole posterior distribution of each $p_{i,k}(\boldsymbol{\psi},\, \boldsymbol{\theta})$, $i = 1, \dots, N, k = 1, \dots, K$ can be explored by processing the `comp.prob3` component of object `mod`.

```
R> print(mod[[1]]$comp.prob3[1:5, 1:10])
```

```
        P(1,1)     P(1,2)    P(2,1)     P(2,2)     P(3,1)    P(3,2)
[1,] 0.6458363 0.35416371 0.7162301 0.28376989 0.51795020 0.4820498
[2,] 0.8258937 0.17410632 0.4011532 0.59884680 0.17718107 0.8228189
[3,] 0.9792811 0.02071886 0.9640948 0.03590523 0.84568169 0.1543183
[4,] 0.3911205 0.60887949 0.4283078 0.57169223 0.37407874 0.6259213
[5,] 0.5422908 0.45770924 0.6198451 0.38015488 0.01935869 0.9806413
          P(4,1)    P(4,2)    P(5,1)      P(5,2)
[1,] 0.041307093 0.9586929 0.9979254 0.002074617
[2,] 0.005138986 0.9948610 0.9951290 0.004871005
[3,] 0.631041614 0.3689584 0.9803490 0.019650991
[4,] 0.038126937 0.9618731 0.9899264 0.010073608
[5,] 0.037846162 0.9621538 0.9800216 0.019978446
```

It is $M \times N \cdot K$ matrix with sampled values of $p_{1,1}\big(\boldsymbol{\psi}^{(m)},\,\boldsymbol{\theta}^{(m)}\big)$, …, $p_{1,K}\big(\boldsymbol{\psi}^{(m)},\,\boldsymbol{\theta}^{(m)}\big)$, …, $p_{N,1}\big(\boldsymbol{\psi}^{(m)},\,\boldsymbol{\theta}^{(m)}\big)$, …, $p_{N,K}\big(\boldsymbol{\psi}^{(m)},\,\boldsymbol{\theta}^{(m)}\big)$ in rows.

To show skewness of the posterior distributions of component probabilities $p_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta})$ and also varying variability, we take appropriate columns from `mod$comp.prob3` and draw histograms of sampled values of $p_{i,1}(\boldsymbol{\psi},\,\boldsymbol{\theta})$ for three selected patients (different from those shown on Figure 3 in Komárek and Komárková 2011a), see Figure 10.

```
R> IDS <- unique(pbc01$id)
R> K <- mod[[1]]$prior.b$Kmax
R> N <- ncol(mod[[1]]$comp.prob3) / K
R> ID <- c(3, 7, 51)
R> par(mfrow = c(1, 3))
R> for (id in ID){
+    i <- (1:N)[IDS == id]
+    hist(mod[[1]]$comp.prob3[, (i - 1) * K + 1], xlim = c(0, 1), prob = TRUE,
+       xlab = expression(paste("P(u=1|", psi, ", ", theta, ", y)", sep = "")),
+       col = heat_hcl(12, c = c(80, 30), l = c(30, 90), power = c(1/5, 2))[12],
+       main = paste("ID", id))
+  }
```

Additionally, highest posterior density (HPD) credible intervals of component probabilities $p_{i,k}(\boldsymbol{\psi},\,\boldsymbol{\theta})$ can easily be calculated using the `HPDinterval` routine from the **coda** package which allows to evaluate uncertainty in classification of individual patients.

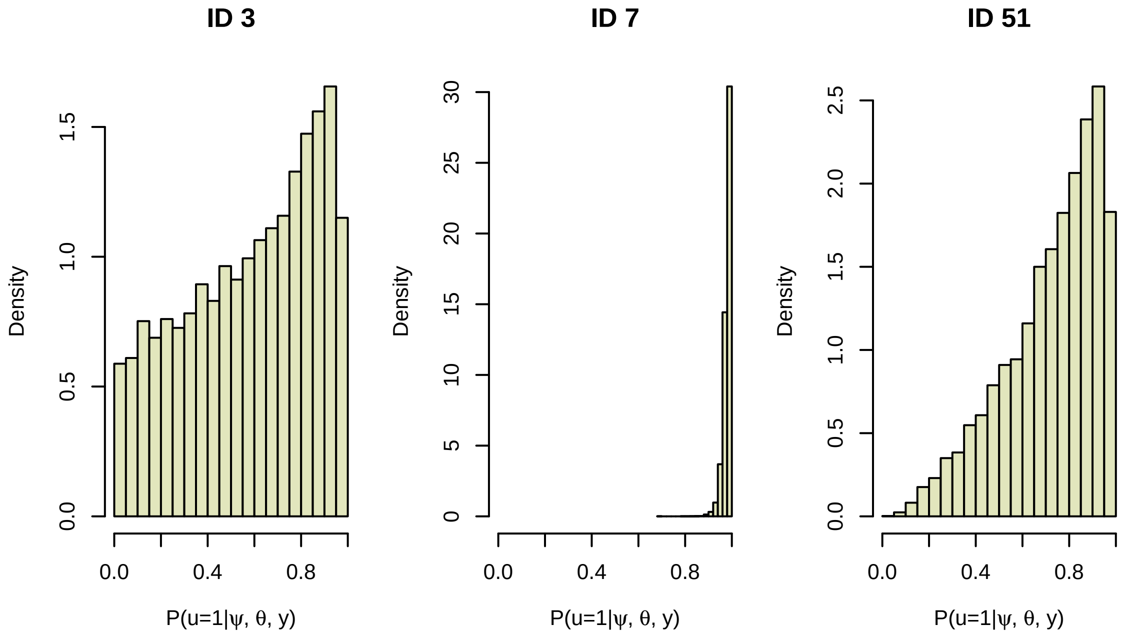

Figure 10: Histograms of sampled values of component probabilities $p_{i,1}(\boldsymbol{\psi},\,\boldsymbol{\theta})$ for three selected patients.

```
R> prob3HPD <- HPDinterval(mcmc(mod[[1]]$comp.prob3))
R> rownames(prob3HPD) <- paste("ID", rep(IDS, each = K), ", k = ", 1:K, ":", sep = "")
R> print(prob3HPD[1:6,])
```

```
                   lower     upper
ID2, k = 1: 0.1116547364 0.9997078
ID2, k = 2: 0.0002921897 0.8883453
ID3, k = 1: 0.0805366250 0.9884059
ID3, k = 2: 0.0115940575 0.9194634
ID4, k = 1: 0.0187111309 0.9447680
ID4, k = 2: 0.0552320442 0.9812889
```

We check posterior means, medians and HPD credible intervals of component probabilities $p_{i,1}(\boldsymbol{\psi},\, \boldsymbol{\theta})$ for patients selected for Figure 10.

```
R> Row <- (1:N)[IDS %in% ID]
R> Mean   <- mod[[1]]$poster.comp.prob3[Row, 1]
R> Median <- mod[[1]]$quant.comp.prob3[["50%"]][Row, 1]
R> HPD    <- prob3HPD[(Row - 1) * K + 1,]
R> Pshow <- data.frame(Mean = Mean, Median = Median,
+                      HPD.lower = HPD[, 1], HPD.upper = HPD[, 2])
R> print(Pshow)
```

```
                  Mean    Median  HPD.lower HPD.upper
ID3, k = 1:  0.5803136 0.6231642 0.08053663 0.9884059
ID7, k = 1:  0.9793557 0.9837618 0.94798459 0.9991267
ID51, k = 1: 0.7227904 0.7692293 0.31930466 0.9951896
```

Objects `mod[[1]]` and `mod[[2]]` additionally contain related components called `poster.comp.prob1`, `poster.comp.prob2`, `quant.comp.prob2` and `comp.prob2`. First, the structure of components `poster.comp.prob1`, `poster.comp.prob2` is the same as structure of the component `poster.comp.prob3`. They all contain estimated values of $\pi_{i,k} = \mathsf{P}(u_i = k \,|\, \boldsymbol{y})$, $i = 1, \ldots, N$, $k = 1, \ldots, K$, nevertheless, calculated in different ways stemming from different representation of $\mathsf{P}(u_i = k \,|\, \boldsymbol{y})$, i.e.,

$$\mathsf{P}(u_i = k \,|\, \boldsymbol{y}) = \mathsf{E}\bigl\{\mathbb{I}(u_i = k) \,\big|\, \boldsymbol{y}\bigr\} \tag{20}$$

$$= \int \mathsf{P}(u_i = k \,|\, \boldsymbol{\psi},\, \boldsymbol{\theta},\, \boldsymbol{b}_i,\, \boldsymbol{y}_i)\, p(\boldsymbol{\psi},\, \boldsymbol{\theta}\, \boldsymbol{b}_i \,|\, \boldsymbol{y})\, d(\boldsymbol{\psi},\, \boldsymbol{\theta},\, \boldsymbol{b}_i), \tag{21}$$

$i = 1, \ldots, N$, $k = 1, \ldots, K$. Using the MCMC sample $\mathcal{S}_M$ from the joint posterior distribution $p(\boldsymbol{\psi},\, \boldsymbol{\theta},\, \boldsymbol{B},\, \boldsymbol{u} \,|\, \boldsymbol{y})$, expression (5), the quantities (20) are estimated as $M^{-1}\sum_{m=1}^{M}\mathbb{I}\bigl(u_i^{(m)} = k\bigr)$ (`poster.comp.prob1`). Similarly, quantities (21) are estimated as $M^{-1}\sum_{m=1}^{M}\mathsf{P}\bigl(u_i = k \,\big|\, \boldsymbol{\psi}^{(m)},\, \boldsymbol{\theta}^{(m)},\, \boldsymbol{b}_i^{(m)},\, \boldsymbol{y}_i\bigr)$ (`poster.comp.prob2`). Component `quant.comp.prob2` then contains posterior quantiles of $\mathsf{P}\bigl(u_i = k \,\big|\, \boldsymbol{\psi},\, \boldsymbol{\theta},\, \boldsymbol{b}_i,\, \boldsymbol{y}_i\bigr)$, $i = 1, \ldots, N$, $k = 1, \ldots, K$. Finally, component `comp.prob2` contains the sampled values of $\mathsf{P}\bigl(u_i = k \,\big|\, \boldsymbol{\psi}^{(m)},\, \boldsymbol{\theta}^{(m)},\, \boldsymbol{b}_i^{(m)},\, \boldsymbol{y}_i\bigr)$, $m = 1, \ldots, M$, $i = 1, \ldots, N$, $k = 1, \ldots, K$.

10.2. Classification

As it is described in Komárek and Komárková (2011a, Sec. 2.5), classification which does not take any uncertainty into account can be based on posterior means or medians of component probabilities $p_{i,k}(\boldsymbol{\psi}, \boldsymbol{\theta})$ when the ith patient is classified into group $g(i)$ for which $\widehat{\pi}_{i,g(i)} = \max_{k=1,\dots,K} \widehat{\pi}_{i,k}$ or $\widetilde{\pi}_{i,g(i)} = \max_{k=1,\dots,K} \widetilde{\pi}_{i,k}$. We perform and compare classification based on the maximal value of the posterior means and medians of component probabilities $p_{i,k}(\boldsymbol{\psi}, \boldsymbol{\theta})$.

```
R> groupMean <- apply(mod[[1]]$poster.comp.prob3, 1, which.max)
R> pMean <- apply(mod[[1]]$poster.comp.prob3, 1, max)
R> groupMed  <- apply(mod[[1]]$quant.comp.prob3[["50%"]], 1, which.max)
R> pMed <- apply(mod[[1]]$quant.comp.prob3[["50%"]], 1, max)
R> classif <- data.frame(id = IDS, groupMean = groupMean, pMean = pMean,
+                                   groupMed = groupMed,   pMed = pMed)
R> print(classif[1:10,])
```

```
   id groupMean     pMean groupMed      pMed
1   2         1 0.7170372        1 0.8266722
2   3         1 0.5803136        1 0.6231642
3   4         1 0.5305126        1 0.5674317
4   5         2 0.8674149        2 0.9571646
5   6         1 0.9883736        1 0.9902083
6   7         1 0.9793557        1 0.9837618
7   8         1 0.9905819        1 0.9950506
8   9         2 0.9997391        2 0.9999553
9  11         1 0.8065861        1 0.8320792
10 13         1 0.9546147        1 0.9651490
```

As might be expected, classification based on posterior means and medians differ only in situation of two patients where all posterior characteristics of component probabilities are close to 0.5 for both groups.

```
R> classif[groupMean != groupMed, ]
```

```
[1] id        groupMean pMean     groupMed  pMed
<0 rows> (or 0-length row.names)
```

Proportions of patients classified in this way correspond indeed to posterior means of mixture weights reported in Section 9.1.

```
R> table(groupMean)
```

```
groupMean
  1   2
167  93
```

```
R> round(prop.table(table(groupMean)) * 100, 2)
```

```
groupMean
    1     2 
64.23 35.77 
```

```
R> table(groupMed)
```

```
groupMed
  1   2 
167  93 
```

```
R> round(prop.table(table(groupMed)) * 100, 2)
```

```
groupMed
    1     2 
64.23 35.77 
```

Uncertainty in classification might be taken into account by incorporation of the credible interval in the classification rule. For example, we ultimately classify the i th patient into group $g(i)$ only if the lower limit of the 95% HPD interval for $p_{i,g(i)}(\boldsymbol{\psi}, \boldsymbol{\theta})$ exceeds a certain threshold, let say 0.9. To perform this classification, we utilize previously calculated HPD intervals stored in the object `prob3HPD`. First, we create a $N \times K$ matrix `prob3HPDlower` containing the lower limits of the 95% HPD intervals for $p_{i,k}(\boldsymbol{\psi}, \boldsymbol{\theta})$, $i = 1, \ldots, N$, $k = 1, \ldots, K$.

```
R> prob3HPDlower <- matrix(prob3HPD[, "lower"], ncol = 2, byrow = TRUE)
R> print(prob3HPDlower[1:5,])
```

```
             [,1]         [,2]
[1,] 1.116547e-01 0.0002921897
[2,] 8.053663e-02 0.0115940575
[3,] 1.871113e-02 0.0552320442
[4,] 5.445502e-08 0.3987641045
[5,] 9.722607e-01 0.0006975426
```

Subsequently, we perform classification where as group 3 we denote unclassified patients for whom none of the lower limits of the 95% HPD intervals exceeds the threshold of 0.9.

```
R> groupHPD <- apply(prob3HPDlower, 1, which.max)
R> pHPD <- apply(prob3HPDlower, 1, max)
R> groupHPD[pHPD < 0.9] <- 3
R> classif$groupHPD <- groupHPD
R> classif$pHPD <- pHPD
R> print(classif[1:10,])
```

```
   id groupMean     pMean groupMed      pMed groupHPD       pHPD
1   2         1 0.7170372        1 0.8266722        3 0.11165474
2   3         1 0.5803136        1 0.6231642        3 0.08053663
3   4         1 0.5305126        1 0.5674317        3 0.05523204
4   5         2 0.8674149        2 0.9571646        3 0.39876410
5   6         1 0.9883736        1 0.9902083        1 0.97226073
6   7         1 0.9793557        1 0.9837618        1 0.94798459
7   8         1 0.9905819        1 0.9950506        1 0.96622386
8   9         2 0.9997391        2 0.9999553        2 0.99889095
9  11         1 0.8065861        1 0.8320792        3 0.57972232
10 13         1 0.9546147        1 0.9651490        3 0.87942996
```

We obtain the following proportions of classified and unclassified patients.

```
R> table(groupHPD)
```

```
groupHPD
  1   2   3
 78  55 127
```

```
R> round(prop.table(table(groupHPD)) * 100, 2)
```

```
groupHPD
    1     2     3
30.00 21.15 48.85
```

10.3. Classification and values of individual random effects

On this place, we underline the fact that the classification (clustering) of individual patients is based on the mixture model for the latent values of individual random effects. Hence obtained classification should be reflected also on individual values of random effects or their estimates provided by their posterior means. Figure 11 shows estimated pairwise joint bivariate marginal posterior predictive densities of random effects supplemented by posterior means of individual random effects (saved in the object `bhat`, see the **R** code on page 33) where colors distinguish classification of individual patients according to the rule based on the lower limits of the HPD credible intervals for component probabilities.

```
R> COL <- c("darkgreen", "red4", "lightblue")
R> #
R> layout(matrix(c(1:9, 0, 10, 0), ncol = 3, byrow = TRUE))
R> for (i in 1:(mod[[1]]$dimb - 1)){
+    for (j in (i+1):mod[[1]]$dimb){
+      image(pdj[[1]]$x[[i]], pdj[[1]]$x[[j]], pdj[[1]]$dens[[paste(i, "-", j, sep = "")]]
+         col = rev(heat_hcl(33, c = c(80, 30), l = c(30, 90), power = c(1/5, 1.3))),
```

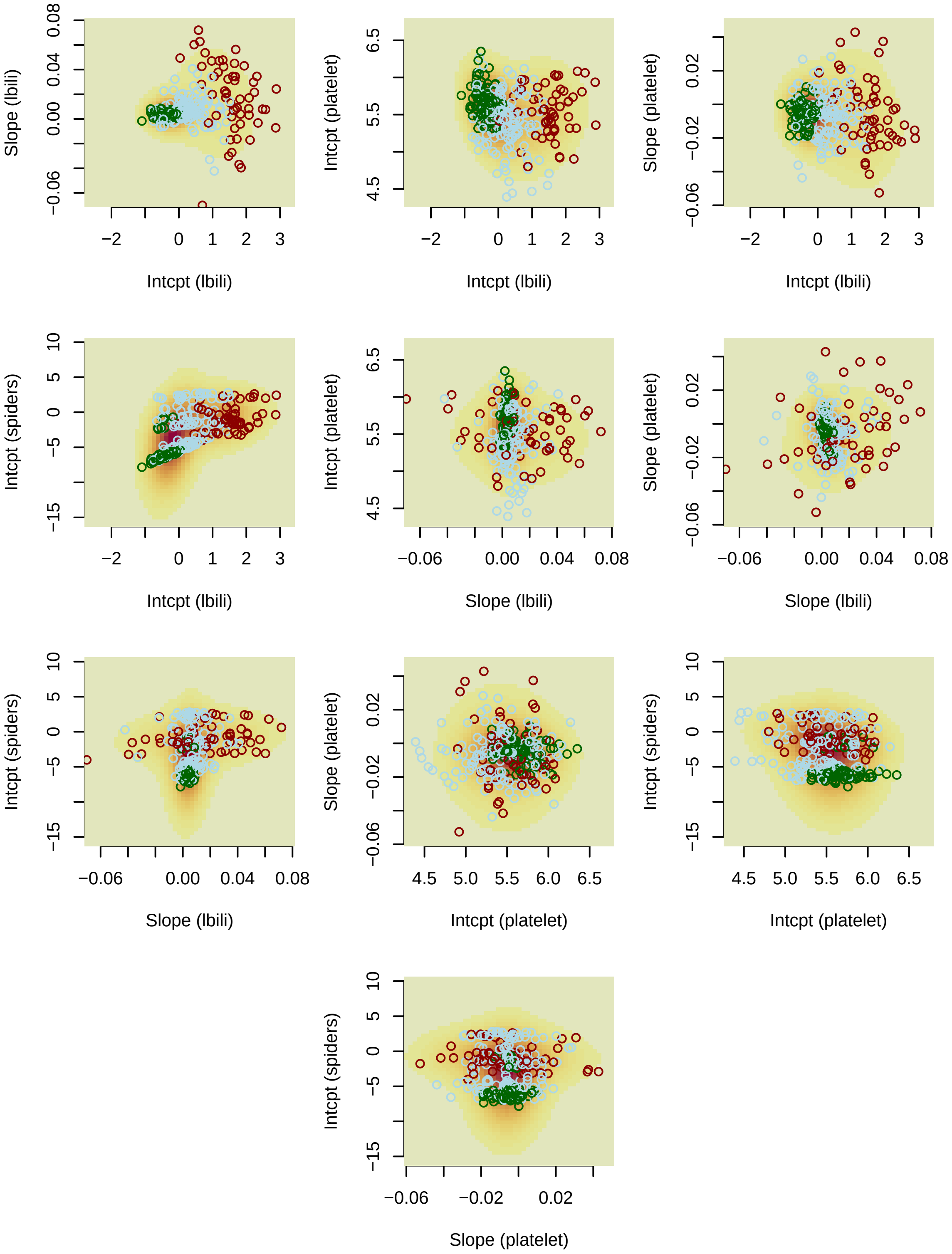

Figure 11: Estimated pairwise joint bivariate marginal posterior predictive densities of random effects supplemented by posterior means of individual random effects where colors distinguish classification of individual patients according to the rule based on the lower limits of the HPD credible intervals for component probabilities (group 1: green, group 2: red, unclassified: lightblue).

```
+         xlab = blab[i], ylab = blab[j])
+      points(bhat[, i], bhat[, j], pch = 1, col = COL[groupHPD])
+    }
+  }
```

10.4. Classification and observed longitudinal profiles

In Section 7.5, we used posterior means $\widehat{\boldsymbol{\alpha}} = \widehat{\alpha}_3$ of fixed effects and posterior means $\widehat{\boldsymbol{\beta}}$ of the overall means $\boldsymbol{\beta}$ of random effects to calculate estimated longitudinal profiles of analyzed markers. As we pointed out in Section 10.1, the groups into which we classify patients correspond to components of the normal mixture in the distribution of random effects which are, upon successful relabelling, characterized by posterior means of mixture parameters (see Section 9.1). In particular, let $\widehat{\boldsymbol{\mu}}_k$, $k = 1, \dots, K$ denote posterior means of mixture means and

$$\widehat{\boldsymbol{\mu}}_k^* = \bigl(\widehat{\mu}_{k,1,1}^*, \, \widehat{\mu}_{k,1,2}^*, \, \widehat{\mu}_{k,2,1}^*, \, \widehat{\mu}_{k,2,2}^*, \, \widehat{\mu}_{k,3}^*\bigr)^\top = \boldsymbol{s} + \boldsymbol{S}\widehat{\boldsymbol{\mu}}_k, \qquad k = 1, \dots, K,$$

their shifted and scaled counterparts (reported also by the `NMixSummComp` routine). Subsequently, we may calculate group-specific estimated longitudinal profiles of each marker which for a grid $t_{pred,j}$, $j = 1, \dots, J$ of time values would be calculated as

1. $\widehat{\mu}_{k,1,1}^* + \widehat{\mu}_{k,1,2}^* t_{pred,j}$, $j = 1, \dots, J$, $k = 1, \dots, K$ (estimated group-specific longitudinal profiles for logarithmic bilirubin);
2. $\exp\bigl(\widehat{\mu}_{k,2,1}^* + \widehat{\mu}_{k,2,2}^* t_{pred,j}\bigr)$, $j = 1, \dots, J$, $k = 1, \dots, K$ (estimated group-specific longitudinal profile for platelet count);
3. $\mathrm{logit}^{-1}\bigl(\widehat{\mu}_3^* + \widehat{\alpha}_3 t_{pred,j}\bigr)$, $j = 1, \dots, J$, $k = 1, \dots, K$ (estimated group-specific longitudinal profile for presence of blood vessel malformations).

Calculation of these group-specific estimated longitudinal profiles is provided by the `fitted` routine introduced in Section 7.5 if we set the value of its `overall` argument to `FALSE` (default value).

```
R> tpred <- seq(0, 30, by = 0.3)
R> fitGroup <- fitted(mod[[1]], x = list("empty", "empty", tpred),
+                                z = list(tpred, tpred, "empty"),
+                                overall = FALSE)
```

Object `fitGroup` is again a `list` with three components (one for each longitudinal marker). Each `list` component is a 101×2 matrix with calculated group-specific estimated longitudinal profiles in columns. For illustration, we print the first 10 values from each group for the second marker – platelet counts.

```
R> print(fitGroup[[2]][1:10,])
```

```
         [,1]     [,2]
[1,] 264.1380 234.7054
[2,] 263.6889 234.1231
```

```
 [3,] 263.2406 233.5422
 [4,] 262.7930 232.9627
 [5,] 262.3463 232.3847
 [6,] 261.9003 231.8081
 [7,] 261.4550 231.2330
 [8,] 261.0105 230.6593
 [9,] 260.5667 230.0870
[10,] 260.1237 229.5161
```

In a sequel, we plot observed longitudinal profiles overlaid by group-specific estimated lon-

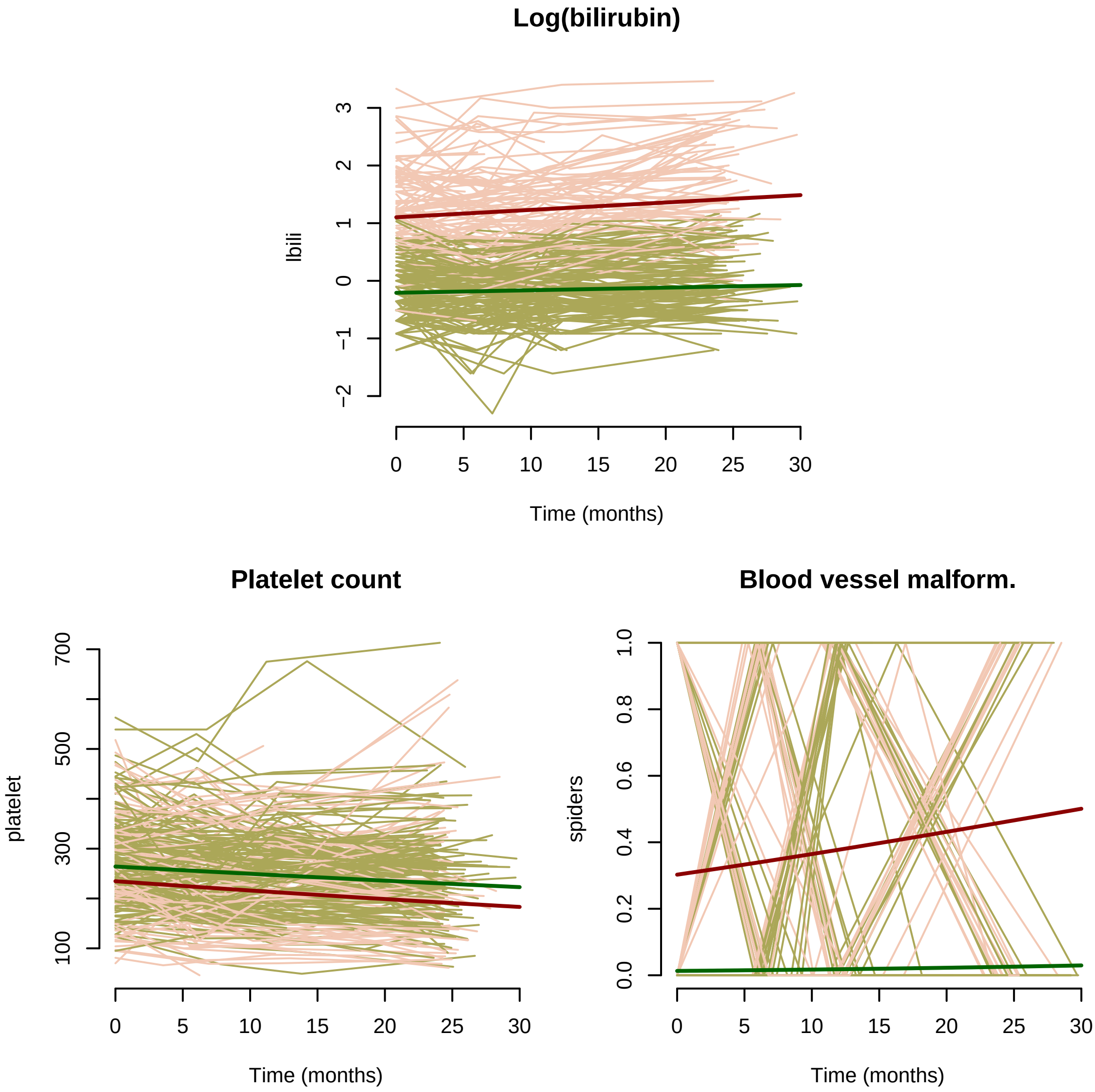

Figure 12: Observed longitudinal profiles of considered markers together with group-specific estimated longitudinal profiles (group 1: green, group 2: red).

gitudinal profiles. At the same time, we use different colors for observed profiles according to classification based on posterior medians of component probabilities (saved in the variable `groupMed`). First, we add classification information (variables `groupMean`, `groupMed`, `groupHPD`) as new `factor` variables to the original `data.frame` `pbc01`. Note that all three classification variables must be appropriately recycled.

```
R> TAB <- table(pbc01$id)
R> pbc01$groupMean <- factor(rep(groupMean, TAB))
R> pbc01$groupMed  <- factor(rep(groupMed, TAB))
R> pbc01$groupHPD  <- factor(rep(groupHPD, TAB))
```

Second, in the same was as on page 4, we extract individual profiles of considered markers and store them in the object `ip`. Nevertheless, besides the marker valus, we keep also the information concerning the classification in the object `ip`.

```
R> ip <- getProfiles(t = "month",
+      y = c("lbili", "platelet", "spiders", "groupMean", "groupMed", "groupHPD"),
+      id = "id", data = pbc01)
R> print(ip[[1]])


      month       lbili platelet spiders groupMean groupMed groupHPD
1  0.000000  0.09531018      221       1         1        1        3
2  5.979466 -0.22314355      188       1         1        1        3
3 11.991786  0.00000000      161       1         1        1        3
4 25.232033  0.64185389      122       1         1        1        3
```

Finally, we use the function `plotProfiles` and by providing two-component color vector we use the group-specific color for each observed longitudinal profile where the group is determined by variable specified in the `gvar` argument (which should refer to a `factor` variable). Plot is then supplemented in a standard way by previsously calculated group-specific estimated longitudinal profiles, see Figure 12.

```
R> #COL <- rainbow_hcl(2, start = 85, end = 40)
R> COL <- terrain_hcl(12, c = c(65, 0), l = c(45, 90), power = c(0.5, 1.5))[c(5, 9)]
R> names(COL) <- levels(pbc01$groupMed)
R> fitCOL <- c("darkgreen", "red4")
R> XLIM <- c(0, 910) / (365.25 / 12)
R> layout(autolayout(3))
R> #
R> plotProfiles(ip = ip, data = pbc01, var = "lbili",
+      gvar = "groupMed", tvar = "month",
+      xlim = XLIM, xlab = "Time (months)", col = COL,
+      auto.layout = FALSE, main = "Log(bilirubin)")
R> lines(tpred, fitGroup[[1]][,1], col = fitCOL[1], lwd = 2)
R> lines(tpred, fitGroup[[1]][,2], col = fitCOL[2], lwd = 2)
R> #
R> plotProfiles(ip = ip, data = pbc01, var = "platelet",
+     gvar = "groupMed", tvar = "month",
+     xlim = XLIM, xlab = "Time (months)", col = COL,
+     auto.layout = FALSE, main = "Platelet count")
R> lines(tpred, fitGroup[[2]][,1], col = fitCOL[1], lwd = 2)
R> lines(tpred, fitGroup[[2]][,2], col = fitCOL[2], lwd = 2)
R> #
R> plotProfiles(ip = ip, data = pbc01, var = "spiders",
+     gvar = "groupMed", tvar = "month",
+     xlim = XLIM, xlab = "Time (months)", col = COL,
+     auto.layout = FALSE, main = "Blood vessel malform.")
R> lines(tpred, fitGroup[[3]][,1], col = fitCOL[1], lwd = 2)
R> lines(tpred, fitGroup[[3]][,2], col = fitCOL[2], lwd = 2)
```

11. Selection of a number of mixture components

As it is explained in Komárek and Komárková (2011a, Sec. 2.6), selection of a number of mixture components and hence of an optimal number of clusters might be based on the posterior distribution of observed data deviances $D(\boldsymbol{\psi}, \boldsymbol{\theta}) = -2\log\bigl\{L(\boldsymbol{\psi}, \boldsymbol{\theta})\bigr\}$ where $L(\boldsymbol{\psi}, \boldsymbol{\theta})$ is given by (8), using the methodology proposed by Aitkin, Liu, and Chadwick (2009) and further elaborated in Chapters 7 and 8 of Aitkin (2010).

Alternatively, penalized expected deviance (Plummer 2008, PED) might be used. [ADD MORE EXPLANATION.]

As we showed in Section 5.7, posterior sample of observed data deviances is automatically saved as component `Deviance` of the object which results from the MCMC simulation invoked by the call to `GLMM_MCMC` function. In this Section, we use this sample to compare posterior distributions of observed data deviances of models with different numbers ($K = 1, 2, 3, 4$) of mixture components. For some of the tasks in this Section, function

- `summaryDiff()` which calculates posterior summary statistics for the difference between the two quantities,

from package **mixAK** will be used. In a sequel, let $D_K(\boldsymbol{\psi}, \boldsymbol{\theta})$ denote the observed data deviance in a model with K mixture components. Further, let $P_{K_2,K_1}(\boldsymbol{y}) = \mathsf{P}\bigl\{D_{K_2}(\boldsymbol{\psi}, \boldsymbol{\theta}) - D_{K_1}(\boldsymbol{\psi}, \boldsymbol{\theta}) < 0 \,\big|\, \boldsymbol{y}\bigr\}$ be the posterior probability that model with K_2 mixture components is better than model with K_1 mixture components.

First, we run the MCMC simulation of the same length and with the same prior distributions as in Section 5.1 for models with $K = 1, 2, 3, 4$ mixture components in the distribution of random effects. We separately store posterior samples of observed data deviances (from chain 1) in a `list devs` and calculated values of the penalized expected deviances in a `data.frame PED`.

```
R> devs <- mods <- list()
R> for (K in 1:4){
+    cat("Calculating K = ", K, "\n======================\n", sep="")
+
+    if (K == 2){
+      mods[[K]] <- mod
+    }else{
+      set.seed(20042007)
+      mods[[K]] <- GLMM_MCMC(y = pbc01[, c("lbili", "platelet", "spiders")],
+         dist = c("gaussian", "poisson(log)", "binomial(logit)"),
+         id = pbc01[, "id"],
+         x = list(lbili    = "empty",
+                 platelet = "empty",
+                 spiders  = pbc01[, "month"]),
+         z = list(lbili    = pbc01[, "month"],
+                  platelet = pbc01[, "month"],
+                  spiders  = "empty"),
+         random.intercept = rep(TRUE, 3),
+         prior.b = list(Kmax = K),
+         nMCMC = c(burn = 1000, keep = 10000, thin = 100, info = 1000),
+         PED = TRUE, parallel = FALSE)
+    }
+
+    devs[[K]] <- mods[[K]]$Deviance1    ### deviance from the first chain
+
```

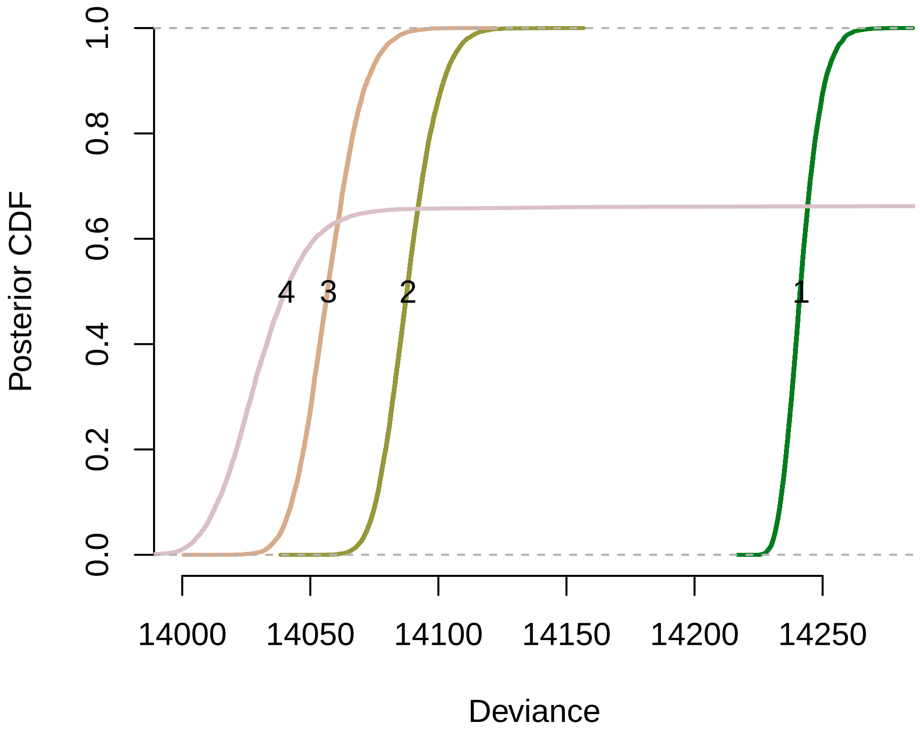

Figure 13: Posterior cumulative distribution functions of observed data deviances for models with $K = 1, 2, 3, 4$.

```
+    if (K == 1){
+      PED <- as.data.frame(matrix(mods[[K]]$PED, nrow=1))
+      colnames(PED) <- names(mods[[K]]$PED)
+    }else PED <- rbind(PED, mods[[K]]$PED)
+  }
```

Calculated penalized expected deviances for the four models (column `PED`):

```
R> print(PED)
```

```
  D.expect      p(opt)       PED    wp(opt)      wPED
1 14241.80   36.06845 14277.87   36.35170 14278.16
2 14088.32   75.80047 14164.12   75.99216 14164.32
3 14057.08  126.01605 14183.09  126.53383 14183.61
4 17244.38 5160.76117 22405.14 2926.41360 20170.79
```

We calculate posterior summary statistics for the difference between the observed data deviance from models with $K = 2$ and $K = 1$.

```
R> summaryDiff(devs[[2]], devs[[1]])
```

```
$summary
     Mean      2.5%       50%     97.5% 
-153.6042 -177.2704 -153.9062 -128.8956 

$Pcut
P(diff < -4.39)     P(diff < 0) 
              1               1 
```

For example, we conclude from the output that the posterior median for $D_2(\boldsymbol{\psi},\, \boldsymbol{\theta}) - D_1(\boldsymbol{\psi},\, \boldsymbol{\theta})$ is -153.91 and posterior probability $P_{2,1}(\boldsymbol{y})$ that model with $K = 2$ is better than model with $K = 1$ is practically equal to 1. The output further reports posterior probability $\mathsf{P}\bigl\{D_2(\boldsymbol{\psi},\, \boldsymbol{\theta}) - D_1(\boldsymbol{\psi},\, \boldsymbol{\theta}) < -2\log(9) \doteq -4.39 \,\big|\, \boldsymbol{y}\bigr\}$, where the value of $-2\log(9)$ is chosen according to the guidelines given in Aitkin *et al.* (2009, Sec. 4). They argue that if this posterior probability is high (0.9 or more like in our case), we have quite strong evidence in favor of model with $K = 2$ over model with $K = 1$. Similarly, we compare model with $K = 3$ to model with $K = 2$ and find out that model a three-component model might be favorable to a two-component model with $P_{3,2}(\boldsymbol{y}) = 0.9848$ and $\mathsf{P}\bigl\{D_3(\boldsymbol{\psi},\, \boldsymbol{\theta}) - D_2(\boldsymbol{\psi},\, \boldsymbol{\theta}) < -2\log(9) \,\big|\, \boldsymbol{y}\bigr\} = 0.9700$.

```
R> summaryDiff(devs[[3]], devs[[2]])
```

```
$summary
     Mean      2.5%       50%     97.5% 
-31.08387 -58.22722 -31.27067  -3.22487 

$Pcut
P(diff < -4.39)     P(diff < 0) 
         0.9700          0.9848 
```

For completeness, we compare also a four-component model to both three- and two-component models and find out that the four-component model is by no means favorable neither to a two-component nor to a three-component model.

```
R> summaryDiff(devs[[4]], devs[[3]])


$summary
      Mean       2.5%        50%      97.5%
1654.37019  -57.04976  -16.08968 5335.57157

$Pcut
P(diff < -4.39)     P(diff < 0)
         0.5976          0.6192


R> summaryDiff(devs[[4]], devs[[2]])


$summary
      Mean       2.5%        50%      97.5%
1623.28632  -88.22094  -46.66565 5304.30897

$Pcut
P(diff < -4.39)     P(diff < 0)
         0.6552          0.6559
```

Overall comparison of the posterior distributions of deviances under competing models is provided by comparison of posterior cumulative distribution functions (cdf's) of deviances which are plotted on Figure 13.

```
R> COL <- terrain_hcl(4, c = c(65, 15), l = c(45, 80), power = c(0.5, 1.5))
R> plot(c(14000, 14275), c(0, 1), type="n",
+       xlab="Deviance", ylab="Posterior CDF")
R> for (K in 1:4){
+    medDEV <- median(devs[[K]])
+    ECDF <- ecdf(devs[[K]])
+    plot(ECDF, col=COL[K], lwd=2, add=TRUE)
+    text(medDEV+0.5, 0.5, labels=K)
+  }
```

Acknowledgments

The work on this paper has been supported by the grant GAČR 201/09/P077, Czech Science Foundation and the grant MSM 0021620839, Ministry of Education, Youth and Sports of the Czech Republic.

This document was prepared using **Sweave** (Leisch 2002).

References

Aitkin M (2010). *Statistical Inference: An Integrated Bayesian/Likelihood Approach.* Chapman & Hall/CRC, Boca Raton. ISBN 978-1-4200-9343-8.

Aitkin M, Liu CC, Chadwick T (2009). "Bayesian model comparison and model averaging for small-area estimation." *The Annals of Applied Statistics*, **3**(1), 199–221. `doi:10.1214/08-AOAS205`.

Bates D, Maechler M, Bolker B (2011). *lme4: Linear mixed-effects models using S4 classes.* R package version 0.999375-42, URL `http://CRAN.R-project.org/package=lme4`.

Cleveland WS (1979). "Robust locally weighted regression and smoothing scatterplots." *Journal of the American Statistical Association*, **74**(368), 829–836. `doi:10.2307/2286407`.

Dickson ER, Grambsch PM, Fleming TR, Fisher LD, Langworthy A (1989). "Prognosis in primary biliary-cirrhosis – Model for decision-making." *Hepatology*, **10**(1), 1–7. `doi:10.1002/hep.1840100102`.

Gelfand AE, Sahu SK, Carlin BP (1995). "Efficient parametrisations for normal linear mixed models." *Biometrika*, **82**(3), 479–499. `doi:10.1093/biomet/82.3.479`.

Genz A, Azzalini A (2011). *mnormt: The multivariate normal and t distributions.* R package version 1.4-3, URL `http://CRAN.R-project.org/package=mnormt`.

Ihaka R, Murrell P, Hornik K, Zeileis A (2009). *colorspace: Color space manipulation.* R package version 1.1-0, URL `http://CRAN.R-project.org/package=colorspace`.

Knaus J (2010). *snowfall: Easier cluster computing (based on snow).* R package version 1.84, URL `http://CRAN.R-project.org/package=snowfall`.

Komárek A (2009). "A new R package for Bayesian estimation of multivariate normal mixtures allowing for selection of the number of components and interval-censored data." *Computational Statistics and Data Analysis*, **53**(12), 3932–3947. `doi:10.1016/j.csda.2009.05.006`.

Komárek A, Komárková L (2011a). "Clustering for multivariate continuous and discrete longitudinal data." *Submitted for publication.*

Komárek A, Komárková L (2011b). "Supplement to "Clustering for Multivariate Continuous and Discrete Longitudinal Data"."

Leisch F (2002). "Dynamic Generation of Statistical Reports Using Literate Data Analysis." In W Härdle, B Rönz (eds.), *COMPSTAT 2002 – Proceedings in Computational Statistics*, pp. 575–580. Physica-Verlag, Heidelberg.

Müller HG (2005). "Functional modelling and classification of longitudinal data." *Scandinavian Journal of Statistics*, **32**(2), 223–240. `doi:10.1111/j.1467-9469.2005.00429.x`.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523–539. doi:10.1093/biostatistics/kxm049.

Plummer M, Best N, Cowles K, Vines K (2006). "CODA: Convergence diagnosis and output analysis for MCMC." *R News*, **6**(1), 7–11. URL http://CRAN.R-project.org/doc/Rnews/.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Richardson S, Green PJ (1997). "On Bayesian analysis of mixtures with unknown number of components (with Discussion)." *Journal of the Royal Statistical Society, Series B*, **59**(4), 731–792. doi:10.1111/1467-9868.00095.

Stephens M (2000). "Dealing with label switching in mixture models." *Journal of the Royal Statistical Society, Series B*, **62**(4), 795–809. doi:10.1111/1467-9868.00265.

Tierney L, Rossini AJ, Li N, Ševčíková H (2011). *snow: Simple network of workstations*. R package version 0.3-7, URL http://CRAN.R-project.org/package=snow.

Zeileis A, Hornik K, Murrell P (2009). "Escaping RGBland: Selecting colors for statistical graphics." *Computational Statistics and Data Analysis*, **53**(9), 3259–3270. doi:10.1016/j.csda.2008.11.033.

Affiliation:

Arnošt Komárek
Dept. of Probability and Mathematical Statistics
Faculty of Mathematics and Physics, Charles University in Prague
Sokolovská 83
186 75 Praha 8 – Karlín, Czech Republic
E-mail: Arnost.Komarek@mff.cuni.cz
URL: http://www.karlin.mff.cuni.cz/~komarek/