June 26, 2008

Package glmmAK: Example Toenail

ARNOŠT KOMÁREK

Department of Probability and Mathematical Statistics, Charles University in Prague
Sokolovská 83, CZ–186 75, Praha 8, Czech Republic

E-mail: arnost.komarek@mff.cuni.cz

This document shows how to perform the analysis of the Toenail data presented in Komárek and Lesaffre (2008) using the functions of the package glmmAK. To process the MCMC output, we also extensively use the coda package (Plummer et al., 2006). It is assumed that the user reads Komárek and Lesaffre's paper first. In this manual, the same notation is used, often without redefining it.

This manual especially supplements the help pages of the following functions of the package glmmAK:

- cumlogitRE,
- summaryGspline1.

The user is encouraged to take a look on the manual pages of these functions first! You can try

```
> help(cumlogitRE, package = glmmAK, htmlhelp = TRUE)
> help(summaryGspline1, package = glmmAK, htmlhelp = TRUE)
```

1 Getting started

We start by loading the package, specifying the working directory and loading the data:

```
> library(glmmAK)
> root <- "/home/komarek/Rlib/glmmAK/Doc/"
> setwd(root)
> data(toenail)
```

Brief summary of the data:

```
> summary(toenail)
```

```
     idnr           infect            trt              time            visit      
 Min.   :  1.0   Min.   :0.0000   Min.   :0.0000   Min.   : 0.000   Min.   :1.000  
 1st Qu.:101.8   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 1.000   1st Qu.:2.000  
 Median :192.0   Median :0.0000   Median :1.0000   Median : 3.000   Median :4.000  
 Mean   :189.8   Mean   :0.2138   Mean   :0.5089   Mean   : 4.691   Mean   :3.896  
 3rd Qu.:276.2   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.: 8.893   3rd Qu.:6.000  
 Max.   :383.0   Max.   :1.0000   Max.   :1.0000   Max.   :18.500   Max.   :7.000  
```

2 Data and models

A longitudinal clinical trial in dermatology was set up to compare the efficacy of two oral treatments for toenail infection (De Backer et al., 1998). In this manual, we will analyze a dichotomized version of the degree of onycholysis which expresses the degree of separation of the nail plate from the nail-bed (0 = absent or mild; 1 = moderate or severe). The response was evaluated at seven visits (approximately on weeks 0, 4, 8, 12, 24, 36 and 48) and in total 937 and 971 measurements on 146 and 148 patients ($N = 294$), respectively are available in the control group (itraconazole 200 mg/day) and in the treatment group (terbinafine 250 mg/day), respectively. Let $Y_{i,l}$ represent the dichotomized onycholysis of the i-th subject at the l-th visit. Further, let Trt_i denote the binary treatment indicator and $\mathsf{Time}_{i,l}$ the visit time in months. We will consider two PGM GLMM and two Normal GLMM's.

2.1 PGM GLMM, not hierarchically centered

PGM GLMM, not hierarchically centered model is the following:

$$\mathrm{logit}\bigl\{\mathrm{P}(Y_{i,l} = 1 \,|\, \boldsymbol{\beta},\, b_i)\bigr\} = \beta_1 + \beta_2\,\mathsf{Trt}_i + \beta_3\,\mathsf{Time}_{i,l} + \beta_4\,\mathsf{Time}_{i,l}\cdot\mathsf{Trt}_i + b_i, \qquad (1)$$

where

$$b_i \overset{\text{i.i.d.}}{\sim} \sum_{j=-K}^{K} w_j(\boldsymbol{a})\mathcal{N}\bigl(\tau\mu_j,\,(\tau\sigma)^2\bigr) \qquad (i = 1,\dots,N).$$

In a sequel, we will denote this model as **PGM GLMM(nhc)**.

The results of this model are shown in Komárek and Lesaffre (2008).

2.2 PGM GLMM, hierarchically centered

PGM GLMM, hierarchically centered model is the following:

$$\mathrm{logit}\bigl\{\mathrm{P}(Y_{i,l} = 1 \,|\, \boldsymbol{\beta},\, b_i)\bigr\} = \beta_2\,\mathsf{Trt}_i + \beta_3\,\mathsf{Time}_{i,l} + \beta_4\,\mathsf{Time}_{i,l}\cdot\mathsf{Trt}_i + b_i, \qquad (2)$$

where

$$b_i \overset{\text{i.i.d.}}{\sim} \alpha + \sum_{j=-K}^{K} w_j(\boldsymbol{a})\mathcal{N}\bigl(\tau\mu_j,\,(\tau\sigma)^2\bigr) \qquad (i = 1,\dots,N).$$

In a sequel, we will denote this model as **PGM GLMM(hc)**.

2.3 Normal GLMM, not hierarchically centered

Normal GLMM, not hierarchically centered model is the following:

$$\mathrm{logit}\bigl\{\mathrm{P}(Y_{i,l} = 1 \,|\, \boldsymbol{\beta},\, b_i)\bigr\} = \beta_1 + \beta_2\,\mathsf{Trt}_i + \beta_3\,\mathsf{Time}_{i,l} + \beta_4\,\mathsf{Time}_{i,l}\cdot\mathsf{Trt}_i + b_i, \qquad (3)$$

where

$$b_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,\,d) \qquad (i = 1,\dots,N).$$

In a sequel, we will denote this model as **Normal GLMM(nhc)**.

The results of this model are shown in Komárek and Lesaffre (2008).

2.4 Normal GLMM, hierarchically centered

Normal GLMM, hierarchically centered model is the following:

$$\operatorname{logit}\bigl\{\mathrm{P}(Y_{i,l} = 1 \,|\, \boldsymbol{\beta},\, b_i)\bigr\} = \beta_2\,\mathsf{Trt}_i + \beta_3\,\mathsf{Time}_{i,l} + \beta_4\,\mathsf{Time}_{i,l}\cdot\mathsf{Trt}_i + b_i, \tag{4}$$

where

$$b_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\alpha,\, d) \qquad (i = 1, \ldots, N).$$

In a sequel, we will denote this model as **Normal GLMM(hc)**.

2.5 Remarks

From the probabilistic point of view, PGM GLMM(nhc) is indeed equivalent to PGM GLMM(hc) and Normal GLMM(nhc) is equivalent to Normal GLMM(hc).

3 Specification of the prior distributions

Choices for the prior distributions are passed as `list` objects to the function `cumlogitRE`. In this Section, we create objects holding the prior information for considered models.

3.1 Prior for the fixed effects $\boldsymbol{\beta}$

In all models, we will assume that the prior distribution for the components of the vector of fixed effects $\boldsymbol{\beta}$ is a product of independent normal distributions $\mathcal{N}(0,\,10\,000)$:

```
> prior.fixed <- list(mean = 0, var = 10000)
```

3.2 Prior for the parameters of the penalized Gaussian mixture in the PGM GLMM's

For the PGM GLMM's (1) and (2), the following choices of the parameters defining the PGM will be used: $K = 15$, that is, $2 \cdot 15 + 1 = 31$ knots. Further, the distance between the two consecutive knots will be $\delta = 0.3$, that is, the knots are

$$\boldsymbol{\mu} = \{\mu_{-15}, \ldots, \mu_{15}\} = \{j\delta : \ j = -15, \ldots, 15\} = \{-4.5,\ -4.2, \ldots, 4.2,\ 4.5\}.$$

The basis standard deviation σ will be equal to 0.2.

The prior distribution for the transformed PGM weights $\boldsymbol{a}$ will be the intrinsic Gaussian Markov random field (IGMRF) based on the 3rd order (CARorder=3) differences between the consecutive weights, i.e.,

$$p(\boldsymbol{a}\,|\,\lambda) \propto \exp\Bigl\{-\frac{\lambda}{2}\sum_{j=-K+3}^{K}\bigl(a_j - 3a_{j-1} + 3a_{j-2} - a_{j-3}\bigr)^2\Bigr\}.$$

For the smoothing hyperparameter λ a gamma prior Gamma(1, 0.005) will be used. The transformed weights $\boldsymbol{a}$ will be updated using the slice sampling of Neal (2003). All above information is stored in a `list`:

```
> prior.gspline.Slice <- list(K = 15, delta = 0.3, sigma = 0.2, CARorder = 3,
+     Ldistrib = "gamma", Lequal = FALSE, Lshape = 1, LinvScale = 0.005,
+     AtypeUpdate = "slice")
```

Alternatively, it is possible to update the transformed weights $\boldsymbol{a}$ jointly using a Metropolis-Hastings step based on a normal proposal constructed using a Taylor expansion of $\log\bigl\{p(\boldsymbol{a}\,|\,\cdots)\bigr\}$ around the mode located with one Newton-Raphson step starting from the current value of $\boldsymbol{a}$, see Rue and Held (2005) for more details. In that case, the component AtypeUpdate has to be modified:

```
> prior.gspline.Block <- list(K = 15, delta = 0.3, sigma = 0.2, CARorder = 3,
+     Ldistrib = "gamma", Lequal = FALSE, Lshape = 1, LinvScale = 0.005,
+     AtypeUpdate = "block")
```

3.3 Prior for the remaining parameters of the random effects distribution in the PGM GLMM's

In both PGM GLMM's (1) and (2) we still have to specify prior choices for the PGM scale parameter τ, in the PGM GLMM(hc) (2) we also have to specify the prior distribution for the PGM location α. We will use the following priors:

$$\tau^{-2} \sim \text{Gamma}(1,\, 0.005), \qquad \alpha \sim \mathcal{N}(0,\, 10\,000),$$

which in the case of the PGM GLMM(nhc) is in R specified as

```
> prior.random.gspl.nhc <- list(Ddistrib = "gamma", Dshape = 1, DinvScale = 0.005)
```

and in the case of the PGM GLMM(hc) as

```
> prior.random.gspl.hc <- list(Mdistrib = "normal", Mmean = 0, Mvar = 10000,
+     Ddistrib = "gamma", Dshape = 1, DinvScale = 0.005)
```

Alternatively, one can assume the uniform prior for the PGM scale parameter τ which is often prefered to the gamma prior, see Gelman (2006) for the discussion of this point. For example, the prior distribution

$$\tau \sim \text{Unif}(0,\, 100)$$

is specified in the following way:

```
> prior.random.gspl.nhc.Unif <- list(Ddistrib = "sduniform", Dupper = 100)
```

3.4 Prior for the parameters of the random effects distribution in the Normal GLMM's

In both Normal GLMM's (3) and (4) we have to specify prior distribution for the variance d of the random intercept and in the Normal GLMM(hc) (4) also the prior for the mean α of the random intercept. We will use the following priors:

$$d^{-1} \sim \text{Gamma}(1,\, 0.005), \qquad \alpha \sim \mathcal{N}(0,\, 10\,000),$$

which in the case of the **Normal GLMM(nhc)** is in R specified as

```
> prior.random.norm.nhc <- list(Ddistrib = "gamma", Dshape = 1, DinvScale = 0.005)
```

and in the case of the **Normal GLMM(hc)** as

```
> prior.random.norm.hc <- list(Mdistrib = "normal", Mmean = 0, Mvar = 10000,
+     Ddistrib = "gamma", Dshape = 1, DinvScale = 0.005)
```

Similarly like above, one can assume the uniform prior for the random intercept standard deviation $\sqrt{d}$. For example, the prior distribution

$$\sqrt{d} \sim \text{Unif}(0,\, 100)$$

is specified in the following way:

```
> prior.random.norm.nhc.Unif <- list(Ddistrib = "sduniform", Dupper = 100)
```

4 MCMC simulation

Having specified the prior distribution we are almost ready to start the MCMC simulation to sample from the posterior distribution of the model parameters.

4.1 Directories to store the chains

For each considered model, we create one directory as a subdirectory of root/chToenail which will afterwards be used to store the sampled chains. Creation of directories can of course be performed outside R as well.

```
> if (!("chToenail" %in% dir(root))) dir.create(paste(root, "chToenail",
+     sep = ""))
> dirNames <- c("PGM_nhc", "PGM_hc", "Normal_nhc", "Normal_hc")
> dirPaths <- paste(root, "chToenail/", dirNames, "/", sep = "")
> for (i in 1:length(dirPaths)) {
+     if (!(dirNames[i] %in% dir(paste(root, "chToenail", sep = ""))))
+         dir.create(dirPaths[i])
+ }
> names(dirPaths) <- c("PGM_nhc", "PGM_hc", "Normal_nhc", "Normal_hc")
```

That is, the chains for considered models will be stored in the following directories:

```
> print(dirPaths)
```

```
                                                 PGM_nhc
   "/home/komarek/Rlib/glmmAK/Doc/chToenail/PGM_nhc/"
                                                  PGM_hc
    "/home/komarek/Rlib/glmmAK/Doc/chToenail/PGM_hc/"
                                              Normal_nhc
"/home/komarek/Rlib/glmmAK/Doc/chToenail/Normal_nhc/"
                                               Normal_hc
 "/home/komarek/Rlib/glmmAK/Doc/chToenail/Normal_hc/"
```

4.2 Matrix of covariates

To pass the covariates to the function cumlogitRE, we have to create a matrix which will contain only the covariates involved in the fitted model. That is, from the original data, we have to take the treatment indicator, time values and create an interaction between treatment and time. We will store the covariates in the matrix iXmat:

```
> iXmat <- data.frame(trt = toenail$trt,
+                      time = toenail$time,
+                      trt.time = toenail$trt * toenail$time)
```

4.3 Length of the MCMC

The length of the MCMC simulation will be passed to the function `cumlogitRE` as a `list`:

```
> nsimul <- list(niter = 4000, nthin = 100, nburn = 2000, nwrite = 100)
```

With this specification, we will perform in total 4000 iterations out of which 2000 iterations will be a burn-in period. Further, we will thin the sample and store only every 100th value. Finally, the iteration count will increase every 100 iterations. That is, for inference, we will have chains of length 2000.

Remark: In the paper Komárek and Lesaffre (2008), longer MCMC simulation was used to derive the results presented there.

4.4 Running MCMC

At this stage, we have specified all the information to start the MCMC simulation by calling the function `cumlogitRE` for each considered model. Be aware that this can take some time, according to the length of the MCMC specified.

PGM GLMM(nhc)

```
> fit.PGM.nhc <- cumlogitRE(y = toenail$infect, x = iXmat,
+     cluster = toenail$idnr, intcpt.random = TRUE, hierar.center = FALSE,
+     C = 1, drandom = "gspline", prior.fixed = prior.fixed,
+     prior.random = prior.random.gspl.nhc, prior.gspline = prior.gspline.Slice,
+     nsimul = nsimul, store = list(prob = FALSE, b = TRUE),
+     dir = dirPaths["PGM_nhc"])
```

```
Simulation started on           Thu May 31 11:02:34 2007
Iteration 2000
Burn-up finished on             Thu May 31 11:20:09 2007   (iteration 2000)
Iteration 4000
Simulation finished on          Thu May 31 11:38:06 2007   (iteration 4000)
```

PGM GLMM(hc)

```
> fit.PGM.hc <- cumlogitRE(y = toenail$infect, x = iXmat,
+     cluster = toenail$idnr, intcpt.random = TRUE, hierar.center = TRUE,
+     C = 1, drandom = "gspline", prior.fixed = prior.fixed,
+     prior.random = prior.random.gspl.hc, prior.gspline = prior.gspline.Slice,
+     nsimul = nsimul, store = list(prob = FALSE, b = TRUE),
+     dir = dirPaths["PGM_hc"])
```

```
Simulation started on           Thu May 31 11:38:06 2007
Iteration 2000
Burn-up finished on             Thu May 31 11:55:33 2007   (iteration 2000)
Iteration 4000
Simulation finished on          Thu May 31 12:11:51 2007   (iteration 4000)
```

Normal GLMM(nhc)

```
> fit.Normal.nhc <- cumlogitRE(y = toenail$infect, x = iXmat,
+     cluster = toenail$idnr, intcpt.random = TRUE, hierar.center = FALSE,
+     C = 1, drandom = "normal", prior.fixed = prior.fixed,
+     prior.random = prior.random.norm.nhc,
+     nsimul = nsimul, store = list(prob = FALSE, b = TRUE),
+     dir = dirPaths["Normal_nhc"])
```

```
Simulation started on           Thu May 31 12:11:51 2007
Iteration 2000
Burn-up finished on             Thu May 31 12:26:44 2007   (iteration 2000)
Iteration 4000
Simulation finished on          Thu May 31 12:41:42 2007   (iteration 4000)
```

Normal GLMM(hc)

```
> fit.Normal.hc <- cumlogitRE(y = toenail$infect, x = iXmat,
+     cluster = toenail$idnr, intcpt.random = TRUE, hierar.center = TRUE,
+     C = 1, drandom = "normal", prior.fixed = prior.fixed,
+     prior.random = prior.random.norm.hc,
+     nsimul = nsimul, store = list(prob = FALSE, b = TRUE),
+     dir = dirPaths["Normal_hc"])
```

```
Simulation started on           Thu May 31 12:41:42 2007
Iteration 2000
Burn-up finished on             Thu May 31 12:55:55 2007   (iteration 2000)
Iteration 4000
Simulation finished on          Thu May 31 13:09:58 2007   (iteration 4000)
```

5 Basic posterior computation

In this Section, we compute posterior summary statistics for regression coefficients $\boldsymbol{\beta}$ and moments of the distribution of random effects.

5.1 Reading the chains into coda objects

Using the commands below, it is possible to read all sampled chains and store them as coda mcmc objects. It is possible to skip some values at the beginning of the chains by setting the argument skip to a positive value.

```
> chPGM.nhc <- glmmAK.files2coda(dir = dirPaths["PGM_nhc"], drandom = "gspline",
+     skip = 0)
> chPGM.hc <- glmmAK.files2coda(dir = dirPaths["PGM_hc"], drandom = "gspline",
+     skip = 0)
> chNormal.nhc <- glmmAK.files2coda(dir = dirPaths["Normal_nhc"],
+     drandom = "normal", skip = 0)
> chNormal.hc <- glmmAK.files2coda(dir = dirPaths["Normal_hc"], drandom = "normal",
+     skip = 0)
```

5.2 Reading only needed chains

On this place, we will read only the chains that will be worked out now, that is the chains for regression coefficients $\boldsymbol{\beta}$ and the chains for the moments of the random intercept distribution. We will use the function scanFH which is a customized version of the R base function scan. All chains will be stored as coda mcmc objects.

PGM GLMM(nhc)

The chains we need now will be stored in the object chPGM.nhc. Note that the chain stored in the file betaRadj.sim, column "`(Intercept)`", is the chain for

$$\gamma_1 = \beta_1 + \mathrm{E}(b) = \beta_1 + \tau\beta_1^*, \quad \text{where } \beta_1^* = \sum_{j=-K}^{K} w_j(\boldsymbol{a})\mu_j,$$

which is the mean intercept value. Similarly, the chain stored in the file varRadj.sim, column "`varR.1`" is the chain for

$$d = \mathrm{var}(b) = \tau^2\, d^*, \quad \text{where } d^* = \sum_{j=-K}^{K} w_j(\boldsymbol{a})(\mu_j - \beta_1^*)^2 + \sigma^2.$$

```
> iters     <- scanFH(paste(dirPaths["PGM_nhc"], "iteration.sim", sep = ""))
> betaF     <- scanFH(paste(dirPaths["PGM_nhc"], "betaF.sim", sep = ""))
> betaRadj  <- scanFH(paste(dirPaths["PGM_nhc"], "betaRadj.sim", sep = ""))
> varRadj   <- scanFH(paste(dirPaths["PGM_nhc"], "varRadj.sim", sep = ""))
> chPGM.nhc <- mcmc(data.frame(Trt = betaF[, "trt"],
+                              Time = betaF[, "time"],
+                              Trt.Time = betaF[, "trt.time"],
+                                Meanb = betaRadj[, "(Intercept)"],
+                                SDb = sqrt(varRadj[, "varR.1"])),
+                     start = iters[1, 1])
> rm(list = c("iters", "betaF", "betaRadj", "varRadj"))
```

PGM GLMM(hc)

The chains we need now will be stored in the object chPGM.hc. Note that the chain stored in the file betaRadj.sim, column "`(Intercept)`", is the chain for

$$\gamma_1 = \mathrm{E}(b) = \alpha + \tau\beta_1^*, \quad \text{where } \beta_1^* = \sum_{j=-K}^{K} w_j(\boldsymbol{a})\mu_j,$$

which is the mean intercept value. Similarly, the chain stored in the file varRadj.sim, column "`varR.1`" is the chain for

$$d = \mathrm{var}(b) = \tau^2\, d^*, \quad \text{where } d^* = \sum_{j=-K}^{K} w_j(\boldsymbol{a})(\mu_j - \beta_1^*)^2 + \sigma^2.$$

```
> iters    <- scanFH(paste(dirPaths["PGM_hc"], "iteration.sim", sep = ""))
> betaF    <- scanFH(paste(dirPaths["PGM_hc"], "betaF.sim", sep = ""))
> betaRadj <- scanFH(paste(dirPaths["PGM_hc"], "betaRadj.sim", sep = ""))
> varRadj  <- scanFH(paste(dirPaths["PGM_hc"], "varRadj.sim", sep = ""))
> chPGM.hc <- mcmc(data.frame(Trt = betaF[, "trt"],
+                             Time = betaF[, "time"],
+                             Trt.Time = betaF[, "trt.time"],
+                             Meanb = betaRadj[, "(Intercept)"],
+                             SDb = sqrt(varRadj[, "varR.1"])),
+                  start = iters[1, 1])
> rm(list = c("iters", "betaF", "betaRadj", "varRadj"))
```

Normal GLMM(nhc)

The chains we need now will be stored in the object chNormal.nhc.

```
> iters <- scanFH(paste(dirPaths["Normal_nhc"], "iteration.sim", sep = ""))
> betaF <- scanFH(paste(dirPaths["Normal_nhc"], "betaF.sim", sep = ""))
> varR  <- scanFH(paste(dirPaths["Normal_nhc"], "varR.sim", sep = ""))
> chNormal.nhc <- mcmc(data.frame(Trt = betaF[, "trt"],
+                                 Time = betaF[, "time"],
+                                 Trt.Time = betaF[, "trt.time"],
+                                 Meanb = betaF[, "(Intercept)"],
+                                 SDb = sqrt(varR[, "varR.1.1"])),
+                      start = iters[1, 1])
> rm(list = c("iters", "betaF", "varR"))
```

Normal GLMM(hc)

The chains we need now will be stored in the object chNormal.hc.

```
> iters <- scanFH(paste(dirPaths["Normal_hc"], "iteration.sim", sep = ""))
> betaF <- scanFH(paste(dirPaths["Normal_hc"], "betaF.sim", sep = ""))
> betaR <- scanFH(paste(dirPaths["Normal_hc"], "betaR.sim", sep=""))
> varR  <- scanFH(paste(dirPaths["Normal_hc"], "varR.sim", sep = ""))
> chNormal.hc <- mcmc(data.frame(Trt = betaF[, "trt"],
+                               Time = betaF[, "time"],
+                               Trt.Time = betaF[, "trt.time"],
+                               Meanb = betaR[, "(Intercept)"],
+                               SDb = sqrt(varR[, "varR.1.1"])),
+                    start = iters[1, 1])
> rm(list = c("iters", "betaF", "betaR", "varR"))
```

5.3 Basic posterior summary statistics

Basic posterior summary statistics can be obtained using the `coda` `summary` function for objects of class `mcmc`:

PGM GLMM(nhc)

```
> summary(chPGM.nhc)
```

```
Iterations = 2001:4000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 2000

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

            Mean      SD Naive SE Time-series SE
Trt       0.3774 0.44109 0.009863       0.022469
Time     -0.3885 0.04546 0.001017       0.001768
Trt.Time -0.1246 0.07072 0.001581       0.003021
Meanb    -1.6255 0.65222 0.014584       0.065039
SDb       3.5959 0.57588 0.012877       0.062029

2. Quantiles for each variable:

            2.5%      25%     50%      75%    97.5%
Trt      -0.5333  0.09603  0.3826  0.66917  1.21108
Time     -0.4799 -0.41982 -0.3868 -0.35607 -0.30455
Trt.Time -0.2664 -0.16980 -0.1249 -0.07544  0.01188
Meanb    -3.1081 -2.00434 -1.5222 -1.16930 -0.60400
SDb       2.7817  3.16391  3.4718  3.92456  4.96336
```

PGM GLMM(hc)

```
> summary(chPGM.hc)


Iterations = 2001:4000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 2000

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

            Mean      SD Naive SE Time-series SE
Trt       0.3153 0.40439 0.009042       0.025264
Time     -0.3891 0.04504 0.001007       0.002327
Trt.Time -0.1236 0.07070 0.001581       0.002793
Meanb    -1.5742 0.65475 0.014641       0.076459
SDb       3.5261 0.64641 0.014454       0.078697

2. Quantiles for each variable:

            2.5%      25%     50%      75%    97.5%
Trt      -0.5288  0.04498  0.3089  0.58201  1.08908
Time     -0.4793 -0.41861 -0.3881 -0.35751 -0.30694
Trt.Time -0.2638 -0.17121 -0.1222 -0.07584  0.01278
Meanb    -3.0730 -1.87187 -1.4623 -1.14067 -0.62062
SDb       2.6909  3.08263  3.3738  3.78198  5.08547
```

Normal GLMM(nhc)

```
> summary(chNormal.nhc)


Iterations = 2001:4000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 2000

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

            Mean      SD  Naive SE Time-series SE
Trt      -0.1539 0.59813 0.0133747       0.019375
Time     -0.3961 0.04465 0.0009985       0.001108
Trt.Time -0.1363 0.06766 0.0015130       0.001673
Meanb    -1.6516 0.44893 0.0100384       0.018702
SDb       4.0557 0.38298 0.0085637       0.010389

2. Quantiles for each variable:
```

```
              2.5%     25%     50%      75%     97.5%
Trt       -1.3806 -0.5444 -0.1262  0.25189  0.978046
Time      -0.4884 -0.4260 -0.3947 -0.36614 -0.313543
Trt.Time  -0.2664 -0.1821 -0.1352 -0.08926 -0.007678
Meanb     -2.5755 -1.9364 -1.6320 -1.34609 -0.846362
SDb        3.3640  3.7857  4.0345  4.31133  4.851494
```

Normal GLMM(hc)

```
> summary(chNormal.hc)


Iterations = 2001:4000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 2000

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

             Mean      SD  Naive SE Time-series SE
Trt       -0.1744 0.60828 0.0136014       0.030140
Time      -0.3954 0.04438 0.0009924       0.001482
Trt.Time  -0.1376 0.06679 0.0014934       0.001957
Meanb     -1.6177 0.43686 0.0097685       0.016118
SDb        4.0529 0.39050 0.0087319       0.012886

2. Quantiles for each variable:

              2.5%     25%     50%      75%     97.5%
Trt       -1.4295 -0.5576 -0.1565  0.23817  1.001406
Time      -0.4858 -0.4240 -0.3937 -0.36463 -0.311575
Trt.Time  -0.2733 -0.1824 -0.1390 -0.09699 -0.001745
Meanb     -2.5021 -1.9068 -1.6202 -1.30989 -0.798895
SDb        3.3827  3.7719  4.0328  4.30002  4.910623
```

5.4 Bayesian P-values

Bayesian P-values as defined in Komárek and Lesaffre (2008) can be computed as follows:

PGM GLMM(nhc)

```
> BPvalue(chPGM.nhc)


      Trt      Time Trt.Time     Meanb       SDb
   0.379     0.000     0.073     0.001     0.000
```

PGM GLMM(hc)

```
> BPvalue(chPGM.hc)
```

```
     Trt      Time Trt.Time     Meanb       SDb
   0.436     0.000    0.081     0.002     0.000
```

Normal GLMM(nhc)

```
> BPvalue(chNormal.nhc)
```

```
     Trt      Time Trt.Time     Meanb       SDb
   0.812     0.000    0.042     0.000     0.000
```

Normal GLMM(hc)

```
> BPvalue(chNormal.hc)
```

```
     Trt      Time Trt.Time     Meanb       SDb
   0.782     0.000    0.048     0.000     0.000
```

5.5 Highest posterior density intervals

Highest posterior density intervals can be computed using the coda function `HPDinterval`:

PGM GLMM(nhc)

```
> HPDinterval(chPGM.nhc, prob = 0.95)
```

```
              lower        upper
Trt      -0.5045359  1.225775590
Time     -0.4742923 -0.298992154
Trt.Time -0.2669074  0.009167199
Meanb    -3.0537721 -0.588356092
SDb       2.7044487  4.718175998
attr(,"Probability")
[1] 0.95
```

PGM GLMM(hc)

```
> HPDinterval(chPGM.hc, prob = 0.95)
```

```
              lower       upper
Trt      -0.5337152  1.08519798
Time     -0.4810197 -0.30866403
Trt.Time -0.2484443  0.02599017
Meanb    -2.8806432 -0.50490536
SDb       2.6311318  4.89183144
attr(,"Probability")
[1] 0.95
```

Normal GLMM(nhc)

```
> HPDinterval(chNormal.nhc, prob = 0.95)
```

```
              lower        upper
Trt      -1.4416230  0.91341958
Time     -0.4884610 -0.31354845
Trt.Time -0.2679827 -0.01188050
Meanb    -2.5777225 -0.84851893
SDb       3.3472936  4.83321265
attr(,"Probability")
[1] 0.95
```

Normal GLMM(hc)

```
> HPDinterval(chNormal.hc, prob = 0.95)
```

```
              lower       upper
Trt      -1.3653932  1.034800950
Time     -0.4861970 -0.311615474
Trt.Time -0.2588812  0.009385655
Meanb    -2.5021196 -0.798933491
SDb       3.3475170  4.853173190
attr(,"Probability")
[1] 0.95
```

The chains can be further processed using the coda package to check for convergence, draw plots, etc. We will skip this in this manual to concentrate more on the issues specific for the glmmAK package.

6 Estimation of the random intercept density in the PGM models

The estimate of the random intercept density in the PGM models can be summarized using the pointwise posterior summary statistics (mean, median, quantiles). To compute these from the sampled chains, we use the function summaryGspline1.

6.1 Standardized version

Firstly, we summarize the standardized version of the random intercept density. That is, when computing the posterior statistics, the random intercept density at each iteration is standardized first to have zero mean and unit variance and summarized afterwards. The pointwise posterior summary statistics will be computed in a grid of points stored in the variable grid. Besides computing pointwise posterior mean, we will also compute pointwise posterior 2.5%, 25%, 50%, 75% and 97.5% quantiles. We will also store the value of the density at each iteration. Note that variables knots and sigma determine the PGM knots $\mu_{-K}, \dots, \mu_K$ and basis standard deviation σ, respectively. Computed posterior summary statistics for the random intercept density will be stored in objects stPGM.nhc and stPGM.hc for PGM GLMM(nhc) and PGM GLMM(hc) model, respectively.

```
> knots <- seq(-4.5, 4.5, by=0.3)
> sigma <- 0.2
> grid <- seq(-2, 4.5, length=300)
>
> ### PGM GLMM(nhc)
> stPGM.nhc <- summaryGspline1(x=grid, mu=knots, sigma=sigma, standard=TRUE,
+                              probs=c(0.025, 0.25, 0.5, 0.75, 0.975),
+                              values=TRUE, dir=dirPaths["PGM_nhc"])
>
> ### PGM GLMM(hc)
> stPGM.hc <- summaryGspline1(x=grid, mu=knots, sigma=sigma, standard=TRUE,
+                             probs=c(0.025, 0.25, 0.5, 0.75, 0.975),
+                             values=TRUE, dir=dirPaths["PGM_hc"])
```

Computed posterior summary statistics and values of the density at each iteration can be plotted as follows, see Figure 1 for the results. The example code below applies for the PGM GLMM(nhc) model.

```
> par(bty = "n", mar = c(4, 4, 1, 1) + 0.1)
> layout(matrix(c(1, 1, 2, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14),
+               ncol = 4, byrow = TRUE))
>
> ### Posterior mean
> plot(stPGM.nhc$summary$x, stPGM.nhc$summary$Mean, type = "l",
+      xlab = "b[st]", ylab = "g(b[st])", col = "blue")
>
> ### Posterior median and 2.5%, 97.5% quantiles
> plot(stPGM.nhc$summary$x, stPGM.nhc$summary[, "97.5%"], type = "l",
+     lty = 2, xlab = "b[st]", ylab = "g(b[st])", col = "red")
> lines(stPGM.nhc$summary$x, stPGM.nhc$summary[, "2.5%"], lty = 2,  col = "red")
> lines(stPGM.nhc$summary$x, stPGM.nhc$summary[, "50%"], lty = 1, col = "blue")
>
> ### Sampled densities at selected iterations
> ylim <- c(0, 2.8)
> for (iters in c(1, 200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 1900, 2000)){
+     plot(stPGM.nhc$summary$x, stPGM.nhc$values[iters, ], type = "l",
+          xlab = "", ylab = "", col = "darkblue", ylim = ylim)
+     title(main = paste("Iteration ", iters, sep = ""))
+ }
```

6.2 Unstandardized version

Unstandardized versions of the random intercept, i.e., taking into account the fixed intercept β_1 in the PGM GLMM(nhc) or the location α in the PGM GLMM(hc) and the scale parameter τ are computed and stored in objects PGM.nhc, PGM.hc as follows.

As before, we specify the PGM knots $\mu_{-K}, \ldots, \mu_K$, basis standard deviation σ and a grid of values where we want to evaluate the random intercept density:

```
> knots <- seq(-4.5, 4.5, by = 0.3)
> sigma <- 0.2
> grid <- seq(-10, 13, length = 300)
```

When computing the unstandardized density the chains for the PGM shift and scale τ must be given. In the PGM GLMM(nhc), the shift is given by the fixed intercept β_1, and its chain is stored in the file betaF.sim, column "`(Intercept)`". In the PGM GLMM(hc), the shift is given by the PGM location parameter α and its chain is stored in the file betaR.sim, column "`(Intercept)`". The chain for the PGM scale parameter τ is obtained as a square root of the column "`varR.1`" in the file varR.sim.

Computation for the **PGM GLMM(nhc)** model proceeds in the following way:

```
> shift.nhc <- scanFH(paste(dirPaths["PGM_nhc"], "betaF.sim", sep = ""))[, "(Intercept)"]
> scale.nhc <- sqrt(scanFH(paste(dirPaths["PGM_nhc"], "varR.sim", sep = ""))[, "varR.1"])
> PGM.nhc <- summaryGspline1(x=grid, mu=knots, sigma=sigma, standard=FALSE,
+                            intcpt=shift.nhc, scale=scale.nhc,
+                            probs=c(0.025, 0.25, 0.5, 0.75, 0.975),
+                            values=TRUE, dir=dirPaths["PGM_nhc"])
```

Computation for the **PGM GLMM(nhc)** model proceeds in the following way:

```
> shift.hc <- scanFH(paste(dirPaths["PGM_hc"], "betaR.sim", sep = ""))[, "(Intercept)"]
> scale.hc <- sqrt(scanFH(paste(dirPaths["PGM_hc"], "varR.sim", sep = ""))[, "varR.1"])
> PGM.hc <- summaryGspline1(x=grid, mu=knots, sigma=sigma, standard=FALSE,
+                           intcpt=shift.hc, scale=scale.hc,
+                           probs=c(0.025, 0.25, 0.5, 0.75, 0.975),
+                           values=TRUE, dir=dirPaths["PGM_hc"])
```

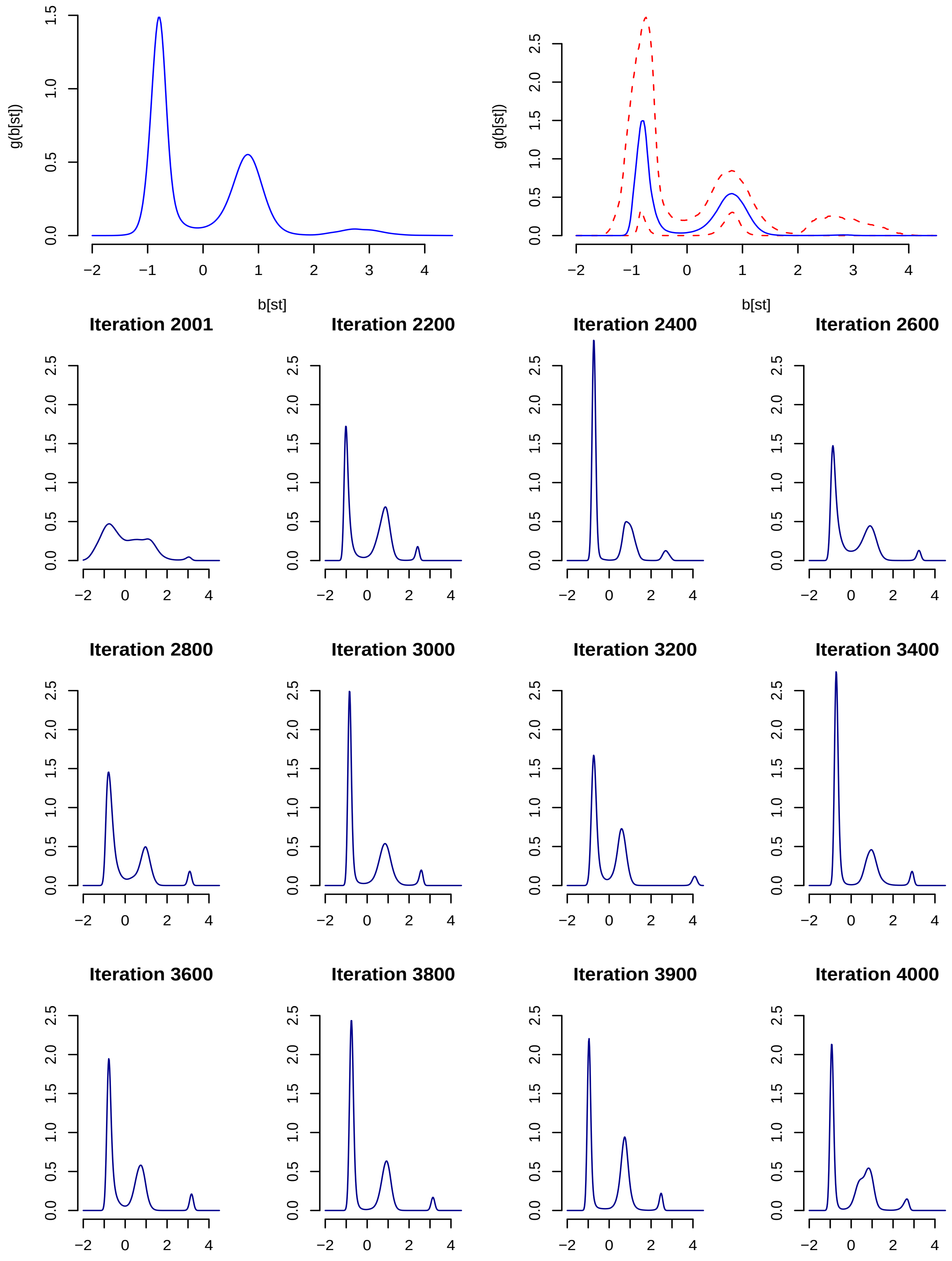

Figure 1: **PGM GLMM(nhc):** Pointwise posterior mean (upper left panel), 2.5%, 50%, 97.5% quantile (upper right panel) of the standardized random intercept density and selected sampled values of this density.

Computed posterior summary statistics and values of the density at each iteration can be plotted similarly as in the standardized case. See Figure 2 for the results. The example below applies for the PGM GLMM(nhc) model.

```
> par(bty = "n", mar = c(4, 4, 1, 1) + 0.1)
> layout(matrix(c(1, 1, 2, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14),
+               ncol = 4, byrow = TRUE))
>
> ### Posterior mean
> plot(PGM.nhc$summary$x, PGM.nhc$summary$Mean, type = "l",
+      xlab = "b", ylab = "g(b)", col = "blue")
>
> ### Posterior median and 2.5%, 97.5% quantiles
> plot(PGM.nhc$summary$x, PGM.nhc$summary[, "97.5%"], type = "l",
+     lty = 2, xlab = "b", ylab = "g(b)", col = "red")
> lines(PGM.nhc$summary$x, PGM.nhc$summary[, "2.5%"], lty = 2,  col = "red")
> lines(PGM.nhc$summary$x, PGM.nhc$summary[, "50%"], lty = 1, col = "blue")
>
> ### Sampled densities at selected iterations
> ylim <- c(0, 1)
> for (iters in c(1, 200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 1900, 2000)){
+     plot(PGM.nhc$summary$x, PGM.nhc$values[iters, ], type = "l",
+          xlab = "", ylab = "", col = "darkblue", ylim = ylim)
+     title(main = paste("Iteration ", iters, sep = ""))
+ }
```

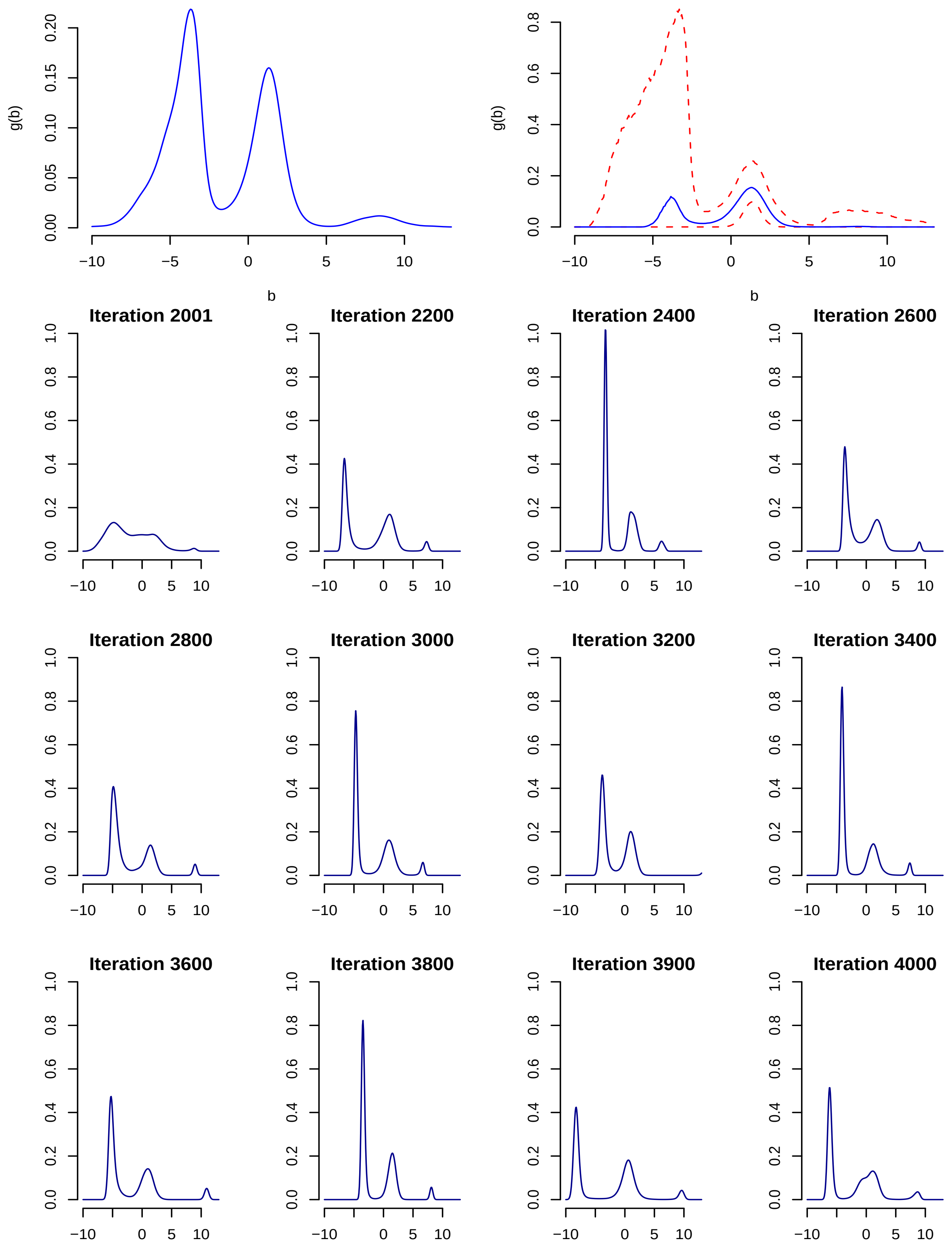

Figure 2: **PGM GLMM(nhc):** Pointwise posterior mean (upper left panel), 2.5%, 50%, 97.5% (upper right panel) quantile of the (unstandardized) random intercept density and selected sampled values of this density.

7 Summary for the values of individual random effects

Sampled values of the individual random effects are stored in the files b.sim. Posterior mean and quantiles can be used to infer on the individual random effects.

In this manual, we will show the results for the **PGM GLMM(nhc)** only. The results for the remaining models would have been obtained analogically. Note that we will compute posterior summary for $\beta_1 + b_i$ $(i = 1, \ldots, N)$, that is for random intercepts shifted by the fixed intercept value.

Firstly, we extract from the original data identification numbers of the patients and divide also these id numbers into two groups according to the treatment.

```
> IDNR <- unique(toenail$idnr)
> IDNR0 <- unique(subset(toenail, trt == 0)$idnr)
> IDNR1 <- unique(subset(toenail, trt == 1)$idnr)
> index.tr0 <- (1:length(IDNR))[IDNR %in% IDNR0]
> index.tr1 <- (1:length(IDNR))[IDNR %in% IDNR1]
```

Now, we read the sampled values of random intercepts and shift them by the sampled fixed intercepts β_1. Note that the sampled fixed intercept values are stored in the column "`(Intercept)`" of the file betaF.sim.

```
> beta1.PGMnhc <- scanFH(paste(dirPaths["PGM_nhc"], "betaF.sim", sep = ""))[, "(Intercept)"]
> b.PGMnhc <- scanFH(paste(dirPaths["PGM_nhc"], "b.sim", sep = "")) +  beta1.PGMnhc
> colnames(b.PGMnhc) <- IDNR
```

We continue by computing posterior mean and median for the individual values of random effects:

```
> bMean.PGMnhc <- apply(b.PGMnhc, 2, mean)
> bMedian.PGMnhc <- apply(b.PGMnhc, 2, median)
```

Finally, we will plot histograms of the posterior means and medians of the random effects, see Figures 3.

```
> xlim <- c(-5, 9)
> ylim <- c(0, 0.55)
> layout(matrix(c(0, 1, 1, 0, 2, 2, 3, 3, 0, 4, 4, 0, 5, 5, 6, 6),
+     ncol = 4, byrow = TRUE))
> par(bty = "n", mar = c(4, 4, 4, 1) + 0.1)
>
> ### Histograms of posterior means
> hist(bMean.PGMnhc, prob = TRUE, xlab = "beta1+b", ylab = "Density",
+     col = "seagreen3", xlim = xlim, ylim = ylim, main = "Posterior mean (all patients)")
> hist(bMean.PGMnhc[index.tr0], prob = TRUE, xlab = "beta1+b", ylab = "Density",
+     col = "skyblue4", xlim = xlim, ylim = ylim, main = "Control")
> hist(bMean.PGMnhc[index.tr1], prob = TRUE, xlab = "beta1+b", ylab = "Density",
+     col = "skyblue4", xlim = xlim, ylim = ylim, main = "Treatment")
```

```
>
> ### Histograms of posterior medians
> hist(bMedian.PGMnhc, prob = TRUE, xlab = "beta1+b", ylab = "Density",
+     col = "seagreen3", xlim = xlim, ylim = ylim, main = "Posterior median (all patients)"
> hist(bMedian.PGMnhc[index.tr0], prob = TRUE, xlab = "beta1+b", ylab = "Density",
+     col = "skyblue4", xlim = xlim, ylim = ylim, main = "Control")
> hist(bMedian.PGMnhc[index.tr1], prob = TRUE, xlab = "beta1+b", ylab = "Density",
+     col = "skyblue4", xlim = xlim, ylim = ylim, main = "Treatment")
```

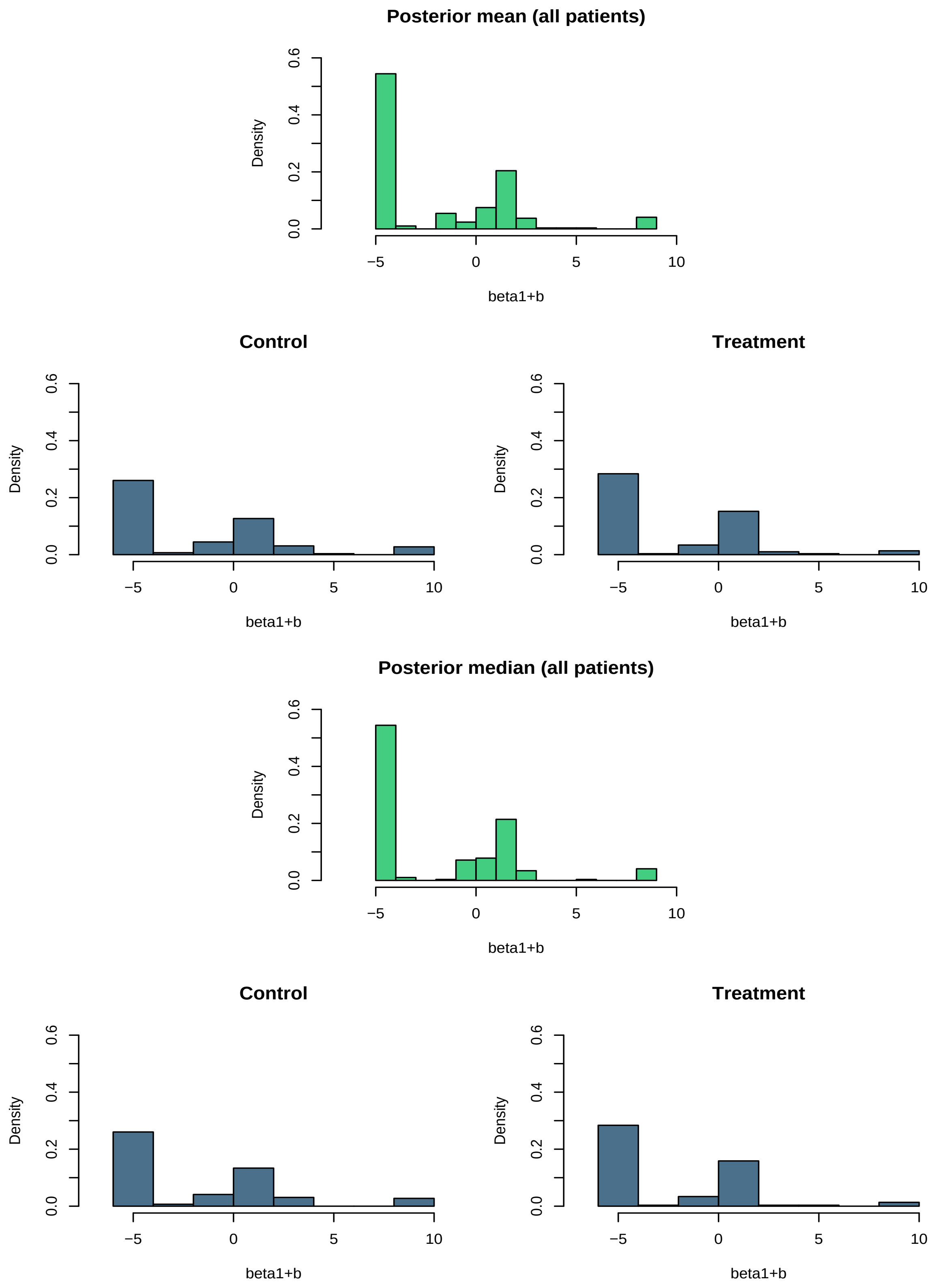

Figure 3: **PGM GLMM(nhc):** Histogram of the posterior means and posterior medians of individual random intercepts shifted by the fixed intercept β_1.

References

De Backer, M., De Vroey, C., Lesaffre, E., Scheys, I., and De Keyser, P. (1998). Twelve weeks of continuous onychomycosis caused by dermatophytes: A double blind comparative trial of terbafine 250 mg/day versus itraconazole 200 mg/day. *Journal of the American Academy of Dermatology*, **38**, S57–S63.

Gelman, A. (2006). Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*, **1**, 515–533.

Komárek, A. and Lesaffre, E. (2008). Generalized linear mixed model with a penalized Gaussian mixture as a random effects distribution. *Computational Statistics and Data Analysis*, **52**, 3441–3458.

Neal, R. M. (2003). Slice sampling (with Discussion). *The Annals of Statistics*, **31**, 705–767.

Plummer, M., Best, N., Cowles, K., and Vines, K. (2006). *coda: Output analysis and diagnostics for MCMC.* R package version 0.10-7.

Rue, H. and Held, L. (2005). *Gaussian Markov Random Fields: Theory and Applications.* Chapman & Hall/CRC, Boca Raton. ISBN 978-1-58488-432-3.