CE-227: Inferência Bayesiana – 4^a Avaliação Semanal (16/04/2014)

GRR: _____ **Nome:** _____ **Turma:** _____

Descreva o modelo sendo ajustado e a estrutura dos dados nas seguintes declarações de modelos em JAGS.

1.
```
model{
    for (i in 1:N){
      x[i] ~ dbern(p)
    }
    p ~ dbeta(alpha, beta)
    alpha <- 1
    beta <- 1
  }
```

Modelo binomial (= ensaios Bernoulli independentes):

$$\begin{aligned} &\text{verossimilhança:} \\ &\qquad X_i|p \sim \text{Bern}(p_i) \\ &\text{priori:} \\ &\qquad p_i \sim \text{Be}(\alpha = 1, \beta = 1)(\equiv \text{U}[0,1]) \end{aligned}$$

Simulação de dados do modelo

```
> set.seed(227)
> p <- runif(1, 0, 1)
> n <- 30
> x <- rbinom(30, size=1, prob=p)
> #Elias (define the list data here instead later, like in ex2)
> q1.dat <- list(x = x, N = length(x))
```

- Estimação não-Bayesiana
  i. Solução analítica (MLE) é disponível neste caso

$$\begin{aligned} \hat{p} &= \frac{\sum_i x_i}{n} = 0.7 \\ \text{sd}(\hat{p}) &= \sqrt{1/I_o(\hat{p})} = \sqrt{\hat{p}(1-\hat{p})/n} = 0.0837 \end{aligned}$$

  ii. (Algumas) soluções numéricas

```
> qa.glm <- glm(x ~ 1, family=binomial(link="logit"))
> unlist(predict(qa.glm, newdata = data.frame(c(1)), type="response", se.fit=TRUE)[1:2])
   fit.1 se.fit.1
 0.70000  0.08367
> #
> lik <- function(p, data, log=TRUE){
+     dbinom(sum(data), size=length(data), prob=p, log=log)
+ }
> (qa.optim <- optimize(lik, interval=c(0,1), data=x, maximum=TRUE))
$maximum
[1] 0.7

$objective
[1] -1.85
> with(qa.optim, sqrt(-1/drop(optimHess(par=maximum, fn=lik, data=x))))
[1] 0.08366
```

- Estimação Bayesiana

i. Posteriori analítica (conjugada neste caso)

$$p|x \sim \text{Be}(\alpha^* = \alpha + \sum_i x_i = 1 + 21 = 22, \beta^* = \beta + n - \sum_i x_i = 1 + 30 - 21 = 10)$$

$$\text{E}[p|x] = \frac{\alpha^*}{\alpha^* + \beta^*} = \frac{\alpha + \sum_i x_i}{\alpha + \beta + n} = 0.688$$

$$\text{Mo}[p|x] = \frac{\alpha^* - 1}{\alpha^* + \beta^* - 2} = \frac{\alpha + \sum_i x_i - 1}{\alpha + \beta + n - 2} = 0.7$$

$$\text{Var}[p|x] = \frac{\alpha^* \beta^*}{(\alpha^* + \beta^*)^2(\alpha^* + \beta^* + 1)} = 0.00651$$

$$\text{SD}[p|x] = 0.0807$$

Pelo resultado da inferência conjugada vamor armazenar os parâmetros da posteriori nos objetos a seguir.

```
> (a1 <- 1 + sum(x))
[1] 22
> (b1 <- 1 + length(x) - sum(x))
[1] 10
> (var.post <- (a1*b1)/(((a1+b1)^2)*(a1+b1+1)))
[1] 0.00651
```

ii. Amostras (MCMC) - JAGS

```
> require(rjags)
> cat("model{
+   for (i in 1:N){
+     x[i] ~ dbern(p)
+   }
+   p ~ dbeta(alpha, beta)
+   alpha <- 1
+   beta <- 1
+   }", file="av04-q1.jags")
> inis <- list(list(p=0.1), list(p=0.5), list(p=0.9))
> q1.model <- jags.model(file="av04-q1.jags", data=q1.dat, n.chains=3, init = inis)
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 34

Initializing model
> q1.sam <- coda.samples(q1.model, c("p"), 20000, thin=10)
> summary(q1.sam)
Iterations = 10:20000
Thinning interval = 10
Number of chains = 3
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
       0.68814        0.08141        0.00105        0.00101

2. Quantiles for each variable:

 2.5%   25%   50%   75% 97.5%
0.516 0.634 0.694 0.746 0.835
```

iii. INLA

```
> require(INLA)
> q1.inla <- inla(x ~ 1, data=q1.dat, family="binomial", control.family=list(link="logit"),
+                 control.predictor=list(compute=TRUE))
> summary(q1.inla)
```

```
Call:
c("inla(formula = x ~ 1, family = \"binomial\", data = q1.dat, control.predictor = list(compute = TRU

Time used:
 Pre-processing    Running inla Post-processing           Total
         0.1673          0.0140          0.0228          0.2041

Fixed effects:
              mean     sd 0.025quant 0.5quant 0.975quant   mode kld
(Intercept) 0.8473 0.3984     0.0972   0.8354      1.664 0.8112   0

The model has no random effects

The model has no hyperparameters

Expected number of effective parameters(std dev): 1.00(0.00)
Number of equivalent replicates : 30.00

Marginal Likelihood:  -18.33
Posterior marginals for linear predictor and fitted values computed
```

```
> q1.inla$summary.fitted.values[1,]
                       mean      sd 0.025quant 0.5quant 0.975quant   mode
fitted.predictor.01 0.6936 0.08143     0.5245   0.6975     0.8407 0.7057
> inla.zmarginal(q1.inla$marginals.fitted.values[[1]])
Mean            0.693629
Stdev           0.081431
Quantile  0.025 0.524297
Quantile  0.25  0.639414
Quantile  0.5   0.697183
Quantile  0.75  0.751411
Quantile  0.975 0.840348
```

O modelo ajustado pelo INLA não corresponde exatamente ao do JAGS. No INLA, os coeficientes de regressão são tratados da mesma forma que os efeitos aleatórios. Então, a priori é necessariamente gaussiana (no caso, foi usada a priori vaga, opção *default*), pois o algoritmo é baseado nisso. O modelo ajustado pelo INLA é então:
Modelo binomial (= ensaios Bernoulli independentes):

verossimilhança:

$$X_i|p \sim \text{Bern}(p_i)$$

$$\log\left(\frac{p}{1-p}\right) = \beta_0 \text{priori:}$$

$$\beta_0 \sim \text{N}(0, +\infty)(\equiv \text{U}[-\infty, +\infty])$$

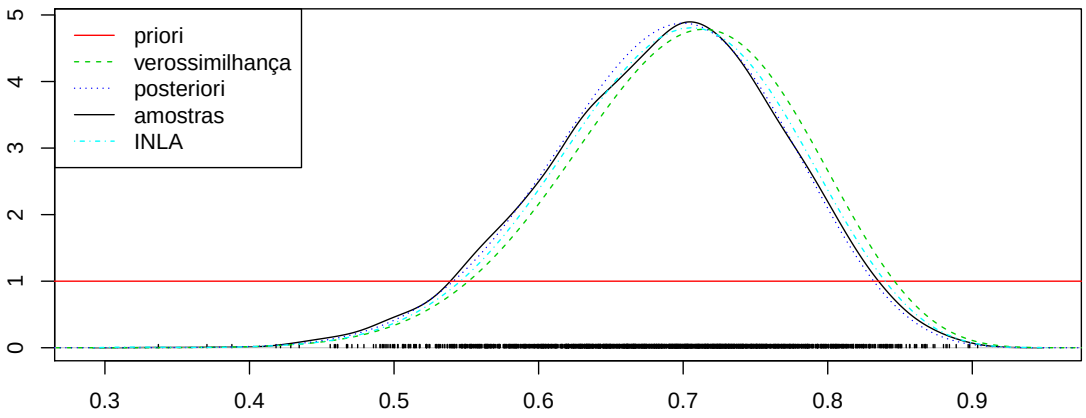
Fica como exercício reescrever o código JAGS que corresponda ao modelo ajustado pelo INLA.
Os comandos a seguir mostram como explorar os resultados armazenados nos objetos.

```
> l.temp <- q1.inla$marginals.fixed[[1]][,1]
> dl.temp <- q1.inla$marginals.fixed[[1]][,2]
> plot(dl.temp ~ l.temp, type="l")
> plot(spline(dl.temp ~ l.temp), type="l")
> p.temp <- exp(l.temp)/(1+exp(l.temp))
> dp.temp <- dl.temp * ((1/p.temp) + (1/(1-p.temp)))
> plot(spline(dp.temp ~ p.temp), type="l")
> ## os dois comandos a seguir são equivalentes:
> with(as.data.frame(q1.inla$marginals.fitted.values[[1]]), lines(spline(y ~ x), col=6))
> ## as transformações acima são feitas internamente pela smarginal:
> lines(inla.smarginal(q1.inla$marginals.fitted.values[[1]]), col=6)
```

Gráficos da priori, verossimilhança e posteriori

```
> p.vals <- seq(0,1,l=501)
> pr.vals <- cbind(dbeta(p.vals, 1, 1), dbeta(p.vals, sum(x), length(x)-sum(x)),dbeta(p.vals, a1, b1))
> densplot(q1.sam, main="", xlab="")
> matlines(p.vals, pr.vals, type="l", xlab="p", ylab="", col=2:4)
> lines(spline(dp.temp ~ p.temp), col=5, lty=4)
```

```
> legend("topleft", c("priori","verossimilhança","posteriori", "amostras", "INLA"), lty=c(1:3,1,4),
+        col=c(2:4,1,5))
```



2.
```
model{
  for(i in 1:M){
     for(j in 1:N){
        y[i,j] ~ dnorm(mu[i], tau)
     }
     mu[i] ~ dnorm(theta, tauD)
  }
  tau <- pow(sigma, -2)
  sigma ~ dunif(0, 100)
  theta ~ dnorm(0, .001)
  tauD <- pow(delta, -2)
  delta ~ dunif(0, 100)
}
```

Modelo (medidas repetidas):

$$\begin{aligned} \text{verossimilhança:} \\ Y_{ij}|\mu_i, \sigma^2 &\sim \mathrm{N}(\mu_i, \sigma^2) \\ \text{variáveis latentes (ef. al.):} \\ \mu_i &\sim \mathrm{N}(\theta, \delta^2) \\ \text{priori's:} \\ \theta &\sim \mathrm{N}(0, 1000) \\ \sigma &\sim \mathrm{U}[0, 100] \\ \delta &\sim \mathrm{U}[0, 100] \end{aligned}$$

Simulação de dados do modelo proposto

```
> set.seed(22701)
> Ng <- 10
> Nobs <- 5
> N <- Ng*Nobs
> delta <- 5
> sigma <- 2
> mus <- rnorm(Ng, mean=50, sd=delta)
> y <- matrix(rnorm(N, mean=mus, sd=sigma), ncol=Ng, byrow=TRUE)
> q2.df <- data.frame(y = as.vector(y), grupo = rep(1:Ng, each=Nobs))
```

- Estimação não-Bayesiana

```
> require(lme4)
> (q2.lmer <- lmer(y ~ 1|grupo, data=q2.df, REML=FALSE))
```

```
Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula: y ~ 1 | grupo
   Data: q2.df
     AIC      BIC   logLik deviance df.resid
   261.3    267.0   -127.6    255.3       47
Random effects:
 Groups   Name        Std.Dev.
 grupo    (Intercept) 3.90
 Residual             2.38
Number of obs: 50, groups: grupo, 10
Fixed Effects:
(Intercept)
       49.6
```

```
> t(ranef(q2.lmer)$grupo)
```

```
                 1      2     3     4      5     6      7      8     9    10
(Intercept) -3.104 -0.207 1.287 5.898 -4.738 2.586 -4.984 -3.535 1.693 5.104
```

```
> t(ranef(q2.lmer)$grupo +  fixef(q2.lmer)[[1]])
```

```
                 1     2     3     4     5     6     7     8     9    10
(Intercept) 46.46 49.36 50.85 55.47 44.83 52.15 44.58 46.03 51.26 54.67
```

- Estimação Bayesiana
  - JAGS

```
> require(rjags)
> cat("model{
+   for(i in 1:M){
+      for(j in 1:N){
+         y[i,j] ~ dnorm(mu[i], tau)
+      }
+      mu[i] ~ dnorm(theta, tauD)
+   }
+   tau <- pow(sigma, -2)
+   sigma ~ dunif(0, 100)
+   theta ~ dnorm(0, .001)
+   tauD <- pow(delta, -2)
+   delta ~ dunif(0, 100)
+ }", file="av04-q2.jags")
> q2.dat <- list(y = t(y), M = Ng, N = Nobs)
> q2.model <- jags.model(file="av04-q2.jags", data=q2.dat, n.chains=3)
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 72

Initializing model
```

```
> q2.sam <- coda.samples(q2.model, c("mu", "sigma", "delta", "theta"), 20000, thin=10)
> summary(q2.sam)
```

```
Iterations = 1010:21000
Thinning interval = 10
Number of chains = 3
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
delta    4.86 1.490  0.01923        0.01923
mu[1]   46.41 1.082  0.01397        0.01380
mu[2]   49.35 1.079  0.01393        0.01469
mu[3]   50.86 1.078  0.01392        0.01318
mu[4]   55.51 1.087  0.01404        0.01382
mu[5]   44.78 1.096  0.01415        0.01495
mu[6]   52.18 1.088  0.01404        0.01455
```

```
mu[7]  44.54 1.091  0.01408        0.01437
mu[8]  45.98 1.101  0.01421        0.01421
mu[9]  51.29 1.069  0.01380        0.01399
mu[10] 54.69 1.101  0.01421        0.01428
sigma   2.47 0.288  0.00371        0.00378
theta  49.43 1.657  0.02139        0.02073

2. Quantiles for each variable:

        2.5%   25%   50%   75% 97.5%
delta   2.84  3.83  4.59  5.52  8.52
mu[1]  44.28 45.70 46.39 47.14 48.57
mu[2]  47.21 48.64 49.35 50.07 51.46
mu[3]  48.76 50.13 50.88 51.60 52.88
mu[4]  53.41 54.79 55.50 56.24 57.64
mu[5]  42.70 44.04 44.76 45.49 46.97
mu[6]  50.05 51.45 52.17 52.92 54.28
mu[7]  42.40 43.82 44.53 45.26 46.71
mu[8]  43.81 45.25 45.98 46.70 48.12
mu[9]  49.15 50.59 51.32 52.02 53.30
mu[10] 52.53 53.97 54.68 55.44 56.86
sigma   1.98  2.26  2.44  2.64  3.09
theta  45.96 48.44 49.45 50.45 52.64
```

– INLA

```
> require(INLA)
> q2.inla <- inla(y ~  f(grupo, model="iid"), family="gaussian", data=q2.df)
> summary(q2.inla)
Call:
c("inla(formula = y ~ f(grupo, model = \"iid\"), family = \"gaussian\", ",  "    data = q2.df)")

Time used:
 Pre-processing    Running inla Post-processing           Total
         0.0755          0.0373          0.0232          0.1360

Fixed effects:
             mean    sd 0.025quant 0.5quant 0.975quant  mode kld
(Intercept) 49.57 1.281         47    49.57      52.13 49.57   0

Random effects:
Name          Model
 grupo   IID model

Model hyperparameters:
                                        mean   sd     0.025quant 0.5quant 0.975quant
Precision for the Gaussian observations 0.1846 0.0406 0.1155     0.1813   0.2740
Precision for grupo                     0.0764 0.0360 0.0261     0.0700   0.1646
                                        mode
Precision for the Gaussian observations 0.1753
Precision for grupo                     0.0572

Expected number of effective parameters(std dev): 9.262(0.3351)
Number of equivalent replicates : 5.398

Marginal Likelihood:  -151.11
> ## passando de precisão para variância:
> sqrt(1/q2.inla$summary.hyperpar[,1])
Precision for the Gaussian observations                    Precision for grupo
                                 2.327                                  3.619
```

No **INLA** há funções para manipular as *posterior marginal distribution's (PMDs)* e extrair medidas resumo.

```
> post.sigma = inla.tmarginal(function(x) sqrt(1/x), q2.inla$marginals.hyperpar[[1]])
> post.delta = inla.tmarginal(function(x) sqrt(1/x), q2.inla$marginals.hyperpar[[2]])
> rbind(delta=inla.zmarginal(post.delta, T), sigma=inla.zmarginal(post.sigma, T))
```

```
      mean   sd     quant0.025 quant0.25 quant0.5 quant0.75 quant0.975
delta 3.923 0.9421 2.474      3.243     3.777    4.45      6.156
sigma 2.369 0.2602 1.914      2.184     2.348    2.533     2.935
```

Efeitos aleatórios: médias e modas por grupos.

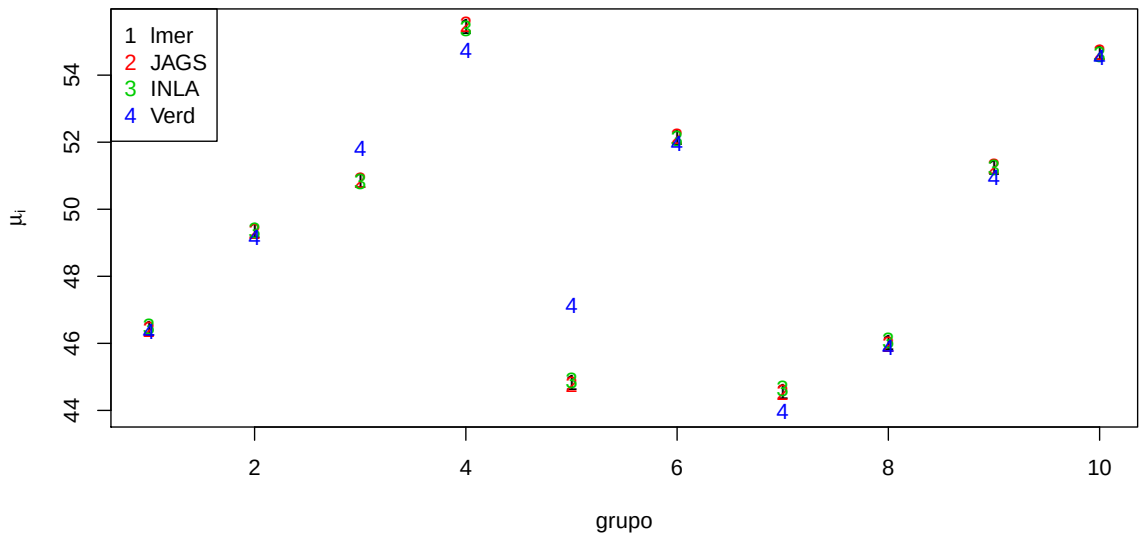
```
> rbind(medias.grupos=q2.inla$summary.fix[1,1] + q2.inla$summary.random$grupo$mean,
+       modas.grupos=q2.inla$summary.fix[1,6] + q2.inla$summary.random$grupo$mode)
                [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10]
medias.grupos 46.51 49.36 50.84 55.39 44.89 52.12 44.65 46.08 51.24 54.60
modas.grupos  46.54 49.37 50.82 55.32 44.95 52.09 44.71 46.12 51.22 54.54
```

Comparação dos valores das variâncias dos grupos e residual.

```
> par(mfrow=c(1,2))
> plot(density(unlist(q2.sam[,"delta",drop=T])), main="", xlab=expression(delta),
+      ylab=expression(group("[",paste(delta,"|",y),"]")), ylim=c(0, 0.5))
> lines(post.delta, lty=2, col=2)
> abline(v=c(as.data.frame(VarCorr(q2.lmer))$sdcor[1], delta), col=3:4, lty=3:4)
> legend("topright", c("JAGS","INLA", "LMER", "Verd."), col=1:4, lty=1:4)
> plot(density(unlist(q2.sam[,"sigma",drop=T])), main="", xlab=expression(sigma),
+      ylab=expression(group("[",paste(sigma,"|",y),"]")), ylim=c(0, 1.5))
> lines(post.sigma, lty=2, col=2)
> abline(v=c(as.data.frame(VarCorr(q2.lmer))$sdcor[2], sigma), col=3:4, lty=3:4)
> legend("topright", c("JAGS","INLA", "LMER", "Verd."), col=1:4, lty=1:4)
```

Médias dos grupos

```
> (q2.grupos <- data.frame(lmer = drop(t(ranef(q2.lmer)$grupo +  fixef(q2.lmer)[[1]])),
+                          JAGS = summary(q2.sam)[[1]][2:11,1],
+                          INLA = q2.inla$summary.fix[1,1] + q2.inla$summary.random$grupo$mean,
+                          verd = mus))
    lmer  JAGS  INLA  verd
1  46.46 46.41 46.51 46.35
2  49.36 49.35 49.36 49.15
3  50.85 50.86 50.84 51.81
4  55.47 55.51 55.39 54.76
5  44.83 44.78 44.89 47.13
6  52.15 52.18 52.12 51.98
7  44.58 44.54 44.65 43.97
8  46.03 45.98 46.08 45.89
9  51.26 51.29 51.24 50.95
10 54.67 54.69 54.60 54.53
> matplot(1:10, q2.grupos, type="p", xlab="grupo", ylab=expression(mu[i]))
> legend("topleft", c("lmer","JAGS","INLA", "Verd"),pch=as.character(1:4), col=1:4)
```



3.
```
model{
  for (i in 1:N){
    y[i] ~ dbern(p[i])
    logit(p[i]) <- a[g[i]] * x[i]
}
for (j in 1:K){
  a[j] ~ dnorm(mu.a, tau.a)
}
mu.a ~ dnorm(0, 0.0001)
tau.a <- pow(sigma.a, -2)
sigma.a ~ dunif(0, 1000)
}
```

Modelo (GLMM - modelo linear generalizado misto):

$$\begin{aligned} \text{verossimilhança:} & \\ Y_i|p_i &\sim \text{Ber}(p_i) \\ \log\left(\frac{p}{1-p}\right) &= a_i x_i \\ \text{variáveis latentes (ef. al.):} & \\ a_i &\sim \text{N}(\mu_a, \sigma_a^2) \\ \text{priori's:} & \\ \mu_a &\sim \text{N}(0, 10000) \\ \sigma &\sim \text{U}[0, 100] \end{aligned}$$

Simulação de dados do modelo proposto

```
> set.seed(22701)
> Ng <- 15
> x <- seq(-4, 4, l=15)
> Nx <- length(x)
> N <- Nx*Ng
> as <- rnorm(Ng, mean=0.5, sd=0.2)
> nus <- as.vector(outer(x, as))
> ps <- exp(nus)/(1+exp(nus))
> q3.df <- data.frame(y = rbinom(N, size=1, prob=ps), x = x,  grupo = rep(1:Ng, each=Nx))
```

- Estimação não-Bayesiana

```
> require(lme4)
> q3.glmer <- glmer(y ~ 0 + x + (0+x|grupo), family=binomial, data=q3.df)
> summary(q3.glmer)
```

```
Generalized linear mixed model fit by maximum likelihood (Laplace Approximation) [
glmerMod]
 Family: binomial ( logit )
Formula: y ~ 0 + x + (0 + x | grupo)
   Data: q3.df

     AIC      BIC   logLik deviance df.resid
   256.1    263.0   -126.1    252.1      223

Scaled residuals:
   Min     1Q Median     3Q    Max
-2.474 -0.596  0.356  0.754  3.040

Random effects:
 Groups Name Variance Std.Dev.
 grupo  x    0.0384   0.196
Number of obs: 225, groups: grupo, 15

Fixed effects:
  Estimate Std. Error z value Pr(>|z|)
x   0.5062     0.0951    5.32    1e-07 ***
```

```
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

- Estimação Bayesiana
  - JAGS

```
> require(rjags)
> cat("model{
+   for (i in 1:N){
+     y[i] ~ dbern(p[i])
+     logit(p[i]) <- a[g[i]] * x[i]
+ }
+ for (j in 1:K){
+   a[j] ~ dnorm(mu.a, tau.a)
+ }
+ mu.a ~ dnorm(0, 0.0001)
+ tau.a <- pow(sigma.a, -2)
+ sigma.a ~ dunif(0, 1000)
+ }", file="av04-q3.jags")
> q3.dat <- list(y = q3.df$y, x = q3.df$x, g = q3.df$grupo, N = N, K=Ng)
> q3.model <- jags.model(file="av04-q3.jags", data=q3.dat, n.chains=3)
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 1150

Initializing model
> q3.sam <- coda.samples(q3.model, c("a", "mu.a", "sigma.a"), 20000, thin=10)
> summary(q3.sam)
Iterations = 1010:21000
Thinning interval = 10
Number of chains = 3
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
a[1]     0.806 0.334  0.00431        0.00671
a[2]     0.555 0.202  0.00260        0.00283
a[3]     0.464 0.187  0.00242        0.00249
a[4]     0.536 0.195  0.00252        0.00281
a[5]     0.458 0.187  0.00241        0.00269
a[6]     0.476 0.188  0.00243        0.00243
a[7]     0.330 0.186  0.00240        0.00290
a[8]     0.596 0.217  0.00280        0.00337
a[9]     0.281 0.195  0.00252        0.00342
a[10]    0.725 0.282  0.00365        0.00535
a[11]    0.670 0.250  0.00322        0.00426
a[12]    0.556 0.205  0.00265        0.00291
a[13]    0.312 0.190  0.00245        0.00310
a[14]    0.740 0.290  0.00374        0.00522
a[15]    0.440 0.185  0.00238        0.00238
mu.a     0.529 0.114  0.00147        0.00190
sigma.a 0.261 0.148  0.00191        0.00399

2. Quantiles for each variable:

           2.5%   25%   50%   75% 97.5%
a[1]     0.3492 0.563 0.745 0.976 1.626
a[2]     0.2050 0.420 0.533 0.669 1.005
a[3]     0.1002 0.343 0.458 0.578 0.850
a[4]     0.1875 0.408 0.519 0.647 0.969
a[5]     0.0819 0.341 0.454 0.572 0.850
```

```
a[6]      0.1070 0.358 0.470 0.587 0.885
a[7]     -0.0583 0.211 0.343 0.458 0.664
a[8]      0.2399 0.453 0.569 0.716 1.094
a[9]     -0.1309 0.149 0.296 0.426 0.621
a[10]     0.3129 0.521 0.673 0.881 1.398
a[11]     0.2838 0.494 0.627 0.808 1.271
a[12]     0.1970 0.422 0.532 0.672 1.015
a[13]    -0.0789 0.185 0.325 0.448 0.646
a[14]     0.3309 0.529 0.684 0.901 1.441
a[15]     0.0754 0.324 0.438 0.549 0.830
mu.a      0.3267 0.452 0.518 0.594 0.778
sigma.a   0.0237 0.155 0.248 0.347 0.595
```

– INLA Em **INLA** é possível definir uma expressão para distribuições a priori `http://www.math.ntnu.no/inla/r-inla.org/doc/prior/expression.pdf`, bem como matriz de precisão para efeitos aleatórios [1]. Neste exemplo, atribuímos uma distribuiç ao $U(0, 100)$ para σ_a, tendo em conta que em **INLA** considera-se logaritmo de precisão.

```
> require(INLA)
> unif.prior = "expression:
+ a = 0;
+ b = 100;
+ sigma = sqrt(exp(-log_precision));
+ return(sigma/(b-a));"
> h.u <- list(theta=list(prior=unif.prior))
```

Neste exemplo temos um efeito aleatório multiplicando uma covariável. Para estimar um efeito aleatório desta forma, passamos a covariável como segundo argumento de `f()`.

```
> q3.inla <- inla(y ~ 0 + x + f(grupo, x, model='iid', hyper=h.u),
+                 family='binomial', data=q3.df,
+                 control.predictor=list(compute=TRUE),
+                 control.fixed=list(mean.intercept=0, prec.intercept=1/10000))
```

Resumos da distribuição marginal do parâmetro de variância dos efeitos aleatório e as médias das posterioris dos efeitos.

```
> post.sigma2 <- inla.tmarginal(function(x) (1/x), q3.inla$marginals.hy[[1]])
> inla.zmarginal(post.sigma2)
```

```
Mean            0.00369106
Stdev           0.00529212
Quantile  0.025 5.4807e-05
Quantile  0.25  0.00134194
Quantile  0.5   0.00289396
Quantile  0.75  0.00443987
Quantile  0.975 0.0110441
```

```
> q3.inla$summary.ran$grupo$mean
```

```
 [1]  0.2050485  0.0157142 -0.0634523 -0.0008034 -0.0634547 -0.0482783 -0.1759007
 [8]  0.0499156 -0.2149608  0.1429919  0.1043079  0.0157161 -0.1890524  0.1631771
[15] -0.0783434
```

4.
```
model{
  for (i in 1:N){
    y[i] ~ dnorm(mu[i], tau)
    mu[i] <- a[g[i]] * x[i] + b[g[i]]
}
for (j in 1:K){
  a[j] ~ dnorm(mu.a, tau.a)
  b[j] ~ dnorm(mu.b, tau.b)
}
mu.a ~ dnorm(0, 0.0001)
mu.b ~ dnorm(0, 0.0001)
tau <- pow(sigma, -2)
sigma ~ dunif(0, 1000)
tau.a <- pow(sigma.a, -2)
```

[1] http://www.math.ntnu.no/inla/r-inla.org/doc/latent/rgeneric.pdf

```
> plot(ranef(q3.glmer)$grupo$x, q3.inla$summary.ran$grupo$mean)
> abline(0,1)
```

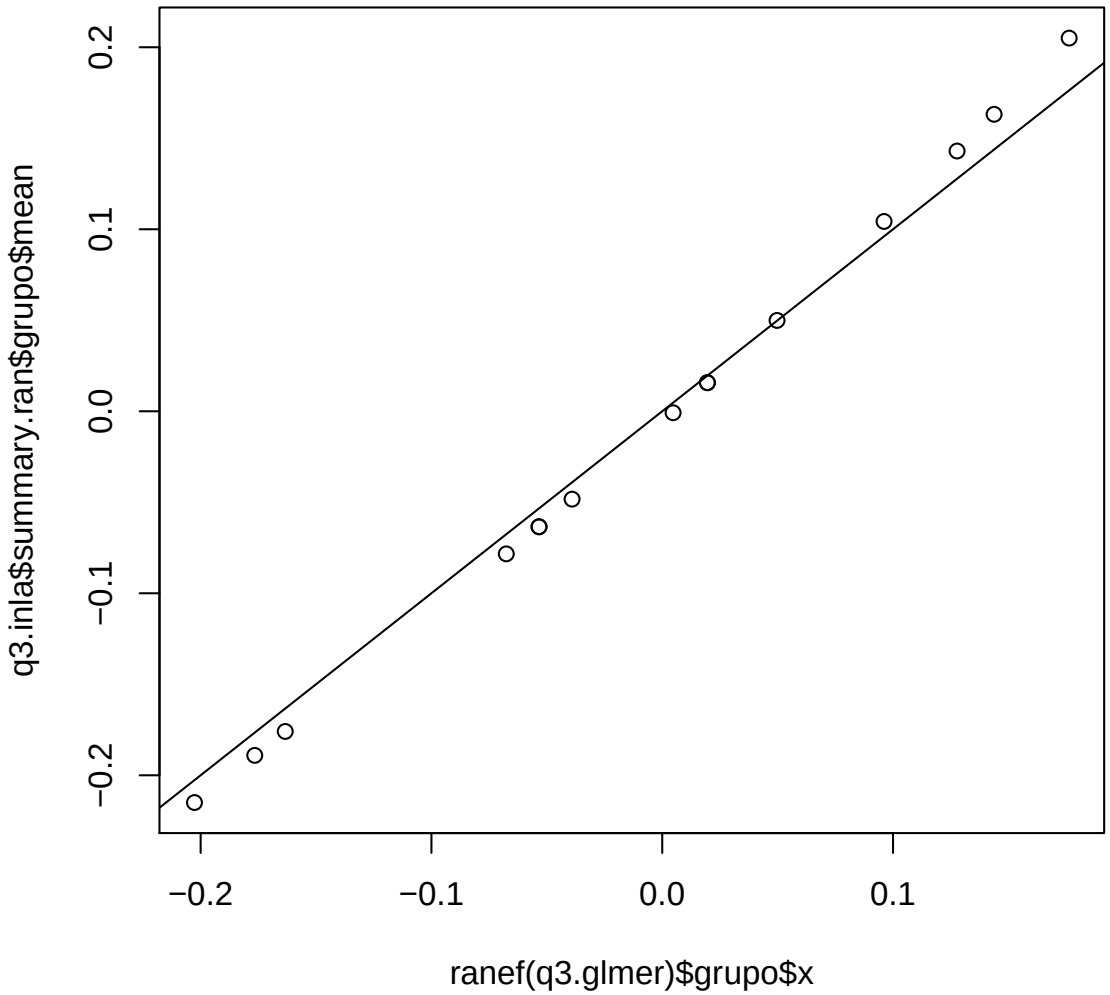


Figura 1: Comparação dos efeitos aleatórios: estimativas do `glmer()` e médias da posteriori retornada por `inla()`.

```
tau.b <- pow(sigma.b, -2)
sigma.a ~ dunif(0, 1000)
sigma.b ~ dunif(0, 1000)
}
```

Modelo misto de regressão linear (intercepto e inclinação aleatórios):

$$
\begin{aligned}
&\text{verossimilhança:}\\
Y_i|\mu_i,\sigma^2 &\sim \mathrm{N}(\mu_i,\sigma^2)\\
\mu_i &= a_i x_i + b_i\\
&\text{variáveis latentes (ef. al.):}\\
a_j &\sim \mathrm{N}(\mu_a,\sigma_a^2)\\
b_j &\sim \mathrm{N}(\mu_b,\sigma_b^2)\\
&\text{priori's:}\\
\mu_a &\sim \mathrm{N}(0,10000)\\
\mu_b &\sim \mathrm{N}(0,10000)\\
\sigma_a &\sim \mathrm{U}[0,1000]\\
\sigma_b &\sim \mathrm{U}[0,1000]
\end{aligned}
$$

Simulação de dados do modelo proposto, com diferentes números de observações e valores da covariável para cada indivíduo.

```
> set.seed(22701)
> Nind <- 12
> (n.ind <- sample(5:12, 12, replace=TRUE))

 [1]  6  6  8  6 10 12 11  7  7  7 10  6

> q4.df <- data.frame(
+   g = rep(1:Nind, times=n.ind),
+   x = unlist(sapply(n.ind, function(n) round(sort(sample(1:20, n)))))
+   )
> ais <- rnorm(Nind, m = -1, sd = 0.2)
> bis <- rnorm(Nind, m = 50, sd = 5)
> q4.df <- transform(q4.df,
+           y = round(rnorm(nrow(q4.df), m = rep(ais, n.ind) * x + rep(bis, n.ind),
+             sd = 3), dig=2))
```

- Estimação não-Bayesiana

```
> require(lme4)
> q4.lmer <- lmer(y ~ x + (x|g), data=q4.df, REML=FALSE)
> summary(q4.lmer)
```

```
Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula: y ~ x + (x | g)
   Data: q4.df

     AIC      BIC   logLik deviance df.resid
   544.2    559.6   -266.1    532.2       90

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.1760 -0.6353  0.0318  0.5572  2.7760

Random effects:
 Groups   Name        Variance Std.Dev. Corr
 g        (Intercept) 52.7804  7.265
          x            0.0136  0.117    0.23
 Residual              8.8042  2.967
Number of obs: 96, groups: g, 12

Fixed effects:
            Estimate Std. Error t value
```

```
(Intercept)  50.8563     2.1986    23.1
x            -1.0014     0.0644   -15.6

Correlation of Fixed Effects:
  (Intr)
x -0.111
```

- Estimação Bayesiana
  - JAGS

```
> require(rjags)
> cat("model{
+   for (i in 1:N){
+     y[i] ~ dnorm(mu[i], tau)
+     mu[i] <- a[g[i]] * x[i] + b[g[i]]
+ }
+ for (j in 1:K){
+   a[j] ~ dnorm(mu.a, tau.a)
+   b[j] ~ dnorm(mu.b, tau.b)
+ }
+ mu.a ~ dnorm(0, 0.0001)
+ mu.b ~ dnorm(0, 0.0001)
+ tau <- pow(sigma, -2)
+ sigma ~ dunif(0, 1000)
+ tau.a <- pow(sigma.a, -2)
+ tau.b <- pow(sigma.b, -2)
+ sigma.a ~ dunif(0, 1000)
+ sigma.b ~ dunif(0, 1000)
+ }", file="av04-q4.jags")
> q4.dat <- c(q4.df, N=nrow(q4.df), K = length(unique(q4.df$g)))
> q4.model <- jags.model(file="av04-q4.jags", data=q4.dat, n.chains=3)
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 519

Initializing model
> q4.sam <- coda.samples(q4.model, c("b", "a", "sigma.b","sigma.a","sigma"), 20000, thin=10)
> summary(q4.sam)
Iterations = 1010:21000
Thinning interval = 10
Number of chains = 3
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
a[1]    -1.148 0.1749  0.00226        0.00351
a[2]    -1.073 0.1332  0.00172        0.00226
a[3]    -0.985 0.1213  0.00157        0.00172
a[4]    -0.934 0.1578  0.00204        0.00229
a[5]    -0.873 0.1319  0.00170        0.00254
a[6]    -0.925 0.1125  0.00145        0.00173
a[7]    -1.076 0.1189  0.00153        0.00194
a[8]    -1.071 0.1329  0.00172        0.00208
a[9]    -1.107 0.1440  0.00186        0.00286
a[10]   -0.922 0.1343  0.00173        0.00227
a[11]   -0.899 0.1407  0.00182        0.00251
a[12]   -1.026 0.1443  0.00186        0.00218
b[1]    45.729 2.0764  0.02681        0.03775
b[2]    56.399 1.9402  0.02505        0.02946
b[3]    35.681 1.8564  0.02397        0.02565
b[4]    45.465 1.7174  0.02217        0.02340
```

```
b[5]    49.600 1.6118  0.02081        0.02748
b[6]    54.701 1.4659  0.01892        0.02119
b[7]    44.352 1.4855  0.01918        0.02208
b[8]    46.988 1.7867  0.02307        0.02585
b[9]    52.026 2.2413  0.02894        0.03982
b[10]   57.900 1.7994  0.02323        0.02621
b[11]   60.799 1.7471  0.02255        0.02897
b[12]   60.989 1.8753  0.02421        0.02744
sigma    3.014 0.2537  0.00328        0.00357
sigma.a  0.162 0.0965  0.00125        0.00247
sigma.b  8.927 2.3845  0.03078        0.03360

2. Quantiles for each variable:

           2.5%     25%    50%    75%  97.5%
a[1]    -1.5425 -1.2505 -1.123 -1.019 -0.878
a[2]    -1.3658 -1.1533 -1.059 -0.980 -0.843
a[3]    -1.2242 -1.0631 -0.986 -0.909 -0.735
a[4]    -1.2205 -1.0344 -0.949 -0.849 -0.578
a[5]    -1.1003 -0.9670 -0.885 -0.785 -0.596
a[6]    -1.1263 -1.0003 -0.933 -0.855 -0.687
a[7]    -1.3317 -1.1514 -1.066 -0.992 -0.868
a[8]    -1.3620 -1.1532 -1.060 -0.978 -0.840
a[9]    -1.4269 -1.1950 -1.090 -1.003 -0.873
a[10]   -1.1610 -1.0117 -0.936 -0.842 -0.626
a[11]   -1.1467 -0.9939 -0.914 -0.818 -0.579
a[12]   -1.3319 -1.1084 -1.019 -0.938 -0.750
b[1]    41.9590 44.2704 45.617 47.018 50.295
b[2]    52.7540 55.0952 56.349 57.650 60.357
b[3]    31.9064 34.4725 35.680 36.916 39.334
b[4]    41.9816 44.3632 45.488 46.600 48.744
b[5]    46.3509 48.5515 49.650 50.660 52.725
b[6]    51.6616 53.7345 54.735 55.703 57.400
b[7]    41.5586 43.3322 44.293 45.345 47.399
b[8]    43.5519 45.7970 46.919 48.113 50.639
b[9]    48.0383 50.4926 51.837 53.438 56.876
b[10]   54.1135 56.7529 57.974 59.116 61.225
b[11]   57.0718 59.7036 60.889 61.995 63.966
b[12]   57.3885 59.7313 60.959 62.199 64.739
sigma    2.5637  2.8365  2.997  3.175  3.559
sigma.a  0.0143  0.0908  0.152  0.219  0.379
sigma.b  5.6046  7.3150  8.475 10.113 14.680
```

– INLA

```
> require(INLA)
```

Como no exemplo anterior, precisamos definir priori Uniforme para o desvio padrão do intercepto e para a inclinação (aleatórios).

```
> unif.prior = "expression:
+ a = 0;
+ b = 100;
+ sigma = sqrt(exp(-log_precision));
+ return(sigma/(b-a));"
> h.u <- list(theta=list(prior=unif.prior))
> q4.df$g.a <- q4.df$g
> form4.inla <- y ~ x + f(g.a, model='iid', hyper=h.u) +
+     f(g, x, model='iid', hyper=h.u)
> q4.inla <- inla(form4.inla, data=q4.df)
> q4.inla$summary.fix
              mean     sd 0.025quant 0.5quant 0.975quant   mode       kld
(Intercept) 50.903 2.5532     45.797   50.906    55.9872 50.913 1.214e-10
x           -1.005 0.0738     -1.154   -1.004    -0.8602 -1.002 8.559e-11
> sigmas.post <- lapply(q4.inla$marginals.hy, function(m)
+                       inla.tmarginal(function(x) sqrt(1/x), m))
> sapply(sigmas.post, inla.zmarginal, silent=TRUE)
```

	Precision for the Gaussian observations	Precision for g.a	Precision for g
mean	2.931	8.385	0.1038
sd	0.2388	2.039	0.03319
quant0.025	2.511	5.235	0.04026
quant0.25	2.76	6.914	0.07971
quant0.5	2.911	8.075	0.1066
quant0.75	3.082	9.531	0.1249
quant0.975	3.448	13.21	0.1671