Package 'lmeNBBayes'

February 21, 2015

Type Package

Title Compute the Personalized Activity Index Based on a Flexible Bayesian Negative Binomial Model

Version 1.3.1

Date 2013-12-13

Author Yumi Kondo

Maintainer Yumi Kondo <y.kondo@stat.ubc.ca>

Description The functions in this package implement the safety monitoring procedures proposed in the paper titled ``A flexible mixed effect negative binomial regression model for detecting unusual increases in MRI lesion counts in individual multiple sclerosis patients" by Kondo, Y., Zhao, Y. and Petkau, A.J. The procedure first models longitudinally collected count variables with a negative binomial mixed-effect regression model. To account for the correlation among repeated measures from the same patient, the model has subject-specific random intercept, which is modelled with the infinite mixture of Beta distributions, very flexible distribution that theoretically allows any form. The package also has the option of a single beta distribution for random effects. These mixed-effect models could be useful beyond the application of the safety monitoring. The inference is based on MCMC samples and this package contains a Gibbs sampler to sample from the posterior distribution of the negative binomial mixed-effect regression model. Based on the fitted model, the personalized activity index is computed for each patient. Lastly, this package is companion to R package lmeNB, which contains the functions to compute the Personalized Activity Index in the frequentist framework.

License GPL (>= 2)

Depends

LinkingTo

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-02-21 01:31:16

R topics documented:

`dqmix` *Estimate the random effect distribution of the flexible mixed effect negative binomial regression.*

Description

Given the output of the `lmeNBBayes`, this function estimate the posterior density of the random effect at grids.

Usage

```
dqmix(weightH1, aGs, rGs, alphas = seq(0, 0.99, 0.01), dens = TRUE)
```

Arguments

`weightH1`	A B by M matrix, containing the probability components of the infinite mixture of beta distribution. The output of `lmeNBBayes`.
`aGs`	A B by M matrix, containing the shape1 parameters of the infinite mixture of beta distribution. The output of `lmeNBBayes`.
`rGs`	A B by M matrix, containing the shape2 parameters of the infinite mixture of beta distribution. The output of `lmeNBBayes`.
`alphas`	The grids of points at which the density is evaluated. Must be the points within 0 and 1.
`dens`	If TRUE, the density estimate of the random effect is return, if FALSE then the quantiles of the random effects are estimated.

Author(s)

Kondo, Y.

References

Kondo, Y., Zhao, Y. and Petkau, A.J., "A flexible mixed effect negative binomial regression model for detecting unusual increases in MRI lesion counts in individual multiple sclerosis patients".

See Also

`lmeNBBayes` `getDIC`

Examples

```
## See the examples of function nbinDP
```

---

getDIC *Compute the DIC given the output from* lmeNBBayes.

Description

If partially marginalized posterior distribution (i.e. Reduce=1 in the computation of lmeNBBayes) is a target distribution, the DIC is computed using the focused likelihood

$Pr(\boldsymbol{Y}_i = \boldsymbol{y}_i | \{a_{G_h}, r_{G_h}\}_{h=1}^{M}, \pi_h\}_{h=1}^{\infty}, \boldsymbol{\beta})$.

If not, then the DIC is computed using the focused likelihood $Pr(\boldsymbol{Y}_i | a_{G_{h_i}}, r_{G_{h_i}}, \boldsymbol{\beta})$.

Usage

```
getDIC(olmeNBB, data,
       ID, useSample=NULL,focus = c("FG","G","aGh.rGh","para"),
       lower.alpha=0.0001,upper.alpha=0.99999,inc.alpha=0.0005)
llk.FG_i(ys,rs,aGs,bGs,ps)
```

Arguments

olmeNBB	The output of lmeNBBayes.
data	See lmeNBBayes.
ID	See the description in lmeNBBayes.
useSample	A vector of length the total number of repeated measures (i.e. the same as the length of Y), containing TRUE or FALSE, indicating which samples to be kept or discarded.
focus	Focused likelihood
lower.alpha	Used only when focus is FG. See details.
upper.alpha	Used only when focus is FG. See details.
inc.alpha	Used only when focus is FG. See details.
ys	A vector containing the response values
rs	A vector containing the size parameters of negative binomial. The length of rs must be the same as the length of ys.
aGs	A vector containing the shape1 parameters of the approximated infinite mixture of betas.
bGs	A vector containing the shape2 parameters of the approximated infinite mixture of betas. The length must be the same as the length of bGs.
ps	A vector containing the probability parameters of the approximated infinite mixture of betas. The length must be the same as the length of aGs.

Details

Denote P be a vector of "focused" parameters.

Using Spiegelhalter et. al.(2002)'s notation, the effective number of parmeters can be computed as:

$p_D = \bar{D} - D(\bar{P})$

where D is the deviance and the $\bar{P}$ is the expectation of P.

When `focus = FG` then the focused parameters, denoted as P, are the random effect distribution (i.e.,infinite mixture of beta distribution) and the regression coefficients. In the computation, the expected regression coefficients are obtained by simply computing the mean of the posterior samples of coefficients. The expected infinite mixture of beta distribution is obtained in the following steps:

STEP 1: Provide a fine grids of points between [0,1]. We chose the grid of points to be

`alphas <- seq(lower.alpha,upper.alpha,inc.alpha)`.

STEP 2: For each sampled infinite mixture of betas, Evaluate its value at every grid provided from STEP 1 for each sample. Obtain `B` by `length(alphas)` matrix.

STEP 3: Given the matrix from STEP2, at each grid of points, we compute the average value of density. Obtain a vector of length `length(alpha)`, that contains the estimated expected random effect density at fine grid of points.

STEP 4: Given the estimated expected coefficients and the estimated expected random effect density, evaluate $D(\bar{P})$ by integrating the conditional likelihood given random effects with respect to the estimated expected random effect density from STEP 3.

Author(s)

Kondo, Y.

References

Kondo, Y., Zhao, Y. and Petkau, A.J., "A flexible mixed effect negative binomial regression model for detecting unusual increases in MRI lesion counts in individual multiple sclerosis patients".

Spiegelhalter, D.J.; Best, N. G.; Carlin, B.P., van der Linde, A. (2002). "Bayesian measures of model complexity and fit (with discussion)". Journal of the Royal Statistical Society, Series B 64 (4): 583-639.

See Also

`lmeNBBayes`

`dqmix` `index.batch.Bayes`

Examples

```
## See the examples of function lmeNBBayes
```

`getS.StatInMed` *Generate samples from the flexible mixed-effect negative binomial distribution*

Description

This function yields samples from the simulation models specified in the paper by Kondo Y et al.

Usage

```
getS.StatInMed(iseed = "random", rev = 4, dist = "b",
               mod = 0, probs = seq(0, 0.99, 0.01),
               ts = seq(0.001, 0.99, 0.001), trueCPI = FALSE,
               full = FALSE, Scenario = "SPMS")
```

Arguments

`iseed`	Necessary only when `mod` = 0. Integers to specify a seed. If iseed="random", seed is not specified.
`rev`	Necessary only when `mod` = 0. At which DSMB reviews, data is generated.
`dist`	Necessary only when `mod` = 0. `dist` must be either `"b"` `"b2"` or `"YZ"`. If `dist="b"` then random effect G_i is from a single beta, if `dist="b2"` then it is from a mixture of two betas and if `dist="YZ"` then it is transformed to range $[0, Inf)$ and from a mixture of normal and gamma. See `details` for more details.
`mod`	If `mod` = 0 then `getS.StatInMed` generates a simulation sample. If `mod` = 1 then `getS.StatInMed` returns true quantiles of G_i at given `probs`. If `mod` = 2 then `getS.StatInMed` returns true densities of G_i at grids of points specified at `ts`. If `mod` = 3 then `getS.StatInMed` returns parameters of the simulation model.
`probs`	Necessary only when `mod` = 1. `probs` can be a vector of probabilities.
`ts`	Necessary only when `mod` = 3. `ts` can be a grid of points in (0,1).
`trueCPI`	Necessary only when `mod` = 0. If `trueCPI=TRUE`, `getS.StatInMed` returns the true conditional probability indices of N patient, computed using separate mean functions for the control and treated groups.
`full`	Necessary only when `mod` = 0. If `full=TRUE`, `rev` is ignored and `getS.StatInMed` returns complete dataset that contains 180 patients and all of them have 10 repeated measures.
`Scenario`	Necessary only when `mod` = 0. If `Scenario` specifies the prior for $\boldsymbol{\beta}$, and it must be either `"full"` or `"SPMS"`. See `details`.

Details

Simulation settings are as follows.

Given the covariate vectors $\boldsymbol{X}_{ij}$ for mean counts, response counts Y_{ij} of the jth repeated measure of ith patient are assumed to be from the mixed-effect negative binomial model:

$Y_{ij}|G_i = g_i, \boldsymbol{\beta}\ i.i.d. \sim NB(Y_{ij}; \text{size} = \exp(\boldsymbol{X}_{ij}^T\boldsymbol{\beta}), \text{prob} = g_i)$.

This formulation results in $\log E(Y_{i,j}) = \log(\mu_{\frac{1}{G}} - 1) + \boldsymbol{X}_{ij}^T\boldsymbol{\beta}$. The mean count is modeled on the log scale as a constant over every four-month follow-up period, where the constants are allowed to dependent on the treatment assignment A_i (A_i = 1 for treatment, else 0).

$\log E(Y_{i,j}|A_i = a_i) = \log(\mu_{\frac{1}{G}} - 1) + \alpha_0 + \sum_{a=0}^{1}\sum_{t=1}^{2}\beta_{a,t}I(j \in \boldsymbol{T}_t, a_i = a)$,

where $\mu_{\frac{1}{G}} = E(\frac{1}{G_i})$ and $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$ respectively contains indices corresponding to scans taken within the 1st and 2nd four-month interval during the follow-up.

The regression coefficients $\boldsymbol{\beta} = (\alpha_0, \beta_{0,1}, \beta_{1,1}, \beta_{0,2}, \beta_{1,2})$ are assumed to differ among studies and are generated from a multivariate normal distribution with $\boldsymbol{\mu}$ and Σ replaced by the estimates from the full informative prior (Scenario A) or the SPMC informative prior (Scenario B) developed in Section 5.3 of the referred paper. To see their values set `mod=3`. (See example below). The proportion of treated patients is assumed to be 0.67.

`getS.StatInMed` allows three random effect model:

Setting 1: $G_i \sim Beta(3, 0.8)$ which returns $(E(Y_{ij}, SD(Y_{ij}))) = (1.48, 3.45)$ and $(1.40, 3.29)$ at baseline under full and SPMC Scenarios respectively.

Setting 2: $G_i \sim 0.3Beta(10, 10)+0.7Beta(20, 1)$ which returns $(E(Y_{ij}, SD(Y_{ij}))) = (4.12, 3.73)$ and $(0.20, 0.51)$ at baseline for the patients whose REs are generated from the first and the second component of the mixture under full Scenario, and $(2.90, 3.59)$ and $(0.18, 0.49)$ under SPMS Scenario.

Setting 3: $G_i = \frac{1}{G_i^*+1}$ where $G_i^* \sim 0.85Gamma(0.176, 2.226) + 0.15N(1.820, 0.303)$ where $Gamma(a, b)$ represents the gamma CDF with variance ab^2. This mixture returns $(E(Y_{ij}, SD(Y_{ij}))) = (1.46, 4.16)$ and $(6.75, 4.56)$ at baseline for the patients whose REs are drawn from the first and second component of the mixture under full Scenario, and $(1.38, 3.97)$ and $(6.39, 4.41)$ under SPMS Scenario.

The choices of the random effect settings can be controlled by the input `dist`.

`getS.StatInMed` assumes MRI scans are taken monthly with a total of 10 MRI scans for each patients, a screening a baseline and eight follow-up scans. We assume that 15 patients are recruited every month so that 180 patients are recruited in 12 months, leading to a study duration of 21 month. DSMB reviews are assumed to occur every 4 months. This means that by the DSMB review 1, 150 scans are available from 60 patients (15 patients have 4 scans each, next 15 patients have 3 scans each, next 15 patients have 2 scans each and the last 15 patient have 1 scan each.) By the DSMB review 2, 540 scans are available from 120 patients (15 patients have 8 scans each, the next 15 patients have 7 scans each,....the last 15 patients have only 1 scan each.) The DSMB reviews can be specified by the input `rev`.

Value

When `mod=0`, the it returns a dataframe that contains:

`Y`: the generated response counts

`Intercept`: all 1

`timeInt1`: 1 if the row corresponds to the count taken at the 1st 4-month followup interval, else 0.

`timeInt2`: 1 if the row corresponds to the count taken at the 2nd 4-month followup interval, else 0.

`ID`: the patient ID.

`gs`: generated random effect G_i.

`scan`: $-1, 0, 1, 2, \cdots$. -1 indicates the screening scan, 0 indicates the baseline and 1,2,.. indicates 1,2-th followup scans.

`days`: `scan` + 2

`hs`: indicates which component the random effect is generated. If `dist="b"`, `hs` is all 1. If `dist="b2"` or `dist="YZ"` then `hs` is either 1 or 2.

`trtAss`: indicates treatment assignments.

`labelnp`: 1 if scans correspond to new scans else 0.

`betPlcb`: The only first three elements are relevant. They are the generated intercept and time effects of placebo patients for mean counts. i.e., $\alpha_0, \beta_{0,1}, \beta_{0,2}$.

`betFull`: The only first five elements are relevant. They are the generated intercept and time effects for mean counts. i.e., $\alpha_0, \beta_{0,1}, \beta_{1,1}, \beta_{0,2}, \beta_{1,2}$.

`probIndex`: The only first N elements are relevant. They are the true conditional probability indices of N patient, computed using true mean functions of the control patients (for both control and treated patients).

`probIndexTRUE`: This appear only if `trueCPI=TRUE`. The only first N elements are relevant. They are the true conditional probability indices of N patient, computed using separate mean functions for the control and treated groups.

Author(s)

Kondo, Y.

References

Kondo, Y., Zhao, Y. and Petkau, A.J., "A flexible mixed effect negative binomial regression model for detecting unusual increases in MRI lesion counts in individual multiple sclerosis patients".

See Also

`lmeNBBayes` `getS.StatInMed` `getDIC` `dqmix` `index.batch.Bayes`

Examples

```
## Not run:

## See the full informative prior for beta
temp <- getS.StatInMed(mod=3,Scenario="full")
temp$mu_beta
temp$Sigma_beta
```

```
## See the SPMS informative prior for beta
temp <- getS.StatInMed(mod=3,Scenario="SPMS")
temp$mu_beta
temp$Sigma_beta

## See also the examples in lmeNBBayes

## End(Not run)
```

`index.batch.Bayes` *The main function to compute the point estimates and 95% credible intervals of the conditional probabilities* $Pr(Y\_i, new+ \geq y\_i, new+ | \boldsymbol{Y}\_i, pre = \boldsymbol{y}\_i, pre)$ *for multiple subjects.*

Description

Let m_i be the number of pre-measurements and n_i be the total number of repeated measures. Then the repeated measure of a subject can be divided into a pre-measurement set and a new measurement set as $\boldsymbol{Y}_i = (\boldsymbol{Y}_{i,pre}, \boldsymbol{Y}_{i,new})$, where $\boldsymbol{Y}_{i,pre} = (Y_{i,1}, \cdots, Y_{i,m_i})$ and $\boldsymbol{Y}_{i,new} = (Y_{i,m_i+1}, \cdots, Y_{i,n_i})$. Given an output of `lmeNBBayes`, this function computes the probability of observing the response counts as large as those new observations of subject i, $\boldsymbol{y}_{i,new}$ conditional on the subject's previous observations $\boldsymbol{y}_{i,pre}$ for subject i. That is, this function returns a point estimate and its asymptotic 95% confidence interval (for a parametric model) of the conditional probability for each subject: $Pr(Y_{i,new+} \geq y_{i,new+} | \boldsymbol{Y}_{i,pre} = \boldsymbol{y}_{i,pre})$, where $Y_{i,new+} = \sum_{j=m_i+1}^{n_i} Y_{ij}$.

Usage

```
index.batch.Bayes(data,labelnp,ID,olmeNBB,thin=NULL,printFreq=10^5,unExpIncrease=TRUE)
```

Arguments

`data`	See `lmeNBBayes`. This `data` does not have to be the same as the one used in the computations of negative binomial mixed effect regression (`lmeNBBayes`).
`labelnp`	See `lmeNBBayes`. `nrow(data) == length(labelnp)` must be satisfied.
`ID`	See the description in `lmeNBBayes`. `nrow(data) == length(labelnp)` must be satisfied.
`olmeNBB`	The output of the function `lmeNBBayes`.
`thin`	The frequency of thinning
`printFreq`	See the description in `lmeNBBayes`.
`unExpIncrease`	Internal use only. Should be always TRUE.

Value

`condProb` `(olmeNBB$para$B-olmeNBB$para$burnin)/thin` by the number of patients, N (=`length(unique(ID))`), matrix, containing the MCMC samples of the conditional probability index for each patient at every selected iteration (after discarding burn-in and thinning). If some patients have 0 pre-scans or 0 new-scans, then `NA` is returned.

`condProbSummary`

4 by N matrix. The first column contains the posterior estimates of the conditional probability index. The second column contains the posterior SE. The third column contains the lower bound of the 95% credible interval. The fourth column contains the upper bound of the 95% credible interval.

Author(s)

Kondo, Y.

References

Kondo, Y., Zhao, Y. and Petkau, A.J., "A flexible mixed effect negative binomial regression model for detecting unusual increases in MRI lesion counts in individual multiple sclerosis patients".

See Also

`lmeNBBayes` `getDIC` `dqmix`

Examples

```
## See the examples of function lmeNBBayes
```

`lmeNBBayes` *Generate posterior samples from a flexible mixed effect negative binomial regression model.*

Description

Let Y_{ij} be the response count at jth repeated measure from the ith patient ($i = 1, \cdots, N$ and $j = 1, \cdots, n_i$). The negative binomial mixed-effect independent model assumes that given the random effect $G_i = g_i$, the count response from the same subjects i.e., Y_{ij} and $Y_{ij'}$ are conditionally independent and follow the negative binomial distribution:

$Y_{ij}|G_i = g_i, \boldsymbol{\beta}\ i.i.d. \sim NB(Y_{ij}; \text{size} = \exp(\boldsymbol{X}_{ij}^T\boldsymbol{\beta}), \text{prob} = g_i)$

where $\boldsymbol{X}_{ij}$ is the covariates for mean counts. This formulation results in $\log E(Y_{i,j}) = \log(\mu_{\frac{1}{G}} - 1) + \boldsymbol{X}_{ij}^T\boldsymbol{\beta}$. To allow flexible form of a random effect distribution, we assume that the patient-specific random effect is assumed to be from Dirichlet process mixture of beta distributions. This essentially means that random effect G_i is from an infinite mixture of Beta distributions:

$G_i|\{a_{G_h} r_{G_h}, \pi_h\}_{h=1}^{\infty} \sim \sum_{h=1}^{\infty} \pi_h Beta(G_i; \text{shape1} = a_{G_h}, \text{shape2} = r_{G_h})$,

where π_h is modelled with the stick-breaking prior. Introducing latent variable $V_h, h = 1, 2, \cdots$, this prior is defined as $\pi_1 = V_1$ and $pi_h = V_h \prod_{l<h}(1 - V_h)$ for $h > 1$ V_h $i.i.d.$ $\sim Beta(1, D)$.

The rest of priors are specified as: $\boldsymbol{\beta} \sim N(\boldsymbol{\mu}, \Sigma)$,

$(a_G, r_G) \sim Unif(a_G; \text{min} = 0.5, \text{max} = max_{a_G}) Unif(r_G; \text{min} = 0.5, \text{max} = max_{r_G})$,

$D \sim Unif(v; \text{min} = a_D, \text{max} = ib_D)$.

The default values of the hyperparameters are $\mu_{\boldsymbol{\beta}}$ = `rep(0,p)`, $\Sigma_{\boldsymbol{\beta}}$ = `diag(5,p)`, max_{a_G} = `30`, a_D = `0.01` and ib_D = `3`. These selections of hyperparameters could be used as uninformative ones.

The function `lmeNBBayes` also allows generating posterior samples from the parametric version of the model above which simply assumes that the random effect is from the single beta distribution. (The rest of the prior specifications are the same).

Usage

```
lmeNBBayes(formula,data, ID, B = 105000, burnin = 5000,
           printFreq = B, M = NULL, probIndex = FALSE,
           thin =1,labelnp=NULL, epsilonM = 1e-4,
   para = list(mu_beta = NULL,Sigma_beta = NULL,
           max_aG=30,mu_lnD=NULL,sd_lnD=NULL),
           DP=TRUE,thinned.sample=FALSE, proposalSD = NULL)
```

Arguments

`formula`	An object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The formula must contain an intercept term.
`data`	A data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. The each row must contains the data corresponding to the repeated measure j of subjects and the rows (i, j)s must be ordered in a way that measurements from a subject is clustered together as $(1, 1)$,...,$(1, n_1)$,$(2, 1)$,...,$(2, n_2)$,...,(N, n_N).
`ID`	A vector of length $\sum_{i=1}^{N} n_i$, containing the patient IDs that corresponds to `data`. i.e., `c(rep(ID_1,n_1),rep(ID_2,n_2),...,rep(ID_N,n_N))`. The length must be the same as the number of rows of `data`. Missing ID values are NOT accepted.
`B`	A scalar, the number of McMC iterations.
`burnin`	A scalar for a burn-in period. The proposal variance of the Metoropolice-Hasting rate is adjusted during the burn-in preiod.
`printFreq`	An integer value, indicating the frequency of iterations to print during the McMC run.
`M`	Necessary only if `DP=1`. Our Gibbs sampler approximates the infinite mixture of beta distributions by truncating it with `M` components by setting $V_M = 1$ so that $pi_M = 1 - \sum_{h=1}^{M-1} \pi_h$. If `M` is NULL, `M` is selected so that the amount of probability assigned to the final mass point is expected to be `epsilonM`. i.e., $E(\pi_M) = E\{(\pi_M \mid D)\} = E(\{1 - 1/(D + 1)\}^{M-1}) < \epsilon$.

probIndex	Logical, if it is TRUE then the conditional probability index is computed for each patient at every thinning after discarding burn-in.
thin	Thinning frequency. Necessary if probIndex is TRUE.
labelnp	A vector of length $\sum_{i=1}^{N} n_i$, containing 0 or 1. Zero indicates that the corresponding repeated measure should be treated as pre-scan and 1 indicates that it is a new scan. labelnp is necessary only if probIndex is TRUE.
epsilonM	A scalar. See the description of M.
para	A list containing hyperparameter values. If DP=0 then the followings must be specified: mu_beta (a vector of length p), Sigma_beta (a p by p covariance matrix) and max_aG (a positive scaler). If DP=1 then in addition to the above parameters,mu_lnD (positive scaler) sd_lnD (positive scaler) must be specified. If some of these are not specified then the default values discussed in description are used.
DP	If DP=1 then the flexible mixed effect negative binomial regression is fit to the dataset. If DP=0 then the random effect distribution is assumed to be a single beta distribution.
thinned.sample	Logical. If true then return only the thinned samples, else returns the entire MCMC sample of size B.
proposalSD	List object containing two list objects min and max, which contain minimum and maximum values of the proposal standard deviations. If DP=0 then a list object min (max) must contains 3 elements corresponding to minimum (maximum) values of the proposal standard deviation of aG, rG and beta. See details for beta. If DP=1 then a list object min (max) must contains 4 elements corresponding to minimum (maximum) values of the proposal standard deviation of aG, rG, beta and ln D. See details for beta.

Details

For the parameters with non-conjugate priors $\boldsymbol{\beta}, D, a_G, b_G$, the Metropolis Hasting (MH) algorithm is employed to sample from their full conditional distributions. For D, a_G, b_G, the MH algorithm is performed separately with a normal proposal distribution, where its proposal variance is tuned during the burn-in to have the acceptance rates range between 0.2 and 0.6. One can adjust the minimum and maximum of the proposal sd via the proposalSD arguments. For each element of $\boldsymbol{\beta}$, we found that updating each regression coefficient with separate MH algorithm resulted in poor mixing in the Markov chain when high correlation is assumed in some of $\boldsymbol{\beta}$ in the prior. Therefore, the MH algorithm is performed simultaneously for all $\boldsymbol{\beta}$ and a MVN proposal distribution is employed with $a\Sigma$ as its proposal covariance matrix, where Σ is the covariance of a prior for $\boldsymbol{\beta}$ and a is a tuning scaler adjusted during the burn-in period to have the acceptance rates range between 0.2 and 0.6.

Author(s)

Kondo, Y.

References

Kondo, Y., Zhao, Y. and Petkau, A.J., "A flexible mixed effect negative binomial regression model for detecting unusual increases in MRI lesion counts in individual multiple sclerosis patients".

See Also

getDIC dqmix index.batch.Bayes

Examples

```
## Not run:

## generate samples from DSMSB review 2
d <- getS.StatInMed(rev=2,iseed=1,dist="YZ",Scenario="full")
formula.fit <- Y ~ timeInt1:trtAss + timeInt2:trtAss

B <- 10000
burnin <- 1000
thin <- 2
fit <- lmeNBBayes(formula=formula.fit,data=d, ID=d$ID,
                  B = B, burnin = burnin,  thin=thin)
## The output can be printed out:
fit


## Now, compute the conditional probability index using the mean function of placebo patients.
## We need to modify two things in output of lmeNBBayes.
## 1st, change the formula so that it does not distinguish between treatment and placebo
fit$para$formula <- Y ~ timeInt1 + timeInt2
## 2nd, disregard the coefficient that corresponds to the treated patients
fit$beta <- fit$beta[,-c(3,5)]
cpi <- index.batch.Bayes(data=d,labelnp=d$labelnp,ID=d$ID,
                         olmeNBB=fit,printFreq=10^7)
cpi

## finally access the accuracy of the CPI estimates in terms of RMSE
Npat <- length(unique(d$ID))
est <- cpi$condProbSummary[,1]
true <- d$probIndex[1:Npat]
sqrt( mean( ( est - true )^2 ,na.rm=TRUE) )



## End(Not run)
```

Index