Package 'bmeta'

January 8, 2016

Type Package

Title Bayesian Meta-Analysis and Meta-Regression

Version 0.1.2

Date 2016-01-08

Author Tao Ding, Gianluca Baio

Maintainer Gianluca Baio <gianluca@stats.ucl.ac.uk>

Depends R2jags,forestplot

Description Provides a collection of functions for conducting meta-analyses under Bayesian context in R. The package includes functions for computing various effect size or outcome measures (e.g. odds ratios, mean difference and incidence rate ratio) for different types of data based on MCMC simulations. Users are allowed to fit fixed- and random-effects models with different priors to the data. Meta-regression can be carried out if effects of additional covariates are observed. Furthermore, the package provides functions for creating posterior distribution plots and forest plot to display main model output. Trace-plots and some other diagnostic plots are also available for assessing model fit and performance.

License GPL (>= 2)

URL http://www.statistica.it/gianluca/bmeta,
http://www.statistica.it/gianluca

NeedsCompilation no

Repository CRAN

Date/Publication 2016-01-08 10:53:20

R topics documented:

`bmeta-package` *bmeta: A Bayesian Meta-Analysis Package for R*

Description

The bmeta package provides a collection of functions for conducting meta-analyses under Bayesian context in R. The package includes functions for computing various effect size or outcome measures (e.g. odds ratios, mean difference and incidence rate ratio) for different types of data based on MCMC simulations. Users are allowed to fit fixed- and random-effects models with different priors to the data. Meta-regression can be carried out if effects of additional covariates are observed. Furthermore, the package provides functions for creating posterior distribution plots and forest plot to display main model output. Traceplots and some other diagnostic plots are also available for assessing model fit and performance.

Details

Package:	bmeta
Type:	Package
Version:	0.1.2
Date:	2016-01-08
License:	GPL2
LazyLoad:	yes

Bayesian meta-analysis is becoming more frequently accepted as a statistical approach for evidence synthesis from multiple studies in health research. The Bayesian methods differ inherently from frequentist ones by assuming that model parameters are random quantities. Therefore, prior distributions for model parameters can be specified, which are normally based on external evidence. The bmeta function provides 22 models with commonly used priors for fitting different types of data (i.e. binary, continuous and count data).

Author(s)

Tao Ding, Gianluca Baio

Maintainer: Gianluca Baio <gianluca@stats.ucl.ac.uk>

References

Alex J Sutton and Keith R Abrams.(2001).Bayesian methods in meta-analysis and evidence synthesis. Statistical Methods in Medical Research,10,277-303.

Welton,N.J., Sutton,A.J., Cooper,N., Abrams,K.R.& Ades,A.E.(2012) Evidence synthesis for decision making in healthcare. Chichester, UK: John Wiley & Sons, Ltd.

`acf.plot` *Autocorrelation function plot*

Description

Function to create autocorrelation function plot to assess convergence

Usage

```
acf.plot(x,node,title="Autocorrelation function")
```

Arguments

`x`	a `bmeta` object with results of the model
`node`	variable to be displayed on the plot
`title`	title of the plot, if specified

Value

A plot showing the autocorrelation for the selected node

Author(s)

Tao Ding Gianluca Baio

Examples

```
### Read and format the data (binary)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-bin.csv"))

### List data for binary outcome
data.list <- list(y0=data$y0,y1=data$y1,n0=data$n0,n1=data$n1)

### generate output from bmeta
x <- bmeta(data=data.list,outcome="bin",model="std.dt",type="ran")

### generate autocorrelation function plot
acf.plot(x,"alpha[1]")

### generate autocorrelation function plot and specify the title
acf.plot(x,"alpha[1]",title="Autocorrelation plot")
```

bmeta *Bayesian Meta Analysis/Meta-regression*

Description

Function to fit the Bayesian fixed- and random-effects meta-analytic models with or without moderators. Models are designed to include non-informative priors.

Usage

```
bmeta(data, outcome = c("bin", "ctns", "count"), model = c("std.norm",
"std.dt", "reg.norm", "reg.dt", "std.ta", "std.mv", "reg.ta", "reg.mv", "std",
"std.unif", "std.hc", "reg", "reg.unif", "reg.hc"), type = c("fix", "ran"),
n.iter = 10000, n.burnin = 5000, n.samples = 1000, n.chains = 2,
model.file = "model.txt")

## Default S3 method:
bmeta(data, outcome = c("bin", "ctns", "count"), model = c("std.norm",
"std.dt", "reg.norm", "reg.dt", "std.ta", "std.mv", "reg.ta", "reg.mv", "std",
"std.unif", "std.hc", "reg", "reg.unif", "reg.hc"), type = c("fix", "ran"),
n.iter = 10000, n.burnin = 5000, n.samples = 1000, n.chains = 2,
model.file = "model.txt")
```

Arguments

`data`	a data list containing information on observed data (including moderators). See 'details'.
`outcome`	type of outcome that needs to be specified. For binary, continuous and count data, `bin`, `ctns` and `count` need to be specified, respectively.
`model`	type of model that needs to be specified. See 'details'.
`type`	model type—either fixed-effects (`fix`) or random-effects model(`ran`) needs to be specified.
`n.iter`	number of iterations to be used in the simulation (default is 10000)
`n.burnin`	number of burn-in to be used in the simulation (default is 5000)
`n.samples`	The total number of MCMC simulations saved (including thinning). Default at 1000
`n.chains`	number of Markov chains to be used in the simulation (default is 2)
`model.file`	Name of the text file to which the model is saved

Details

Specifying the data

The function can be used to evaluate odds ratios (or log odds ratios), mean difference and incidence rate ratios (or log incidence rate ratios). Users need to specify a list of data to be used in the function.

For binary data, events out of case and control arm and sample size of case and control arm need to be listed. For continuous data, mean and standard errors of case and control arm need to be listed if information is available. However, if only mean difference and variance can be retrieved from each study, users need to list mean difference and precision (inverse of variance). Notice that information of all the studies need to be provided in the same format for the function to work properly. For example, the function cannot work if some of the studies provide mean and standard errors of the two arms while the rest studies provide mean difference and variance. For count data, total number of events in the follow-up period of case and control arm, total follow-up person-time in case and control arm should be listed.

If additional impacts of a variable or more than one variable are observed (when meta-regression is expected to be used), users need to provide a matrix with each column either containing a dummy variable or a continuous variable. In case that categorical variables (i.e. ethnicity, age band) are observed and included, users need to first choose a 'baseline' category as reference and then create dummies for each of the rest categories.

Model selection

Apart from 'null' models which apply Bayesian methods to obtain study-specific without pooling-effects, there are 22 models included in this package for pooling study-specific estimates together and producing summary estimate. The number of models designed for binary, continuous and count data are 8, 8 and 6, respectively. The model selection process for binary and count data requires users to specify not only whether meta-analysis or meta-regression is wanted but also the priors to be used.

For binary data, normal and Student t-distribution priors for summary estimates (on log scale) can be selected and it is indicated that Student t-distribution has heavier tails and is therefore more robust to outliers. The argument 'model' here includes 4 options — `std.norm`, `std.dt`, `reg.norm`, `reg.dt`.

For continuous data, rather than specifying prior, users need to select whether all studies included report mean and standard errors of two arms separately or only mean difference and variance as discussed above in the 'Specifying the data' section. The argument 'model' here includes 4 options— `std.ta`, `std.mv`, `reg.ta`, `reg.mv` ('model' ending with 'ta' represents 'two arms' and ending with 'mv' represents 'mean and variance').

For count data, uniform and half-Cauchy distribution priors for the variability of summary estimates (on log scale) can be selected. It is suggested that half-Cauchy distribution has heavier tails and allows for outliers and accommodates small variances closing to zero. It should be noticed that there is no need to specify priors for fixed-effects models for count data. The argument 'model' here includes 6 options — `std`, `std.unif`, `std.hc`, `reg`, `reg.unif`, `reg.hc`.

In conjunction with the argument 'type'— `fix` or `ran`, users can select the specific model wanted for a certain type of data.

Value

`mod`	A `rjags` object with the results of the model
`params`	a list of monitored parameters to be saved
`data`	the original dataset
`inits`	a list with `n.chains` elements, with each element itself being a list of starting values for the model or a function generating initial values

outcome	selected type of outcome (i.e. bin/ctns/count)
type	selected type of model (either fixed-/random-effects)
model	selected model with specific priors
mod0	independent model without pooling effects

Author(s)

Tao Ding Gianluca Baio

References

Baio, G.(2012) Bayesian methods in health economics. Chapman Hall, CRC.

Welton, N.J., Sutton, A.J., Cooper, N., Abrams, K.R. & Ades, A.E. (2012) Evidence synthesis for decision making in healthcare. Chichester, UK: John Wiley & Sons, Ltd.

Examples

```
### Read and format the data (binary)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-bin.csv"))

### List data for binary outcome (for meta-analysis)
d1 <- data.list <- list(y0=data$y0,y1=data$y1,n0=data$n0,n1=data$n1)

### List data for binary outcome when there is a covariate (for meta-regression)
d1 <- data.list <- list(y0=data$y0,y1=data$y1,n0=data$n0,n1=data$n1,X=cbind(data$X0))

### Select fixed-effects meta-analysis with normal prior for binary data
m1 <- bmeta(d1, outcome="bin", model="std.norm", type="fix",n.iter=100)

### Select random-effects meta-regression with t-distribution prior for binary
### data
m2 <- bmeta(data.list, outcome="bin", model="reg.dt", type="ran",n.iter=100)



### Read and format the data (continuous)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-ctns.csv"))

### List data for continuous outcome for studies reporting two arms separately
### (for meta-analysis)
d1 <- data.list <- list(y0=data$y0,y1=data$y1,se0=data$se0,se1=data$se1)

### List data for continuous outcome for studies reporting mean difference and
### variance with a covariate (for meta-regression)
d2 <- data.list2 <- list(y=data$y,prec=data$prec,X=cbind(data$X0))

### Select fixed-effects meta-analysis with studies reporting information of
### both arm for continuous data
m1 <- bmeta(data.list, outcome="ctns", model="std.ta", type="fix",n.iter=100)

### Select random-effects meta-regression with studies reporting mean difference and
```

```
### variance only for continuous data
m2 <- bmeta(data.list2, outcome="ctns", model="reg.mv", type="ran",n.iter=100)



### Read and format the data (count)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-count.csv"))

### List data for count outcome (for meta-analysis)
d1 <- data.list <- list(y0=data$y0,y1=data$y1,p0=data[,6],p1=data[,10])

### List data for count outcome when there is a covariate (for meta-regression)
d2 <- data.list <- list(y0=data$y0,y1=data$y1,p0=data[,6],p1=data[,10],X=cbind(data$X0))

### Select fixed-effects meta-analysis for count data
m1 <- bmeta(d1, outcome="count", model="std", type="fix",n.iter=100)

### Select random-effects meta-analysis with half-Cauchy prior for count data
m2 <- bmeta(d1, outcome="count", model="std.hc", type="ran",n.iter=100)

### Select random-effects meta-regression with uniform prior for count data
m3 <- bmeta(d2, outcome="count", model="reg.unif", type="ran",n.iter=100)
```

`diag.plot` *Diagnostic plot to examine model fit*

Description

Function to produce plot based on different diagnostic statistics

Usage

```
diag.plot(x,diag="Rhat")
```

Arguments

`x`	a `bmeta` object with results of the model
`diag`	diagnostic statistics to be used—either the Gelman-Rubin statistic (Rhat) by default or effective sample size (n.eff)

Value

A plot showing the relevant diagnostic stats for each node in the model

Author(s)

Tao Ding Gianluca Baio

Examples

```
### Read and format the data (binary)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-bin.csv"))

### List data for binary outcome
data.list <- list(y0=data$y0,y1=data$y1,n0=data$n0,n1=data$n1)

### generate output using bmeta
x <- bmeta(data=data.list,outcome="bin",model="std.norm",type="fix")

### run the diagnostic plot to examine the Gelman-Rubin statistic
diag.plot(x)

### run the diagnostic plot to examine the effective sample size
diag.plot(x,diag="n.eff")
```

`forest.plot` *Function to create forest plot*

Description

A function to call package `forestplot` from R library and produce forest plot using results from `bmeta`. The posterior estimate and credible interval for each study are given by a square and a horizontal line, respectively. The summary estimate is drawn as a diamond.

Usage

```
forest.plot(x,title=NULL,xlab=NULL,log=FALSE,study.label=NULL,clip=c(-3,3),
lines="black",box="blue",summary="orange",box.symb="box",label.cex=.8,
xlab.cex=1,ticks.cex=.8,...)
```

Arguments

`x`	a `bmeta` object with results of the model
`title`	title of the plot
`xlab`	title of the x-axis label
`log`	estimates on natural scale is displayed by default. If TRUE, log scale is used (i.e. log odds ratio, log incidence rate ratio). For continuous data, estimates are always presented on natural scale and users do not need to specify this argument.
`study.label`	label for each study and the summary estimate. See details.
`clip`	lower and upper limits for clipping credible intervals to arrows
`lines`	selects the colour for the lines of the intervals. If the extra option `add.null` is set to `TRUE`, then `lines` should be specified as a two-element vector. If the user fails to do so, `bmeta` will overwrite this setting and select suitable values.

box	selects the colour for mean study-specific estimates. If the extra option add.null is set to TRUE, then box should be specified as a two-element vector. If the user fails to do so, bmeta will overwrite this setting and select suitable values.
summary	selects the colour for the pooled estimate
box.symb	selects the symbol used to plot the mean. Options are "box" (default) or "circle"
label.cex	defines the size of the text for the label. Defaults at .8 of normal size
xlab.cex	defines the size of the text for the x-label. Defaults at 1 of the normal size
ticks.cex	defines the size of the text for the x-axis ticks. Defaults at .8 of the normal size
...	Additional arguments. Includes - add.null = TRUE/FALSE. If set to true, adds a plot of the null (no-pooling model) - line.margin = the distance between lines in case multiple graphs are shown on the same plot - box.size = the size of the summary box - new.page = TRUE/FALSE. If set to true, then a new graph overwrite the existing one - zero (x-axis coordinate for zero line. If you provide a vector of length 2 it will print a rectangle instead of just a line. Default at 0 or 1 depending on log scale) - legend = a legend for the multi-graph plot (including the null/no-pooling model)

Author(s)

Tao Ding Gianluca Baio

Examples

```
### Read and format the data (binary)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-bin.csv"))

### List data for binary outcome
data.list <- list(y0=data$y0,y1=data$y1,n0=data$n0,n1=data$n1)

### Select fixed-effects meta-analysis with normal prior for binary data
x <- bmeta(data.list, outcome="bin", model="std.norm", type="fix")

### Plot forest plot
forest.plot(x)

### Plot forest plot on log scale
forest.plot(x,log=TRUE)

### Select random-effects meta-analysis with t-distribution prior for binary
### data
x <- bmeta(data.list, outcome="bin", model="std.dt", type="ran")

### Plot 'two-line' forest plot showing estimates from both randome-effects
### model and no-pooling effects model for comparison
forest.plot(x,add.null=TRUE,title="Two-line forestplot for comparison")


### Read and format the data (continuous)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-ctns.csv"))
```

```
### List data for continuous outcome
data.list <- list(y0=data$y0,y1=data$y1,se0=data$se0,se1=data$se1)

### Select fix-effects meta-analysis for studies reporting two arms separately
x <- bmeta(data=data.list,outcome="ctns",model="std.ta",type="fix")

### Define for individual studies
study.label <- c(paste0(data$study,", ",data$year),"Summary estimate")

### Produce forest plot with label for each study and control the lower and upper
### limits for clipping credible intervals to arrows
forest.plot(x,study.label=study.label,clip=c(-7,4))
```

`funnel.plot` *Funnel plot to examine publication bias*

Description

Function to examine publication bias. For both fixed- and random-effects models, estimates from no-pooling effects model are used as study-specific estimates. For random-effects models, the corresponding fixed-effects models are implemented at background to obtain pooled estimate. For example, if users call `bmeta` to run random-effects meta-analysis with normal prior, fixed-effects meta-analysis with normal prior are implemented at background to obtain pooled estimate for graphing. In the absence of publication and heterogeneity, the scatter resembles a symmetrical funnel and the triangle area formed by connecting the centred summary estimate with its 2.5% and 97.5% quantiles on either side includes about 95% of the studies if the fixed-effects model assumption holds (i.e. all the studies estimate the same effect).

Usage

```
funnel.plot(x,xlab=NULL,ylab=NULL,title=NULL,xlim=NULL)
```

Arguments

`x`	a `bmeta` object with results of the model
`xlab`	title of x-axis. If unspecified, the function sets an appropriate lable by default.
`ylab`	title of x-axis. If unspecified, the function sets an appropriate lable by default.
`title`	title of the plot if specified
`xlim`	horozontal limits of the plot region. If unspecified, the function sets the horizontal plot limits to (-6,6).

Author(s)

Tao Ding Gianluca Baio

Examples

```
### Read and format the data (binary)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-bin.csv"))

### List data for binary outcome
data.list <- list(y0=data$y0,y1=data$y1,n0=data$n0,n1=data$n1)

### Select random-effects meta-analysis with t-distribution prior for binary
### data
x <- bmeta(data.list, outcome="bin", model="std.dt", type="ran")

### using output from bmeta to produce funnel plot
funnel.plot(x)

### using output from bmeta and specify title of the plot
funnel.plot(x,title="funnel plot")

### using output from bmeta and specify the limit of x-axis and title
funnel.plot(x,title="funnel plot",xlim=c(-2,1))
```

`posterior.plot` *Posterior distribution plots for summary estimates and between-study standard deviation (measurement of heterogeneity)*

Description

Function to create posterior distribution plots for summary estimates and between-study standard deviation based on output from `bmeta`

Usage

```
posterior.plot(x, xlim = NULL, xlab="", main="Posterior distribution Plot",
scale = "log",heterogeneity=FALSE)
```

Arguments

`x`	a `bmeta` object with results of the model
`xlim`	horozontal limits of the plot region. If unspecified, the function sets the horizontal plot limits to (-3,3) for binary and count data and (-5,5) for continuous data.
`xlab`	title for the x-axis
`main`	title of the plot. If unspecified, the function sets an appropriate title by default.
`scale`	logical specifying whether summary estimates need to be displayed on log ("log") or natural scale ("exp"). For continuous data, summary estimates are always displayed on natural scale, therefore, users do not need to specify this option.
`heterogeneity`	logical specifying whether to resent posterior plot for between-study standard deviation (TRUE) to examine heterogeneity of different studies. If unspecified, FALSE by default.

Author(s)

Tao Ding Gianluca Baio

References

Anzures-Cabrera,J & Higgins,J.P.T.(2010) Graphical displays for meta-analysis: An overview with suggestions for practice.Res Synth Methods,1,66-80.

Examples

```
### Read and format the data (binary)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-bin.csv"))

### List data for binary outcome
data.list <- list(y0=data$y0,y1=data$y1,n0=data$n0,n1=data$n1)

### Select random-effects meta-analysis with t-distribution prior for binary
### data
x <- bmeta(data.list, outcome="bin", model="std.dt", type="ran")

### using output from bmeta to produce posterior plot
posterior.plot(x)

### using output from bmeta and specify the horizontal limits
posterior.plot(x,xlim=c(-2,1))

### using output from bmeta on natural scale and specify more options
posterior.plot(x,xlim=c(-0.5,2.5),xlab="odds ratio",main="Posterior distribution
of pooled odds ratio", scale="exp")

### examine heterogeneity by producing posterior plot for between-study standard
### deviation
posterior.plot(x,heterogeneity=TRUE,xlim=c(0,3),xlab="between-study standard
deviation")
```

---

print.bmeta *Print method for* bmeta *objects*

Description

Function to print output from function bmeta

Usage

```
## S3 method for class 'bmeta'
print(x, ...)
```

Arguments

x	a bmeta object with results of the model
...	other arguments

Author(s)

Tao Ding Gianluca Baio

---

traceplot.bmeta *Traceplot to assess convergence*

Description

Function to display a plot of iteration vs. sample values for each variable in the chain

Usage

```
traceplot.bmeta(x,node,title="",lab="")
```

Arguments

x	a bmeta object with results of the model
node	variable to be displayed on the traceplot
title	title of the plot if specified
lab	name of the variable to be displayed on the traceplot

Author(s)

Tao Ding Gianluca Baio

Examples

```
### Read and format the data (binary)
data = read.csv(url("http://www.statistica.it/gianluca/bmeta/Data-bin.csv"))

### List data for binary outcome
data.list <- list(y0=data$y0,y1=data$y1,n0=data$n0,n1=data$n1)

### Select random-effects meta-analysis with t-distribution prior for binary
### data
x <- bmeta(data.list, outcome="bin", model="std.dt", type="ran")


### using output from bmeta to produce traceplot for a specific node
traceplot.bmeta(x,"mu")

### using output from bmeta to produce traceplot and specify the node used
traceplot.bmeta(x,"mu",lab="mu")
```

`writeModel` *A function to write a text file encoding the modelling assumptions*

Description

The `writeModel` function helps to select the proper model to be contained in the 'model.file' for MCMC simulation based on users' specifications.

Usage

```
writeModel(outcome, model, type, model.file, data)
```

Arguments

`outcome`	type of outcome that needs to be specified. For binary, continuous and count data, 'bin', 'ctns' and 'count' need to be specified, respectively.
`model`	type of model that needs to be specified. There are 14 options: 'std.norm','std.dt', 'reg.norm','reg.dt','std.ta','std.mv','reg.ta','reg.mv','std','std.unif','std.hc','reg', 'reg.unif','reg.hc'.
`type`	model type—either fixed-effects("fix") or random-effects model("ran") needs to be specified.
`model.file`	file containing the appropriate model selected by user
`data`	a data list containing information on observed data (including moderators).

Author(s)

Tao Ding Gianluca Baio

Index