Package ‘missingHE’

June 1, 2026

Type Package

Title Missing Outcome Data in Health Economic Evaluation

Version 1.6.1

Description Contains a suite of functions for health economic evaluations with missing outcome data.
The package can fit different types of statistical models under a fully Bayesian approach using the software 'JAGS' (which should be installed locally and which is loaded in 'missingHE' via the 'R' package 'R2jags').
Three classes of models can be fitted under a variety of missing data assumptions: selection models, pattern mixture models and hurdle models.
In addition to model fitting, 'missingHE' provides a set of specialised functions to assess model convergence and fit, and to summarise the statistical and economic results using different types of measures and graphs.
The methods implemented are described in Mason (2018) <doi:10.1002/hec.3793>, Molenberghs (2000) <doi:10.1007/978-1-4419-0300-6_18> and Gabrio (2019) <doi:10.1002/sim.8045>.

Depends R (>= 4.0.0)

Suggests knitr, rmarkdown, bookdown, devtools

VignetteBuilder knitr

License GPL-2

Encoding UTF-8

LazyData true

Imports ggpubr, ggmcmc, ggthemes, BCEA, ggplot2, bayesplot, methods,
R2jags, loo, coda, mcmcr

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Andrea Gabrio [aut, cre]

Maintainer Andrea Gabrio <`a.gabrio@maastrichtuniversity.nl`>

Repository CRAN

Date/Publication 2026-06-01 16:30:02 UTC

Contents

`anyBars` *An internal function to detect the random effects component from an object of class formula*

Description

An internal function to detect the random effects component from an object of class formula

Usage

```
anyBars(term)
```

Arguments

`term`	formula to be processed

Examples

```
#Internal function only
#no examples
#
#
```

`coef.missingHE` *Extract regression coefficient estimates from objects in the class* `missingHE`

Description

Produces a table printout with summary statistics for the regression coefficients of the health economic evaluation probabilistic model run using the function `selection`, `pattern`, `hurdle` or `lmdm`.

Usage

```
## S3 method for class 'missingHE'
coef(object, prob = c(0.025, 0.975), random = FALSE, digits = 3, ...)
```

Arguments

`object`	A `missingHE` object containing the results of the Bayesian modelling and the economic evaluation
`prob`	A numeric vector of probabilities within the range (0,1), representing the upper and lower CI sample quantiles to be calculated and returned for the estimates.
`random`	Logical. If `random` is `TRUE`, the estimates of the random effects parameters are printed, when available.
`digits`	Integer indicating the number of decimal places to be used for rounding (default = 3).
`...`	Additional arguments affecting the summary produced.

Value

Prints a table with some summary statistics, including posterior mean, standard deviation and lower and upper quantiles based on the values specified in `prob`, for the posterior distributions of the regression coefficients of the effects and costs models run using the function `selection`, `pattern`, `hurdle` or `lmdm`.

Author(s)

Andrea Gabrio

See Also

`selection` `pattern` `hurdle` `lmdm` `diagnostic` `plot.missingHE`

Examples

```
# For examples see the function \code{\link{selection}}, \code{\link{pattern}},
# \code{\link{hurdle}} or \code{\link{lmdm}}
#
#
```

`data_read_hurdle` *A function to read and re-arrange the data in different ways for the hurdle model*

Description

This internal function imports the data and outputs only those variables that are needed to run the hurdle model according to the information provided by the user.

Usage

```
data_read_hurdle(
  data,
  model.eff,
  model.cost,
  model.se,
  model.sc,
  se,
  sc,
  cov_matrix,
  type,
  center,
  fixed_e,
  fixed_c,
  random_e,
  random_c,
  trt_lev,
  trt_pos
)
```

Arguments

`data`	A data frame in which to find variables supplied in `model.eff`, `model.cost` (model formulas for effects and costs) and `model.se`, `model.sc` (model formulas for the structural effect and cost models) . Among these, effectiveness, cost and treatment indicator variables must always be provided and named 'e', 'c' and 'trt' respectively.
`model.eff`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics effectiveness outcome ('e') whose name must correspond to that used in `data`, and any covariates are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. Random effects can also be specified for each model parameter.
`model.cost`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics cost outcome ('c') whose name must correspond to that used in `data`, and any covariates are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter.
`model.se`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics effectiveness outcome ('e') whose name must correspond to that used in `data`, and any covariates used to estimate the probability of structural effects are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "probability" parameter for the structural effects through a logistic-linear model. Random effects can also be specified for each model parameter.
`model.sc`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics cost outcome ('c') whose name must correspond to that used in `data`, and any covariates used to estimate the probability of structural costs are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "probability" parameter for the structural costs through a logistic-linear model. Random effects can also be specified for each model parameter.
`se`	Structural value to be found in the effect data defined in `data`. If set to `NULL`, no structural value is chosen and a standard model for the effects is run.
`sc`	Structural value to be found in the cost data defined in `data`. If set to `NULL`, no structural value is chosen and a standard model for the costs is run.
`cov_matrix`	Covariance matrix containing all model covariates.
`type`	Type of structural value mechanism assumed, either 'SCAR' (Structural Completely At Random) or 'SAR' (Strcutural At Random).
`center`	Logical. If `center` is `TRUE` all the covariates in the model are centered.
`fixed_e`	Formula of the fixed effects specified in the effectiveness model.
`fixed_c`	Formula of the fixed effects specified in the cost model.
`random_e`	Formula of the random effects specified in the effectiveness model.
`random_c`	Formula of the random effects specified in the cost model.
`trt_lev`	Levels of the treatment indicator factor.
`trt_pos`	Numeric position of the treatment indicator within the model formula.

Examples

```
#Internal function only
#no examples
#
#
```

`data_read_lmdm` *A function to read and re-arrange the data in different ways*

Description

This internal function imports the data and outputs only those variables that are needed to run the model according to the information provided by the user.

Usage

```
data_read_lmdm(
  data,
  model.eff,
  model.cost,
  model.me,
  model.mc,
  cov_matrix,
  type,
  center,
  fixed_e,
  fixed_c,
  random_e,
  random_c,
  trt_lev,
  trt_pos
)
```

Arguments

`data`	A data frame in which to find variables supplied in `model.eff` and `model.cost`. Among these, effectiveness, cost and treatment indicator variables must always be provided and named 'e', 'c' and 'trt' respectively.
`model.eff`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics effectiveness outcome ('e') whose name must correspond to that used in `data`, and any covariates are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter.

`model.cost`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics cost outcome ('c') whose name must correspond to that used in `data`, and any covariates are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter.
`model.me`	A formula expression in conventional `R` linear modelling syntax. The response must be indicated with the term 'me'(missing effects) and any covariates used to estimate the probability of missing effects are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "probability" parameter for the missing effects through a logistic-linear model. Random effects can also be specified for each model parameter.
`model.mc`	A formula expression in conventional R linear modelling syntax. The response must be indicated with the term 'mc'(missing costs) and any covariates used to estimate the probability of missing costs should be given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "probability" parameter for the missing costs through a logistic-linear model. Random effects can also be specified for each model parameter.
`cov_matrix`	Data frame containing the covariate matrix of the model.
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR) and Missing Not At Random (MNAR).
`center`	Logical. If `center` is `TRUE` all the covariates in the model are centered.
`fixed_e`	Fixed effects variables to be included in the effects model
`fixed_c`	Fixed effects variables to be included in the costs model
`random_e`	Random effects variables to be included in the effects model
`random_c`	Random effects variables to be included in the costs model
`trt_lev`	Vector of names of each treatment factor level
`trt_pos`	Vector of name indices of each treatment factor level

Examples

```
#Internal function only
#no examples
#
#
```

`data_read_pattern` *A function to read and re-arrange the data in different ways*

Description

This internal function imports the data and outputs only those variables that are needed to run the model according to the information provided by the user.

Usage

```
data_read_pattern(
  data,
  model.eff,
  model.cost,
  cov_matrix,
  type,
  center,
  fixed_e,
  fixed_c,
  random_e,
  random_c,
  trt_lev,
  trt_pos
)
```

Arguments

`data`	A data frame in which to find variables supplied in `model.eff` and `model.cost`. Among these, effectiveness, cost and treatment indicator variables must always be provided and named 'e', 'c' and 'trt' respectively.
`model.eff`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics effectiveness outcome ('e') whose name must correspond to that used in `data`, and any covariates are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter.
`model.cost`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics cost outcome ('c') whose name must correspond to that used in `data`, and any covariates are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter.
`cov_matrix`	Data frame containing the covariate matrix of the model.
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR) and Missing Not At Random (MNAR).
`center`	Logical. If `center` is `TRUE` all the covariates in the model are centered.
`fixed_e`	Fixed effects variables to be included in the effects model
`fixed_c`	Fixed effects variables to be included in the costs model
`random_e`	Random effects variables to be included in the effects model
`random_c`	Random effects variables to be included in the costs model
`trt_lev`	Vector of names of each treatment factor level
`trt_pos`	Vector of name indices of each treatment factor level

Examples

```
#Internal function only
#no examples
#
#
```

`data_read_selection` *A function to read and re-arrange the data in different ways*

Description

This internal function imports the data and outputs only those variables that are needed to run the model according to the information provided by the user.

Usage

```
data_read_selection(
  data,
  model.eff,
  model.cost,
  model.me,
  model.mc,
  cov_matrix,
  type,
  center,
  fixed_e,
  fixed_c,
  random_e,
  random_c,
  trt_lev,
  trt_pos
)
```

Arguments

`data`	A data frame in which to find variables supplied in `model.eff` and `model.cost`. Among these, effectiveness, cost and treatment indicator variables must always be provided and named 'e', 'c' and 'trt' respectively.
`model.eff`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics effectiveness outcome ('e') whose name must correspond to that used in `data`, and any covariates are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter.

`model.cost`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economics cost outcome ('c') whose name must correspond to that used in `data`, and any covariates are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter.
`model.me`	A formula expression in conventional `R` linear modelling syntax. The response must be indicated with the term 'me'(missing effects) and any covariates used to estimate the probability of missing effects are given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "probability" parameter for the missing effects through a logistic-linear model. Random effects can also be specified for each model parameter.
`model.mc`	A formula expression in conventional R linear modelling syntax. The response must be indicated with the term 'mc'(missing costs) and any covariates used to estimate the probability of missing costs should be given on the right-hand side. If there are no covariates, specify `1` on the right hand side. By default, covariates are placed on the "probability" parameter for the missing costs through a logistic-linear model. Random effects can also be specified for each model parameter.
`cov_matrix`	Data frame containing the covariate matrix of the model.
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR) and Missing Not At Random (MNAR).
`center`	Logical. If `center` is `TRUE` all the covariates in the model are centered.
`fixed_e`	Fixed effects variables to be included in the effects model
`fixed_c`	Fixed effects variables to be included in the costs model
`random_e`	Random effects variables to be included in the effects model
`random_c`	Random effects variables to be included in the costs model
`trt_lev`	Vector of names of each treatment factor level
`trt_pos`	Vector of name indices of each treatment factor level

Examples

```
#Internal function only
#no examples
#
#
```

`diagnostic` *Diagnostic checks for assessing MCMC convergence of Bayesian models fitted in* `JAGS` *using the function* `selection`, `pattern`, `hurdle` *or* `lmdm`.

Description

The focus is restricted to full Bayesian models in cost-effectiveness analyses based on the function `selection`, `pattern`, `hurdle` and `lmdm`, with convergence of the MCMC chains that is assessed through graphical checks of the posterior distribution of the parameters of interest, Examples are density plots, trace plots, autocorrelation plots, etc. Other types of posterior checks are related to some summary MCMC statistics that are able to detect possible issues in the convergence of the algorithm, such as the potential scale reduction factor or the effective sample size. Different types of diagnostic tools and statistics are used to assess model convergence using functions contained in the package **ggmcmc**. Graphics and plots are managed using functions contained in the package **ggplot2** and **ggthemes**.

Usage

```
diagnostic(x, type = ”denplot”, param = ”all”, theme = NULL, ...)
```

Arguments

`x`	An object of class "missingHE" containing the posterior results of a full Bayesian model implemented using the function `selection`, `pattern`, `hurdle` or `lmdm`.
`type`	Type of diagnostic check to be plotted for the model parameter selected. Available choices include: ’histogram’ for histogram plots, ’denplot’ for density plots, ’traceplot’ for trace plots, ’acf’ for autocorrelation plots, ’running’ for running mean plots, ’compare’ for comparing the distribution of the whole chain with only its last part, ’cross’ for cross correlation plots, ’Rhat’ for the potential scale reduction factor, ’geweke’ for the geweke diagnostic, ’pairs’ for posterior correlation among the parameters,’caterpillar’ for caterpillar plots.
`param`	Name of the family of parameters to process, as given by a regular expression. For example the mean parameters for the effect and cost variables can be specified using ’mu.e’ and ’mu.c’, respectively. Different types of models may have different parameters depending on the assumed distributions and missing data assumptions. To see a complete list of all possible parameters by types of models assumed see details.
`theme`	Type of ggplot theme among some pre-defined themes, mostly taken from the package **ggthemes**. For a full list of available themes see details.
`...`	Additional parameters that can be provided to manage the graphical output of `diagnostic`.

Details

Depending on the types of plots specified in the argument `type`, the output of `diagnostic` can produce different combinations of MCMC visual posterior checks for the family of parameters indicated in the argument `param`. For a full list of the available plots see the description of the argument `type` or see the corresponding plots in the package **ggmcmc**.

The parameters that can be assessed through `diagnostic` are only those included in the object `x` (see Arguments). Specific character names must be specified in the argument `param` according to the specific model implemented. The available names and the parameters associated with them are:

- "mu.e" the mean of the effects across treatment arms.

- "mu.c" the mean of the costs across treatment arms.
- "sd.e" the standard deviation of the effects.
- "sd.c" the standard deviation of the costs.
- "alpha" the regression coefficients for the effects.
- "beta" the regression coefficients for the costs.
- "beta.f" the regression coefficients for the costs related to the effects predictor.
- "alpha.time" the autoregressive coefficients for the effects (only with the function `lmdm`).
- "beta.time" the autoregressive coefficients for the costs (only with the function `lmdm`).
- "random.alpha" the regression random effects coefficients for the effects.
- "random.beta" the regression random effects coefficients for the costs.
- "random.alpha.time" the autoregressive random effects coefficients for the effects (only with the function `lmdm`).
- "random.beta.time" the autoregressive random effects coefficients for the costs (only with the function `lmdm`).
- "p.e" the probability of missingness or structural values for the effects (only with the function `selection`, `hurdle` or `lmdm`).
- "p.c" the probability of missingness or structural values for the costs (only with the function `selection`, `hurdle` or `lmdm`).
- "gamma.e" the regression coefficients of missingness or structural values for the effects (only with the function `selection`, `hurdle`).
- "gamma.c" the regression coefficientd of missingness or structural values for the costs (only with the function `selection`, `hurdle`).
- "random.gamma.e" the random effects regression coefficients of missingness or structural values for the effects (only with the function `selection`, `hurdle` or `lmdm`).
- "random.gamma.c" the random effects regression coefficients of missingness or structural values for the costs (only with the function `selection`, `hurdle` or `lmdm`).
- "pattern" the probabilities of the missingness patterns (only with the function `pattern`).
- "delta.e" the mnar parameter for the effects (only with the function `selection`, `pattern` or `lmdm`).
- "delta.c" the mnar parameters for the costs (only with the function `selection`, `pattern` or `lmdm`).
- "random.delta.e" the random effects mnar parameters for the effects (only with the function `selection` or `lmdm`).
- "random.delta.c" the random effects mnar parameters for the costs (only with the function `selection` or `lmdm`).
- "all" all available parameters stored in `x`.

When the object `x` is created using the function `pattern`, pattern-specific standard deviation ("sd.e", "sd.c") and regression coefficient parameters ("alpha", "beta") for both outcomes can be visualised. The parameters associated with a missingness mechanism can be accessed only when `x` is created using the function `selection`, `pattern` or `lmdm`, while the parameters associated with the model for the structural values mechanism can be accessed only when `x` is created using the function `hurdle`.

The argument `theme` allows to customise the graphical output of the plots generated by `diagnostic` and allows to choose among a set of possible pre-defined themes taken form the package **ggtheme**. For a complete list of the available character names for each theme, see **ggthemes**.

Value

A **ggplot** object containing the plots specified in the argument `type`

Author(s)

Andrea Gabrio

References

Gelman, A. Carlin, JB., Stern, HS. Rubin, DB.(2003). *Bayesian Data Analysis, 2nd edition*, CRC Press.

Brooks, S. Gelman, A. Jones, JL. Meng, XL. (2011). *Handbook of Markov Chain Monte Carlo*, CRC/Chapman and Hall.

See Also

`ggs` `selection` `pattern` `hurdle` `lmdm`.

Examples

```
# For examples see the function \code{\link{selection}}, \code{\link{pattern}},
# \code{\link{hurdle}} or \code{\link{lmdm}}
#
#
```

`fb` *An internal function to extract the random effects component from an object of class formula*

Description

An internal function to extract the random effects component from an object of class formula

Usage

```
fb(term)
```

Arguments

`term`	formula to be processed

Examples

```
#Internal function only
#no examples
#
#
```

`hurdle` *Full Bayesian Models to handle missingness in Economic Evaluations (Hurdle Models)*

Description

Full Bayesian cost-effectiveness models to handle missing data in the outcomes using Hurdle models under a variatey of alternative parametric distributions for the effect and cost variables. Alternative assumptions about the mechanisms of the structural values are implemented using a hurdle approach. The analysis is performed using the `BUGS` language, which is implemented in the software `JAGS` using the function `jags`. The output is stored in an object of class 'missingHE'.

Usage

```
hurdle(
  data,
  model.eff,
  model.cost,
  model.se = se ~ 1,
  model.sc = sc ~ 1,
  se = 1,
  sc = 0,
  dist_e,
  dist_c,
  type,
  prob = c(0.025, 0.975),
  n.chains = 2,
  n.iter = 10000,
  n.burnin = floor(n.iter/2),
  inits = NULL,
  n.thin = 1,
  save_model = FALSE,
  prior = "default",
  center = FALSE,
  ...
)
```

Arguments

`data`	A data frame in which to find the variables supplied in `model.eff`, `model.cost` (model formulas for effects and costs) and `model.se`, `model.sc` (model formulas for the structural effect and cost models). Among these, effectiveness, cost and treatment indicator (only two arms) variables must always be provided and named 'e', 'c' and 'trt', respectively.
`model.eff`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economic effectiveness outcome ('e') whose name must correspond to that used in `data`. Any covariates in the model must be provided on the

	right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.cost`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economic cost outcome ('c') whose name must correspond to that used in `data`. Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "location" parameter of the distribution through a linear model. A joint bivariate distribution for effects and costs can be specified by including 'e' on the right-hand side of the formula for the costs model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.se`	A formula expression in conventional `R` linear modelling syntax. The response must be indicated with the term 'se'(structural effects). Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "probability" parameter for the structural effects through a logistic-linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.sc`	A formula expression in conventional `R` linear modelling syntax. The response must be indicated with the term 'sc'(structural costs). Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "probability" parameter for the structural costs through a logistic-linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`se`	Structural value to be found in the effect variables defined in `data`. If set to `NULL`, no structural value is chosen and a standard model for the effects is run.
`sc`	Structural value to be found in the cost variables defined in `data`. If set to `NULL`, no structural value is chosen and a standard model for the costs is run.
`dist_e`	Distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern').
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm').
`type`	Type of structural value mechanism assumed. Choices are Structural Completely At Random (SCAR), and Structural At Random (SAR).
`prob`	A numeric vector of probabilities within the range (0, 1), representing the upper and lower CI sample quantiles to be calculated and returned for the imputed values.
`n.chains`	Number of chains.
`n.iter`	Number of iterations.

`n.burnin`	Number of warmup iterations.
`inits`	A list with elements equal to the number of chains selected; each element of the list is itself a list of starting values for the `JAGS` model, or a function creating (possibly random) initial values. If `inits` is `NULL`, `JAGS` will generate initial values for all the model parameters.
`n.thin`	Thinning interval.
`save_model`	Logical. If `save_model` is `TRUE` a `txt` file containing the model code is printed in the current working directory.
`prior`	A list containing the hyperprior values provided by the user. Each element of this list must be a vector containing the user-provided prior distribution and parameter values and must be named with the name of the corresponding parameter. For example, the hyperprior values for the standard deviation parameter for the effects can be provided using the list `prior = list('sigma.prior.e' = c("unif", 0, 10))`. For more information about how to provide prior hypervalues for different types of parameters and models see details. If `prior` is set to 'default', the default values will be used.
`center`	Logical. If `center` is `TRUE`, all covariates in both the effect and cost models will be centred.
`...`	Additional arguments that can be provided by the user. Examples are the additional arguments that can be provided to the function `bcea` to summarise the health economic evaluation results.

Details

Depending on the distributions specified for the outcome variables in the arguments `dist_e` and `dist_c` and the type of structural value mechanism specified in the argument `type`, different hurdle models are built and run in the background by the function `hurdle`. These are mixture models defined by two components: the first one is a mass distribution at the spike, while the second is a parametric model applied to the natural range of the relevant variable. Usually, a logistic regression is used to estimate the probability of incurring a "structural" value (e.g. 0 for the costs, or 1 for the effects); this is then used to weigh the mean of the "non-structural" values estimated in the second component. A simple example can be used to show how hurdle models are specified. Consider a data set comprising a response variable y and a set of centered covariates X_j, for $i = j, ..., J$.Specifically, for each subject in the trial $i = 1, ..., n$ we define an indicator variable s_i taking value `1` if the i-th individual is associated with a structural value and `0` otherwise. This is modelled as:

$$s_i \; Bernoulli(\pi_i)$$

$$logit(\pi_i) = \sum \gamma_j X_j$$

where

- π_i is the individual probability of a structural value in y.
- γ_j represents the impact on the probability of a structural value in y of the covariate X_j.

When $\gamma_j = 0$, the model assumes a 'SCAR' mechanism, while when $\gamma_j! = 0$ the mechanism is 'SAR'. For the parameters indexing the structural value model, the default prior distributions assumed are:

- γ_j $Normal(0, 0.01)$

When user-defined hyperprior values are supplied via the argument `prior` in the function `hurdle`, the elements of this list (see Arguments) must be vectors containing the user-provided distribution name and hyperprior values and must take specific names according to the parameters they are associated with. Specifically, the names accepted by **missingHE** are the following:

- auxiliary parameters σ: "sigma.prior.e"(effects) and/or "sigma.prior.c"(costs)
- covariate parameters α_j, β_j: "alpha.prior"(effects) and/or "beta.prior"(costs)
- covariate parameters in the model of the structural values γ_j (if covariate data provided): "gamma.prior.e"(effects) and/or "gamma.prior.c"(costs)

For simplicity, here we assumed that the set of covariates X_j used in the models for the effects/costs and in the model of the structural effect/cost values is the same. However, it is possible to specify different sets of covariates for each model using the arguments in the function `hurdle` (see Arguments).

For each model, random effects can also be specified for each parameter by adding the term + (x | z) to each model formula, where x is the fixed regression coefficient for which also the random effects are desired and z is the clustering variable across which the random effects are specified (must be the name of a factor variable in the dataset). Multiple random effects can be specified using the notation + (x1 + x2 | site) for each covariate that was included in the fixed effects formula.

Value

An object of the class 'missingHE' containing the following elements

data_set A list containing the original data set provided in `data` (see Arguments). Additional information is also included about, among others, the number of observed and missing individuals, the total number of individuals by treatment arm and the indicator vectors for the structural values

model_output A list containing the output of a `JAGS` model generated from the functions `jags`, and the posterior samples for the main parameters of the model

cea A list containing the output of the economic evaluation performed using the function `bcea`

type A character variable that indicate which type of structural value mechanism used to run the model, either `SCAR` or `SAR` (see details)

data_format A character variable that indicates which type of analysis was conducted, either using a `wide` or `long` dataset

Author(s)

Andrea Gabrio

References

Ntzoufras I. (2009). *Bayesian Modelling Using WinBUGS*, John Wiley and Sons.

Daniels, MJ. Hogan, JW. (2008). *Missing Data in Longitudinal Studies: strategies for Bayesian modelling and sensitivity analysis*, CRC/Chapman Hall.

Baio, G.(2012). *Bayesian Methods in Health Economics*. CRC/Chapman Hall, London.

Gelman, A. Carlin, JB., Stern, HS. Rubin, DB.(2003). *Bayesian Data Analysis, 2nd edition*, CRC Press.

Plummer, M. *JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling.* (2003).

See Also

`jags`, `bcea`

Examples

```
# Quck example to run using subset of MenSS dataset
MenSS.subset <- MenSS[50:100, ]

# Run the model using the hurdle function assuming a SCAR mechanism
# Use only 100 iterations to run a quick check
model.hurdle <- hurdle(data = MenSS.subset, model.eff = e ~ trt, model.cost = c ~ trt,
   model.se = se ~ 1, model.sc = sc ~ 1, se = 1, sc = 0, dist_e = "norm", dist_c = "norm",
   type = "SCAR", n.chains = 2, n.iter = 100)

# Print the results of the JAGS model
print(model.hurdle)
#

# Use dic information criterion to assess model fit
pic.dic <- pic(model.hurdle, criterion = "dic", cases = "cc")
pic.dic
#

# Extract regression coefficient estimates
coef(model.hurdle)
#



# Assess model convergence using graphical tools
# Produce histograms of the posterior samples for the mean effects
diag.hist <- diagnostic(model.hurdle, type = "histogram", param = "mu.e")
#

# Compare observed effect data with imputations from the model
# using plots (posteiror means and credible intervals)
p1 <- plot(model.hurdle, class = "scatter", outcome = "effects")
#

# Summarise the CEA information from the model
summary(model.hurdle)


# Further examples which take longer to run
model.hurdle <- hurdle(data = MenSS, model.eff = e ~ trt, model.cost = c ~ trt + e,
  model.se = se ~ age, model.sc = sc ~ age, se = 1, sc = 0, dist_e = "norm", dist_c = "norm",
```

```
   type = "SAR", n.chains = 2, n.iter = 500)
#
# Print results for all imputed values
print(model.hurdle)

# Use looic to assess model fit
pic.looic <- pic(model.hurdle, criterion = "looic", cases = "cc")
pic.looic

# Show density plots for mean costs parameters
diag.den <- diagnostic(model.hurdle, type = "denplot", param = "mu.c")

# Plots of imputations for all data
p1 <- plot(model.hurdle, class = "scatter", outcome = "both")

# Summarise the CEA results
summary(model.hurdle)


#
#
```

`isAnyArgBar` *An internal function to detect the random effects component from an object of class formula*

Description

An internal function to detect the random effects component from an object of class formula

Usage

```
isAnyArgBar(term)
```

Arguments

`term`	formula to be processed

Examples

```
#Internal function only
#no examples
#
#
```

`isBar` *An internal function to detect the random effects component from an object of class formula*

Description

An internal function to detect the random effects component from an object of class formula

Usage

```
isBar(term)
```

Arguments

`term`	formula to be processed

Examples

```
#Internal function only
#no examples
#
#
```

`jagsresults` *An internal function to summarise results from BUGS model*

Description

This function hides missing data distribution from summary results of BUGS models

Usage

```
jagsresults(
  x,
  params,
  regex = FALSE,
  invert = FALSE,
  probs = c(0.025, 0.25, 0.5, 0.75, 0.975),
  signif,
  ...
)
```

Arguments

`x`	The `rjags`, `rjags.parallel`, or `mcmc.list` object for which results will be printed.
`params`	Character vector or a regular expression pattern. The parameters for which results will be printed (unless `invert` is `FALSE`, in which case results for all parameters other than those given in `params` will be returned). If `regex` is `FALSE`, only those parameters that match `params` exactly will be returned. If `regex` is `TRUE`, `param` should be a character string giving the regular expression pattern to be matched.
`regex`	If `regex` is `TRUE`, then `param` is expected to be a single string giving a text pattern to be matched. Parameters with names matching the pattern will be returned (unless `invert` is `TRUE`, which results in all parameters that do not match the pattern being returned). Text pattern matching uses regular expressions (`regex`).
`invert`	Logical. If `invert` is `TRUE`, only those parameters that do not match elements of `params` will be returned.
`probs`	A numeric vector of probabilities within range [0, 1], representing the sample quantiles to be calculated and returned.
`signif`	If supplied, all columns other than `n.eff` will have their values rounded such that the most extreme value has the specified number of significant digits.
`...`	Additional arguments accepted by `grep`, e.g. `perl=TRUE`, to allow look-around pattern matching.

Examples

```
## Not run:
## Data
N <- 100
temp <- runif(N)
rain <- runif(N)
wind <- runif(N)
a <- 0.13
beta.temp <- 1.3
beta.rain <- 0.86
beta.wind <- -0.44
sd <- 0.16
y <- rnorm(N, a + beta.temp*temp + beta.rain*rain + beta.wind*wind, sd)
dat <- list(N=N, temp=temp, rain=rain, wind=wind, y=y)

### bugs example
library(R2jags)

## Model
M <- function() {
  for (i in 1:N) {
    y[i] ~ dnorm(y.hat[i], sd^-2)
    y.hat[i] <- a + beta.temp*temp[i] + beta.rain*rain[i] + beta.wind*wind[i]
    resid[i] <- y[i] - y.hat[i]
  }
```

```
  sd ~ dunif(0, 100)
  a ~ dnorm(0, 0.0001)
  beta.temp ~ dnorm(0, 0.0001)
  beta.rain ~ dnorm(0, 0.0001)
  beta.wind ~ dnorm(0, 0.0001)
}

## Fit model
jagsfit <- jags(dat, inits=NULL,
                parameters.to.save=c('a', 'beta.temp', 'beta.rain',
                                     'beta.wind', 'sd', 'resid'),
                model.file=M, n.iter=10000)

## Output
# model summary
jagsfit

# Results for beta.rain only
jagsresults(x=jagsfit, param='beta.rain')

# Results for 'a' and 'sd' only
jagsresults(x=jagsfit, param=c('a', 'sd'))
jagsresults(x=jagsfit, param=c('a', 'sd'),
            probs=c(0.01, 0.025, 0.1, 0.25, 0.5, 0.75, 0.9, 0.975))

# Results for all parameters including the string 'beta'
jagsresults(x=jagsfit, param='beta', regex=TRUE)

# Results for all parameters not including the string 'beta'
jagsresults(x=jagsfit, param='beta', regex=TRUE, invert=TRUE)

# Note that the above is NOT equivalent to the following, which returns all
# parameters that are not EXACTLY equal to 'beta'.
jagsresults(x=jagsfit, param='beta', invert=TRUE)

# Results for all parameters beginning with 'b' or including 'sd'.
jagsresults(x=jagsfit, param='^b|sd', regex=TRUE)

# Results for all parameters not beginning with 'beta'.
# This is equivalent to using param='^beta' with invert=TRUE and regex=TRUE
jagsresults(x=jagsfit, param='^(?!beta)', regex=TRUE, perl=TRUE)

## End(Not run)
#
#
```

`lmdm` *Full Bayesian Models to handle missingness in Economic Evaluations (Longitudinal Missing Data Models)*

Description

Full Bayesian cost-effectiveness models to handle missing data in longitudinal outcomes under different missing data mechanism assumptions, using alternative parametric distributions for the effect and cost variables. The analysis is performed using the `BUGS` language, which is implemented in the software `JAGS` using the function `jags` The output is stored in an object of class 'missingHE'.

Usage

```
lmdm(
  data,
  model.eff,
  model.cost,
  model.me = me ~ 1,
  model.mc = mc ~ 1,
  dist_e,
  dist_c,
  type,
  time_dep = "AR1",
  prob = c(0.025, 0.975),
  n.chains = 2,
  n.iter = 10000,
  n.burnin = floor(n.iter/2),
  inits = NULL,
  n.thin = 1,
  save_model = FALSE,
  prior = "default",
  center = FALSE,
  ...
)
```

Arguments

`data`	A data frame in which to find the longitudinal variables supplied in `model.eff`, `model.cost` (model formulas for effects and costs) and `model.me`, `model.mc` (model formulas for the missing effect and cost models). Among these, effectiveness, cost, time and treatment indicator (only two arms) variables must always be provided and named 'e', 'c', 'time' and 'trt', respectively.
`model.eff`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economic effectiveness outcome ('e') whose name must correspond to that used in `data`. Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.cost`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economic cost outcome ('c') whose name must correspond to

	that used in `data`. Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "location" parameter of the distribution through a linear model. A joint bivariate distribution for effects and costs can be specified by including ’e’ on the right-hand side of the formula for the costs model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.me`	A formula expression in conventional `R` linear modelling syntax. The response must be indicated with the term ’me’ (missing effects) and any covariates must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "probability" parameter for the missing effects through a logistic-linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.mc`	A formula expression in conventional `R` linear modelling syntax. The response must be indicated with the term ’mc’ (missing costs) and any covariates must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "probability" parameter for the missing costs through a logistic-linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`dist_e`	Distribution assumed for the effects. Current available chocies are: Normal (’norm’), Beta (’beta’), Gamma (’gamma’), Exponential (’exp’), Weibull (’weib’), Logistic (’logis’), Poisson (’pois’), Negative Binomial (’negbin’) or Bernoulli (’bern’).
`dist_c`	Distribution assumed for the costs. Current available chocies are: Normal (’norm’), Gamma (’gamma’) or LogNormal (’lnorm’).
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR) and Missing Not At Random (MNAR).
`time_dep`	Type of dependence structure assumed between effectiveness and cost outcomes. Current choices include: autoregressive structure of order one (’AR1’) - default, bivariate at each time (’biv’) and independence (’none’).
`prob`	A numeric vector of probabilities within the range (0, 1), representing the upper and lower CI sample quantiles to be calculated and returned for the imputed values.
`n.chains`	Number of chains.
`n.iter`	Number of iterations.
`n.burnin`	Number of warmup iterations.
`inits`	A list with elements equal to the number of chains selected; each element of the list is itself a list of starting values for the `JAGS` model, or a function creating (possibly random) initial values. If `inits` is `NULL`, `JAGS` will generate initial values for all the model parameters.
`n.thin`	Thinning interval.
`save_model`	Logical. If `save_model` is `TRUE`, a `txt` file containing the model code is printed in the current working directory.

`prior`	A list containing the hyperprior values provided by the user. Each element of this list must be a vector containing the user-provided prior distribution and parameter values and must be named with the name of the corresponding parameter. For example, the hyperprior values for the standard deviation parameter for the effects can be provided using the list `prior = list('sigma.prior.e' = c("unif", 0, 10))`. For more information about how to provide prior hypervalues for different types of parameters and models see details. If `prior` is set to 'default', the default values will be used.
`center`	Logical. If `center` is `TRUE`, all covariates in both the effect and cost models will be centred.
`...`	Additional arguments that can be provided by the user. Examples are the additional arguments that can be provided to the function `bcea` to summarise the health economic evaluation results.

Details

Depending on the distributions specified for the outcome variables in the arguments `dist_e` and `dist_c` and the type of missingness mechanism specified in the argument `type`, different models are built and run in the background by the function `lmdm`. These models fit multinomial logistic regressions to estimate the probability of a given missing data pattern `k` (1 = completers, 2 = intermittent, 3 = dropout) in one or both longitudinal outcomes. A simple example can be used to show how longitudinal missing data models are specified. Consider a longitudinal data set comprising a response variable y measured at `S` occasions and a set of centered covariate X_j, for $i = j, ..., J$. For each subject in the trial $i = 1, ..., n$ and time $s = 1, ..., S$ we define an indicator variable m_i taking value `k = 1` if the i-th individual is associated with no missing value (completer), a value `k = 2` for intermittent missingness over the study period, and a value `k = 3` for dropout. This is modelled as:

$$m_i\ Multinomial(\pi_i^k)$$

$$\pi_i^k = \phi_i^k / \sum \phi_i$$

$$log(\phi_i^k) = \sum \gamma_j^k X_j + \delta^k y_i$$

where

- π_i^k is the individual probability of a missing value in y for pattern k at a given time.
- γ_j^k represents the impact on the missing data probability in y of the covariate X_j for pattern k at a given time.
- δ^k represents the impact on the missing data probability in y for pattern k of missingness itself at a given time.

When $\delta = 0$ the model assumes a 'MAR' mechanism, while when $\delta! = 0$ the mechanism is 'MNAR'. For the parameters indexing the missingness model, the default prior distributions assumed are:

- $\gamma_j^k\ Normal(0, 0.01)$
- $\delta^k\ Normal(0, 1)$

When user-defined hyperprior values are supplied via the argument `prior` in the function `lmdm`, the elements of this list (see Arguments) must be vectors containing the user-provided distribution name and hyperprior values and must take specific names according to the parameters they are associated with. Specifically, the names for the parameters indexing the model which are accepted by **missingHE** are the following:

- auxiliary parameters σ: "sigma.prior.e"(effects) and/or "sigma.prior.c"(costs)
- covariate parameters α_j and β_j: "alpha.prior"(effects) and/or "beta.prior"(costs)
- covariate parameters in the missingness model γ_j (if covariate data provided): "gamma.prior.e"(effects) and/or "gamma.prior.c"(costs)
- mnar parameter δ: "delta.prior.e"(effects) and/or "delta.prior.c"(costs)

For simplicity, here we have assumed that the set of covariates X_j used in the models for the effects/costs and in the model of the missing effect/cost values is the same. However, it is possible to specify different sets of covariates for each model using the arguments in the function `lmdm` (see Arguments).

For each model, random effects can also be specified for each parameter by adding the term + (x | z) to each model formula, where x is the fixed regression coefficient for which also the random effects are desired and z is the clustering variable across which the random effects are specified (must be the name of a factor variable in the dataset).

Value

An object of the class 'missingHE' containing the following elements

data_set A list containing the original data set provided in `data` (see Arguments). Additional information is also included about, among others, the number of observed and missing individuals, the total number of individuals by treatment arm and the indicator vectors for the missing values

model_output A list containing the output of a `JAGS` model generated from the functions `jags`, and the posterior samples for the main parameters of the model

cea A list containing the output of the economic evaluation performed using the function `bcea`

type A character variable that indicate which type of missing value mechanism used to run the model, either `MAR` or `MNAR` (see details)

data_format A character variable that indicates which type of analysis was conducted, either using a `wide` or `long` dataset

time_dep A character variable that indicate which type of time dependence assumption was made, either `none` or `AR1`

Author(s)

Andrea Gabrio

References

Mason, AJ. Gomes, M. Carpenter, J. Grieve, R. (2021). *Flexible Bayesian longitudinal models for cost-effectiveness analyses with informative missing data*. Health economics, 30(12), 3138-3158.

Daniels, MJ. Hogan, JW. *Missing Data in Longitudinal Studies: strategies for Bayesian modelling and sensitivity analysis*, CRC/Chapman Hall.

Baio, G.(2012). *Bayesian Methods in Health Economics*. CRC/Chapman Hall, London.

Gelman, A. Carlin, JB., Stern, HS. Rubin, DB.(2003). *Bayesian Data Analysis, 2nd edition*, CRC Press.

Plummer, M. *JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling.* (2003).

See Also

`jags`, `bcea`

Examples

```
# Quick example to run using subset of PBS dataset

# Load longitudinal dataset

PBS.long <- PBS


# Run the model using the long_miss function assuming a MAR mechanism
# Use only 100 iterations to run a quick check
model.long <- lmdm(data = PBS.long, model.eff = e ~ trt, model.cost = c ~ trt,
   model.me = me ~ 1, model.mc = mc ~ 1, dist_e = "norm", dist_c = "norm",
   type = "MAR", n.chains = 2, n.iter = 100, time_dep = "none")

# Extract regression coefficient estimates
coef(model.long)
#

# Summarise the CEA information from the model
summary(model.long)

# Further examples which take longer to run
model.long <- lmdm(data = PBS.long, model.eff = e ~ trt, model.cost = c ~ trt + age,
   model.me = me ~ 1, model.mc = mc ~ 1, dist_e = "norm", dist_c = "norm",
   type = "MAR", n.chains = 2, n.iter = 500, time_dep = "none")

# Use looic to assess model fit
pic.looic <- pic(model.long, criterion = "looic")
pic.looic

# Show density plots for all parameters
diag.den <- diagnostic(model.long, type = "denplot", param = "alpha")

# Plots of imputations for all effect data
```

```
p1 <- plot(model.long, class = "scatter", outcome = "effects", time.plot = "all")

# Summarise the CEA results
summary(model.long)


#
#
```

`MenSS` *MenSS economic data on STIs*

Description

Data from a pilot RCT trial (The MenSS trial) on youn men at risk of Sexually Trasmitted Infections (STIs). A total of 159 individuals were enrolled in trial: 75 in the control (t=1) and 84 in the active intervention (t=2). Clinical and health economic outcome data were collected via self-reported questionnaires at four time points throughout the study: baseline, 3 months, 6 months and 12 months follow-up. Health economic data include utility scores related to quality of life and costs, from which QALYs and total costs were then computed using the area under the curve method and by summing up the cost components at each time point. Clinical data include the total number of instances of unprotected sex and whether the individual was associated with an STI diagnosis or not. Baseline data are available for the utilities (no baseline costs collected), instances of unprotected sex, sti diagnosis, age, ethnicity and employment variables.

Data from a pilot RCT trial (The MenSS trial) on youn men at risk of Sexually Trasmitted Infections (STIs). A total of 159 individuals were enrolled in trial: 75 in the control (t=1) and 84 in the active intervention (t=2). Clinical and health economic outcome data were collected via self-reported questionnaires at four time points throughout the study: baseline, 3 months, 6 months and 12 months follow-up. Health economic data include utility scores related to quality of life and costs, from which QALYs and total costs were then computed using the area under the curve method and by summing up the cost components at each time point. Clinical data include the total number of instances of unprotected sex and whether the individual was associated with an STI diagnosis or not. Baseline data are available for the utilities (no baseline costs collected), instances of unprotected sex, sti diagnosis, age, ethnicity and employment variables.

Usage

```
data(MenSS)
```

```
data(MenSS)
```

Format

A data frame with 159 rows and 12 variables

A data frame with 159 rows and 12 variables

Details

id id number

e Quality Adjusted Life Years (QALYs)

c Total costs in pounds

u.0 baseline utilities

age Age in years

ethnicity binary: white (1) and other (0)

employment binary: working (1) and other (0)

t Treatment arm indicator for the control (t=1) and the active intervention (t=2)

sex_inst.0 baseline number of instances of unprotected sex

sex_inst number of instances of unprotected sex at 12 months follow-up

sti.0 binary : baseline sti diagnosis (1) and no baseline sti diagnosis (0)

sti binary : sti diagnosis (1) and no sti diagnosis (0) at 12 months follow-up

site site number

id id number

e Quality Adjusted Life Years (QALYs)

c Total costs in pounds

u.0 baseline utilities

age Age in years

ethnicity binary: white (1) and other (0)

employment binary: working (1) and other (0)

trt Treatment arm indicator for the control (t=1) and the active intervention (t=2)

sex_inst.0 baseline number of instances of unprotected sex

sex_inst number of instances of unprotected sex at 12 months follow-up

sti.0 binary : baseline sti diagnosis (1) and no baseline sti diagnosis (0)

sti binary : sti diagnosis (1) and no sti diagnosis (0) at 12 months follow-up

site site number

References

Bailey et al. (2016) Health Technology Assessment 20 (PubMed)

Bailey et al. (2016) Health Technology Assessment 20 (PubMed)

Examples

```
MenSS <- data(MenSS)
summary(MenSS)
str(MenSS)
MenSS <- data(MenSS)
summary(MenSS)
str(MenSS)
```

`nobars_` *An internal function to separate the fixed and random effects components from an object of class formula*

Description

An internal function to separate the fixed and random effects components from an object of class formula

Usage

```
nobars_(term)
```

Arguments

`term`	formula to be processed

Examples

```
#Internal function only
#no examples
#
#
```

`pattern` *Full Bayesian Models to handle missingness in Economic Evaluations (Pattern Mixture Models)*

Description

Full Bayesian cost-effectiveness models to handle missing data in the outcomes under different missing data mechanism assumptions, using alternative parametric distributions for the effect and cost variables and using a pattern mixture model approach to identify the model. The analysis is performed using the `BUGS` language, which is implemented in the software `JAGS` using the function `jags` The output is stored in an object of class 'missingHE'.

Usage

```
pattern(
  data,
  model.eff,
  model.cost,
  dist_e,
  dist_c,
  type,
  restriction = "CC",
```

```
  prob = c(0.025, 0.975),
  n.chains = 2,
  n.iter = 10000,
  n.burnin = floor(n.iter/2),
  inits = NULL,
  n.thin = 1,
  save_model = FALSE,
  prior = "default",
  center = FALSE,
  ...
)
```

Arguments

`data`	A data frame in which to find the variables supplied in `model.eff`, `model.cost` (model formulas for effects and costs). Among these, effectiveness, cost and treatment indicator (only two arms) variables must always be provided and named 'e', 'c' and 'trt', respectively.
`model.eff`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economic effectiveness outcome ('e') whose name must correspond to that used in `data`. Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.cost`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economic cost outcome ('c') whose name must correspond to that used in `data`. Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "location" parameter of the distribution through a linear model. A joint bivariate distribution for effects and costs can be specified by including 'e' on the right-hand side of the formula for the costs model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`dist_e`	Distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern').
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm').
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR) and Missing Not At Random (MNAR).
`restriction`	type of identifying restriction to be imposed to identify the distributions of the missing data in each pattern. Available choices are: complete case restrcition ('CC') - default - or available case restriction ('AC').

`prob`	A numeric vector of probabilities within the range (0, 1), representing the upper and lower CI sample quantiles to be calculated and returned for the imputed values.
`n.chains`	Number of chains.
`n.iter`	Number of iterations.
`n.burnin`	Number of warmup iterations.
`inits`	A list with elements equal to the number of chains selected; each element of the list is itself a list of starting values for the `JAGS` model, or a function creating (possibly random) initial values. If `inits` is `NULL`, `JAGS` will generate initial values for all the model parameters.
`n.thin`	Thinning interval.
`save_model`	Logical. If `save_model` is `TRUE`, a `txt` file containing the model code is printed in the current working directory.
`prior`	A list containing the hyperprior values provided by the user. Each element of this list must be a vector containing the user-provided prior distribution and parameter values and must be named with the name of the corresponding parameter. For example, the hyperprior values for the standard deviation parameter for the effects can be provided using the list `prior = list('sigma.prior.e' = c("unif", 0, 10))`. For more information about how to provide prior hypervalues for different types of parameters and models see details. If `prior` is set to 'default', the default values will be used.
`center`	Logical. If `center` is `TRUE`, all covariates in both the effect and cost models will be centred.
`...`	Additional arguments that can be provided by the user. Examples are the additional arguments that can be provided to the function `bcea` to summarise the health economic evaluation results.

Details

Depending on the distributions specified for the outcome variables in the arguments `dist_e` and `dist_c` and the type of missingness mechanism specified in the argument `type`, different pattern mixture models are built and run in the background by the function `pattern`. The model for the outcomes is fitted in each missingness pattern and the parameters indexing the missing data distributions are identified using: the corresponding parameters identified from the observed data in other patterns (under 'MAR'); or a combination of the parameters identified by the observed data and some sensitivity parameters (under 'MNAR'). A simple example can be used to show how pattern mixture models are specified. Consider a data set comprising a response variable y and a set of centered covariates X_j. We denote with d_i the patterns' indicator variable for each subject in the trial $i = 1, ..., n$ such that: $d_i = 1$ indicates the completers (both e and c observed), $d_i = 2$ and $d_i = 3$ indicate that only the costs or effects are observed, respectively, while $d_i = 4$ indicates that neither of the two outcomes is observed. d_i is assigned a multinomial distribution, which probabilities are modelled using a Dirichlet prior. Next, the outcomes model is fitted in each pattern and parameters that cannot be identified in each pattern (d = 2, 3, 4), e.g. $mu_e[d]$ and `mu_c[d]`, are identified using some restrictions based on the parameters estimated from other patterns. Two choices are currently available: the complete cases ('CC') or available cases ('AC'). For example, using the

’CC’ restriction, the parameters indexing the distributions of the missing data are identified as:

$$mu_e[2] = \mu_e[4] = \mu_e[1] + \delta_e$$

$$mu_c[3] = \mu_c[4] = \mu_c[1] + \delta_c$$

where

- $\mu_e[1]$ is the effects mean for the completers.
- $\mu_c[1]$ is the costs mean for the completers.
- δ_e is the sensitivity parameters associated with the marginal effects mean.
- δ_c is the sensitivity parameters associated with the marginal costs mean.

If the ’AC’ restriction is chosen, only the parameters estimated from the observed data in pattern 2 (costs) and pattern 3 (effects) are used to identify those in the other patterns. When $\delta_e = 0$ and $\delta_c = 0$ the model assumes a ’MAR’ mechanism. When $\delta_e! = 0$ and/or $\delta_c! = 0$ ’MNAR’ departures for the effects and/or costs are explored with priors on sensitivity parameters that must be provided by the user (see Arguments).

When user-defined hyperprior values are supplied via the argument `prior` in the function `pattern`, the elements of this list (see Arguments) must be vectors containing the user-provided distribution name and hyperprior values and must take specific names according to the parameters they are associated with. Specifically, the names for the parameters indexing the model which are accepted by **missingHE** are the following:

- auxiliary parameters σ: "sigma.prior.e"(effects) and/or "sigma.prior.c"(costs)
- covariate parameters α_j and β_j: "alpha.prior"(effects) and/or "beta.prior"(costs)
- covariate parameters in the missingness model γ_j (if covariate data provided): "gamma.prior.e"(effects) and/or "gamma.prior.c"(costs)
- sensitivity parameters δ: "delta.prior.e"(effects) and/or "delta.prior.c"(costs)
- missingness patterns’ probabilities π: "patterns.prior"

For each outcome model, random effects can also be specified for each parameter by adding the term + (x | z) to each model formula, where x is the fixed regression coefficient for which also the random effects are desired and z is the clustering variable across which the random effects are specified (must be the name of a factor variable in the dataset). Multiple random effects can be specified using the notation + (x1 + x2 | site) for each covariate that was included in the fixed effects formula.

Value

An object of the class ’missingHE’ containing the following elements

data_set A list containing the original data set provided in `data` (see Arguments). Additional information is also included about, among others, the number of observed and missing individuals, the total number of individuals by treatment arm and the indicator vectors for the missing values

model_output A list containing the output of a `JAGS` model generated from the functions `jags`, and the posterior samples for the main parameters of the model

cea A list containing the output of the economic evaluation performed using the function `bcea`

type A character variable that indicate which type of missing value mechanism used to run the model, either `MAR` or `MNAR` (see details)

data_format A character variable that indicates which type of analysis was conducted, either using a `wide` or `long` dataset

Author(s)

Andrea Gabrio

References

Daniels, MJ. Hogan, JW. *Missing Data in Longitudinal Studies: strategies for Bayesian modelling and sensitivity analysis*, CRC/Chapman Hall.

Baio, G.(2012). *Bayesian Methods in Health Economics*. CRC/Chapman Hall, London.

Gelman, A. Carlin, JB., Stern, HS. Rubin, DB.(2003). *Bayesian Data Analysis, 2nd edition*, CRC Press.

Plummer, M. *JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling.* (2003).

See Also

`jags`, `bcea`

Examples

```
# Quck example to run using subset of MenSS dataset
MenSS.subset <- MenSS[50:100, ]

# Run the model using the pattern function assuming a MAR mechanism
# Use only 100 iterations to run a quick check
model.pattern <- pattern(data = MenSS.subset,model.eff = e ~ trt, model.cost = c ~ trt,
   dist_e = "norm", dist_c = "norm", type = "MAR", n.chains = 2, n.iter = 100)

# Print the results of the JAGS model
print(model.pattern)
#

# Use dic information criterion to assess model fit
pic.dic <- pic(model.pattern, criterion = "dic", cases = "cc")
pic.dic
#

# Extract regression coefficient estimates
coef(model.pattern)
#



# Assess model convergence using graphical tools
```

```
# Produce histograms of the posterior samples for the mean effects
diag.hist <- diagnostic(model.pattern, type = "histogram", param = "mu.e")
#

# Compare observed effect data with imputations from the model
# using plots (posteiror means and credible intervals)
p1 <- plot(model.pattern, class = "scatter", outcome = "effects")
#

# Summarise the CEA information from the model
summary(model.pattern)


# Further examples which take longer to run
model.pattern <- pattern(data = MenSS, model.eff = e ~ trt, model.cost = c ~ trt + e,
   dist_e = "norm", dist_c = "norm", type = "MAR", n.chains = 2, n.iter = 500)
#
# Print results for all imputed values
print(model.pattern)

# Use looic to assess model fit
pic.looic <- pic(model.pattern, criterion = "looic", cases = "cc")
pic.looic

# Show density plots for mean costs parameters
diag.den <- diagnostic(model.pattern, type = "denplot", param = "mu.c")

# Plots of imputations for all data
p1 <- plot(model.pattern, class = "scatter", outcome = "both")

# Summarise the CEA results
summary(model.pattern)


#
#
```

PBS *PBS economic data on intellectual disability and challenging behaviour*

Description

Longitudinal data from a cluster RCT trial (The PBS trial) on people suffering from intellectual disability and challenging behaviour. A total of 244 individuals across 23 sites were enrolled in the trial: 136 in the control (t=1) and 108 in the active intervention (t=2). Health economic outcome data were collected via self-reported questionnaires at three time points throughout the study: baseline (time=1), 6 months (time=2) and 12 months (time=3) follow-up, and included utility scores related to quality of life and costs. Baseline data are available for age, gender, ethnicity, living status, type of carer, marital status, and disability level variables.

Usage

```
data(PBS)
```

Format

A data frame with 732 rows and 16 variables

Details

id id number

time time indicator

e HRQoL utilities

c costs (in pounds)

trt Treatment arm indicator for the control (t=1) and the active intervention (t=2)

age age in years

gender binary: male (1) and female (0)

ethnicity binary: white (1) and other (0)

carer binary: paid carer (1) and family carer (0)

marital binary: single (1) and married (0)

living categorical: alone (1), with partner (2) and with parents (3)

disability categorical: mild (1), moderate (2) and severe (3)

site site number

References

Hassiotis et al. (2014) BMC Psychiatry 14 (PubMed)

Examples

```
PBS <- data(PBS)
summary(PBS)
str(PBS)
```

`pic` *Predictive information criteria for Bayesian models fitted in* `JAGS` *using the function* `selection`, `pattern`, `hurdle` *or* `lmdm`

Description

Efficient approximate leave-one-out cross validation (LOO), deviance information criterion (DIC) and widely applicable information criterion (WAIC) for Bayesian models, calculated on the observed data.

Usage

```
pic(x, criterion = "dic", cases = "cc", ...)
```

Arguments

`x`	A `missingHE` object containing the results of a Bayesian model fitted in cost-effectiveness analysis using the function `selection`, `pattern`, `hurdle` or `lmdm`.
`criterion`	type of information criteria to be produced. Available choices are `'dic'` for the Deviance Information Criterion, `'waic'` for the Widely Applicable Information Criterion, and `'looic'` for the Leave-One-Out Information Criterion.
`cases`	group of cases for which information criteria should be computed for: either `'all'` for all cases, `'cc'` for complete cases, `'ac_e'` and `'ac_c'` for only the observed effect and cost cases, respectively.
`...`	Additional parameters that can be provided to manage the output of `pic` when `'looic'` is selected. For more details see **bayesplot**.

Details

The Deviance Information Criterion (DIC), Leave-One-Out Information Criterion (LOOIC) and the Widely Applicable Information Criterion (WAIC) are methods for estimating out-of-sample predictive accuracy from a Bayesian model using the log-likelihood evaluated at the posterior simulations of the parameters. DIC is computationally simple to calculate but it is known to have some problems, arising in part from it not being fully Bayesian in that it is based on a point estimate. LOOIC can be computationally expensive but can be easily approximated using importance weights that are smoothed by fitting a generalised Pareto distribution to the upper tail of the distribution of the importance weights. WAIC is fully Bayesian and closely approximates Bayesian cross-validation. Unlike DIC, WAIC is invariant to parameterisation and also works for singular models. In finite cases, WAIC and LOOIC give similar estimates, but for influential observations WAIC underestimates the effect of leaving out one observation.

Value

A named list containing different predictive information criteria results and quantities according to the value of `criterion`. In all cases, the measures are computed on the observed data for the specific modules of the model selected in `module`.

d_bar Posterior mean deviance (only if `criterion` is `'dic'`).

pD Effective number of parameters calculated with the formula used by `JAGS` (only if `criterion` is `'dic'`).

dic Deviance Information Criterion calculated with the formula used by `JAGS` (only if `criterion` is `'dic'`).

d_hat Deviance evaluated at the posterior mean of the parameters and calculated with the formula used by `JAGS` (only if `criterion` is `'dic'`)

elpd, elpd_se Expected log pointwise predictive density and standard error calculated on the observed data for the model nodes indicated in `module` (only if `criterion` is `'waic'` or `'looic'`).

p, p_se Effective number of parameters and standard error calculated on the cases indicated in `cases`. (only if `criterion` is `'waic'` or `'looic'`).

looic, looic_se The leave-one-out information criterion and standard error calculated on the cases indicated in `cases`. (only if `criterion` is `'looic'`).

waic, waic_se The widely applicable information criterion and standard error calculated on the cases indicated in `cases`. (only if `criterion` is `'waic'`).

pointwise A matrix containing the pointwise contributions of each of the above measures calculated on the cases indicated in `cases`. (only if `criterion` is `'waic'` or `'looic'`).

diagnostics A named list containing additional diagnostic measures (only if `criterion` is `'looic'`). See `loo` for details about interpreting the list elements.

psis_object A named list containing the matrix of (smoothed) log weights (only if `criterion` is `'looic'` with the optional argument `'save_psis'` is set to `TRUE`). See `loo` for details about interpreting the list elements.'

Author(s)

Andrea Gabrio

References

Plummer, M. *JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling.* (2003).

Vehtari, A. Gelman, A. Gabry, J. (2016a) Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*. Advance online publication.

Vehtari, A. Gelman, A. Gabry, J. (2016b) Pareto smoothed importance sampling. *ArXiv* preprint.

Gelman, A. Hwang, J. Vehtari, A. (2014) Understanding predictive information criteria for Bayesian models. *Statistics and Computing* 24, 997-1016.

Watanable, S. (2010). Asymptotic equivalence of Bayes cross validation and widely application information criterion in singular learning theory. *Journal of Machine Learning Research* 11, 3571-3594.

See Also

`jags`, `loo`, `waic`

Examples

```
# For examples see the function \code{\link{selection}}, \code{\link{pattern}},
# \code{\link{hurdle}} or \code{\link{lmdm}}
#
#
```

`plot.missingHE` *Plot method for the imputed data contained in the objects of class* `missingHE`

Description

Produces a plot of the observed and imputed values (with credible intervals) for the effect and cost outcomes from a Bayesian cost-effectiveness analysis model with two treatment arms, implemented using the function `selection`, `pattern`, `hurdle` or `lmdm`. The graphical layout is obtained from the functions contained in the package **ggplot2** and **ggthemes**.

Usage

```
## S3 method for class 'missingHE'
plot(
  x,
  prob = c(0.025, 0.975),
  class = "scatter",
  outcome = "both",
  trt = "all",
  time_plot = NULL,
  theme = NULL,
  only.plot = TRUE,
  ...
)
```

Arguments

`x`	A `missingHE` object containing the results of the Bayesian model for cost-effectiveness analysis.
`prob`	A numeric vector of probabilities representing the upper and lower CI sample quantiles to be calculated and returned for the imputed values.
`class`	Type of the plot comparing the observed and imputed outcome data. Available choices are 'scatter' (default), 'histogram' and 'boxplot' for a scatter, histogram or a boxplot of the observed and average imputed outcome data, respectively.
`outcome`	The outcome variables that should be displayed. Options are: 'both' (default) for both effects and costs; 'effects' or 'costs' for the effects or costs separately.
`trt`	treatment group for which plots should be displayed. Choices include: 'all' (default) for all groups; 'none' for results across all groups; any character or numeric value denoting the treatment group name or index associated with the treatment variable in the original data set.
`time_plot`	Time point for which the graphs should be displayed.
`theme`	Type of ggplot theme among some pre-defined themes, mostly taken from the package **ggthemes**. For a full list of available themes see details.

`only.plot`	Logical. If `only.plot` is `TRUE` (default) only the plot is returned. When `only.plot` is `FALSE`, a list containing the plot object and the data is returned instead.
`...`	Additional parameters that can be provided to manage the output of `plot.missingHE`.

Details

The function produces a plot of the observed and imputed effect and cost data in a two-arm based cost-effectiveness model implemented using the function `selection`, `pattern`, `hurdle` or or `lmdm`. The purpose of this graph is to visually compare the outcome values for the fully-observed individuals with those imputed by the model for the missing individuals. For scatter plots, average imputed values are also associated with credible intervals specified in the argument `prob`. The argument `theme` allows to customise the graphical aspect of the plots generated by `plot.missingHE` and allows to choose among a set of possible pre-defined themes taken form the package **ggtheme**. For a complete list of the available character names for each theme and scheme set, see **ggthemes** and **bayesplot**.

Value

A `ggplot` object containing the plots specified in the argument `class`.

Author(s)

Andrea Gabrio

References

Daniels, MJ. Hogan, JW. (2008) *Missing Data in Longitudinal Studies: strategies for Bayesian modelling and sensitivity analysis*, CRC/Chapman Hall.

Molenberghs, G. Fitzmaurice, G. Kenward, MG. Tsiatis, A. Verbeke, G. (2015) *Handbook of Missing Data Methodology*, CRC/Chapman Hall.

See Also

`selection` `pattern` `hurdle` `lmdm` `diagnostic`

Examples

```
# For examples see the function \code{\link{selection}}, \code{\link{pattern}},
# \code{\link{hurdle}} or \code{\link{lmdm}}
#
#
```

`ppc` *Posterior predictive checks for assessing the fit to the observed data of Bayesian models implemented in* `JAGS` *using the function* `selection`, `pattern`, `hurdle` *or* `lmdm`

Description

The focus is restricted to full Bayesian models in cost-effectiveness analyses based on the function `selection`, `pattern`, `hurdle` or `lmdm` with the fit to the observed data being assessed through graphical checks based on the posterior replications generated from the model. Examples include the comparison of histograms, density plots, intervals, test statistics, evaluated using both the observed and replicated data. Different types of posterior predictive checks are implemented to assess model fit using functions contained in the package **bayesplot**. Graphics and plots are managed using functions contained in the package **ggplot2** and **ggthemes**.

Usage

```
ppc(
  x,
  type = "histogram",
  outcome = "both",
  ndisplay = 15,
  trt = "all",
  theme = NULL,
  scheme_set = NULL,
  ...
)
```

Arguments

`x`	An object of class "missingHE" containing the posterior results of a full Bayesian model implemented using the function `selection`, `pattern`, `hurdle` or `lmdm`.
`type`	Type of posterior predictive check among some pre-defined types, mostly taken from the package **bayesplot**. For a full list of available options see details.
`outcome`	The outcome variables that should be displayed. Options are: 'both' (default) for both effects and costs; 'effects' or 'costs' for the effects or costs separately.
`ndisplay`	Number of posterior replications to be used to generate the plots.
`trt`	treatment group for which plots should be displayed. Choices include: 'all' (default) for all groups; 'none' for results across all groups; any character or numeric value denoting the treatment group name or index associated with the treatment variable in the original data set.
`theme`	Type of ggplot theme among some pre-defined themes, mostly taken from the package **ggthemes**. For a full list of available themes see details.
`scheme_set`	Type of scheme sets among some pre-defined schemes, mostly taken from the package **bayesplot**. For a full list of available themes see details.

`...`	Additional parameters that can be provided to manage the output of `ppc`. For more details see **bayesplot**.

Details

The function produces different types of graphical posterior predictive checks using the estimates from a Bayesian cost-effectiveness model implemented with the function `selection`, `pattern`, `hurdle` or `lmdm`. The purpose of these checks is to visually compare the distribution (or some relevant quantity) of the observed data with respect to that from the replicated data for both effectiveness and cost outcomes in each treatment arm. Since predictive checks are meaningful only with respect to the observed data, only the observed outcome values are used to assess the fit of the model. The arguments `theme` and `scheme_set` allow to customise the graphical aspect of the plots generated by `ppc` and allow to choose among a set of possible pre-defined themes and scheme sets taken form the package **ggtheme** and `bayesplot`. For a complete list of the available character names for each theme and scheme set, see **ggthemes** and **bayesplot**.

Value

A `ggplot` object containing the plots specified in the argument `type`.

Author(s)

Andrea Gabrio

References

Gelman, A. Carlin, JB., Stern, HS. Rubin, DB.(2003). *Bayesian Data Analysis, 2nd edition*, CRC Press.

See Also

`selection` `pattern` `hurdle` `lmdm` `diagnostic`

Examples

```
# For examples see the function \code{\link{selection}}, \code{\link{pattern}},
# \code{\link{hurdle}} or \code{\link{lmdm}}
#
```

`print.missingHE` *Print method for the posterior results contained in the objects of class* `missingHE`

Description

Prints the summary table for the model fitted, with the estimate of the parameters and/or missing values.

Usage

```
## S3 method for class 'missingHE'
print(x, display = "fixed", digits = 3, ...)
```

Arguments

`x`	A `missingHE` object containing the results of the Bayesian model run using the function `selection`, `pattern`, `hurdle` or `lmdm`.
`display`	Set of parameters for which posterior summaries are displayed. Choices are: fixed effects ('fixed'), random effects ('random'), individual log likelihood values ('loglik'), individual conditional parameter values ('conditional') and individual imputed values ('mis').
`digits`	Integer indicating the number of decimal places to be used for rounding (default = 3).
`...`	additional arguments affecting the output produced. For example: `param` = JAGS name assigned to the specific model parameters to be shown in the printed table.

Author(s)

Andrea Gabrio

Examples

```
# For examples see the function \code{\link{selection}}, \code{\link{pattern}},
# \code{\link{hurdle}} or \code{\link{lmdm}}
#
#
```

`prior_hurdle` *An internal function to change the hyperprior parameters in the hurdle model provided by the user depending on the type of structural value mechanism and outcome distributions assumed*

Description

This function modifies default hyper prior parameter values in the type of hurdle model selected according to the type of structural value mechanism and distributions for the outcomes assumed.

Usage

```
prior_hurdle(type, dist_e, dist_c, model_txt_info, model_string_jags)
```

Arguments

`type`	Type of structural value mechanism assumed. Choices are Structural Completely At Random (SCAR) and Structural At Random (SAR). For a complete list of all available hyper parameters and types of models see the manual.
`dist_e`	distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weibull'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('nbinom') or Bernoulli ('bern')
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm')
`model_txt_info`	list containing model specification information used to write the txt file of the JAGS model.
`model_string_jags`	text file of the model.

Examples

```
#Internal function only
#no examples
#
#
```

`prior_lmdm` *An internal function to change the hyperprior parameters in the selection model provided by the user depending on the type of missingness mechanism and outcome distributions assumed*

Description

This function modifies default hyper prior parameter values in the type of longitudinal missing data model selected according to the type of missingness mechanism and distributions for the outcomes assumed.

Usage

```
prior_lmdm(type, dist_e, dist_c, model_txt_info, model_string_jags)
```

Arguments

`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR). For a complete list of all available hyper parameters and types of models see the manual.

`dist_e`	distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern')
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm')
`model_txt_info`	list containing model specification information used to write the txt file of the JAGS model.
`model_string_jags`	text file of the model.

Examples

```
#Internal function only
#no examples
#
#
```

`prior_pattern` *An internal function to change the hyperprior parameters in the pattern mixture model provided by the user depending on the type of missingness mechanism and outcome distributions assumed*

Description

This function modifies default hyper prior parameter values in the type of pattern mixture model selected according to the type of missingness mechanism and distributions for the outcomes assumed.

Usage

```
prior_pattern(type, dist_e, dist_c, model_txt_info, model_string_jags)
```

Arguments

`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR). For a complete list of all available hyper parameters and types of models see the manual.
`dist_e`	distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern')
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm')

`model_txt_info`	list containing model specification information used to write the txt file of the JAGS model.
`model_string_jags`	text file of the model.

Examples

```
#Internal function only
#no examples
#
#
```

`prior_selection` *An internal function to change the hyperprior parameters in the selection model provided by the user depending on the type of missingness mechanism and outcome distributions assumed*

Description

This function modifies default hyper prior parameter values in the type of selection model selected according to the type of missingness mechanism and distributions for the outcomes assumed.

Usage

```
prior_selection(type, dist_e, dist_c, model_txt_info, model_string_jags)
```

Arguments

`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR). For a complete list of all available hyper parameters and types of models see the manual.
`dist_e`	distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern')
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm')
`model_txt_info`	list containing model specification information used to write the txt file of the JAGS model.
`model_string_jags`	text file of the model.

Examples

```
#Internal function only
#no examples
#
#
```

`run_hurdle` *An internal function to execute a JAGS hurdle model and get posterior results*

Description

This function fits a JAGS using the `jags` function and obtain posterior inferences.

Usage

```
run_hurdle(data_model, type, dist_e, dist_c, model_info)
```

Arguments

`data_model`	list containing the data for the model to be passed to JAGS.
`type`	Type of structural value mechanism assumed. Choices are Structural Completely At Random (SCAR), Structural At Random (SAR),
`dist_e`	distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern').
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm').
`model_info`	list containing model and MCMC information to be passed to JAGS.

Examples

```
#Internal function only
#No examples
#
#
```

---

run_lmdm *An internal function to execute a JAGS longitudinal missing data model and get posterior results*

Description

This function fits a JAGS using the `jags` function and obtain posterior inferences.

Usage

```
run_lmdm(data_model, type, dist_e, dist_c, model_info)
```

Arguments

`data_model`	list containing the data for the model to be passed to JAGS.
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR).
`dist_e`	distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern').
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm').
`model_info`	list containing model and MCMC information to be passed to JAGS.

Examples

```
#Internal function only
#No examples
#
#
```

---

run_pattern *An internal function to execute a JAGS pattern mixture model and get posterior results*

Description

This function fits a JAGS using the `jags` function and obtain posterior inferences.

Usage

```
run_pattern(data_model, type, dist_e, dist_c, model_info)
```

Arguments

`data_model`	list containing the data for the model to be passed to JAGS.
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR).
`dist_e`	distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern').
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm').
`model_info`	list containing model and MCMC information to be passed to JAGS.

Examples

```
#Internal function only
#No examples
#
#
```

`run_selection` *An internal function to execute a JAGS selection model and get posterior results*

Description

This function fits a JAGS using the `jags` function and obtain posterior inferences.

Usage

```
run_selection(data_model, type, dist_e, dist_c, model_info)
```

Arguments

`data_model`	list containing the data for the model to be passed to JAGS.
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR).
`dist_e`	distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern').
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm').
`model_info`	list containing model and MCMC information to be passed to JAGS.

Examples

```
#Internal function only
#No examples
#
#
```

`selection` *Full Bayesian Models to handle missingness in Economic Evaluations (Selection Models)*

Description

Full Bayesian cost-effectiveness models to handle missing data in the outcomes under different missing data mechanism assumptions, using alternative parametric distributions for the effect and cost variables and using a selection model approach to identify the model. The analysis is performed using the BUGS language, which is implemented in the software JAGS using the function `jags` The output is stored in an object of class 'missingHE'.

Usage

```
selection(
  data,
  model.eff,
  model.cost,
  model.me = me ~ 1,
  model.mc = mc ~ 1,
  dist_e,
  dist_c,
  type,
  prob = c(0.025, 0.975),
  n.chains = 2,
  n.iter = 10000,
  n.burnin = floor(n.iter/2),
  inits = NULL,
  n.thin = 1,
  save_model = FALSE,
  prior = "default",
  center = FALSE,
  ...
)
```

Arguments

`data`	A data frame in which to find the variables supplied in `model.eff`, `model.cost` (model formulas for effects and costs) and `model.me`, `model.mc` (model formulas for the missing effect and cost models). Among these, effectiveness, cost and treatment indicator (only two arms) variables must always be provided and named 'e', 'c' and 'trt', respectively.

`model.eff`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economic effectiveness outcome ('e') whose name must correspond to that used in `data`. Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "location" parameter of the distribution through a linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.cost`	A formula expression in conventional `R` linear modelling syntax. The response must be a health economic cost outcome ('c') whose name must correspond to that used in `data`. Any covariates in the model must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "location" parameter of the distribution through a linear model. A joint bivariate distribution for effects and costs can be specified by including 'e' on the right-hand side of the formula for the costs model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.me`	A formula expression in conventional `R` linear modelling syntax. The response must be indicated with the term 'me'(missing effects) and any covariates must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "probability" parameter for the missing effects through a logistic-linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`model.mc`	A formula expression in conventional `R` linear modelling syntax. The response must be indicated with the term 'mc'(missing costs) and any covariates must be provided on the right-hand side of the formula. If there are no covariates, `1` should be specified on the right hand side of the formula. By default, covariates are placed on the "probability" parameter for the missing costs through a logistic-linear model. Random effects can also be specified for each model parameter. See details for how these can be specified.
`dist_e`	Distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern').
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm').
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR) and Missing Not At Random (MNAR).
`prob`	A numeric vector of probabilities within the range (0, 1), representing the upper and lower CI sample quantiles to be calculated and returned for the imputed values.
`n.chains`	Number of chains.
`n.iter`	Number of iterations.
`n.burnin`	Number of warmup iterations.

`inits`	A list with elements equal to the number of chains selected; each element of the list is itself a list of starting values for the `JAGS` model, or a function creating (possibly random) initial values. If `inits` is `NULL`, `JAGS` will generate initial values for all the model parameters.
`n.thin`	Thinning interval.
`save_model`	Logical. If `save_model` is `TRUE`, a `txt` file containing the model code is printed in the current working directory.
`prior`	A list containing the hyperprior values provided by the user. Each element of this list must be a vector containing the user-provided prior distribution and parameter values and must be named with the name of the corresponding parameter. For example, the hyperprior values for the standard deviation parameter for the effects can be provided using the list `prior = list('sigma.prior.e' = c("unif", 0, 10))`. For more information about how to provide prior hypervalues for different types of parameters and models see details. If `prior` is set to 'default', the default values will be used.
`center`	Logical. If `center` is `TRUE`, all covariates in both the effect and cost models will be centred.
`...`	Additional arguments that can be provided by the user. Examples are the additional arguments that can be provided to the function `bcea` to summarise the health economic evaluation results.

Details

Depending on the distributions specified for the outcome variables in the arguments `dist_e` and `dist_c` and the type of missingness mechanism specified in the argument `type`, different selection models are built and run in the background by the function `selection`. These models consist in logistic regressions to estimate the probability of missingness in one or both outcomes. A simple example can be used to show how selection models are specified. Consider a data set comprising a response variable y and a set of covariates X_j, for $i = j, ..., J$. For each subject in the trial $i = 1, ..., n$ we define an indicator variable m_i taking value `1` if the i-th individual is associated with a missing value and `0` otherwise. This is modelled as:

$$m_i\ Bernoulli(\pi_i)$$

$$logit(\pi_i) = \sum \gamma_j X_j + \delta y_i$$

where

- π_i is the individual probability of a missing value in y.
- γ_j represents the impact on the probability of a missing value in y of the covariate X_j.
- δ represents the impact on the probability of a missing value in y of the outcome values.

When $\delta = 0$ the model assumes a 'MAR' mechanism, while when $\delta ! = 0$ the mechanism is 'MNAR'. For the parameters indexing the missingness model, the default prior distributions assumed are:

- $\gamma_j\ Normal(0, 0.01)$
- $\delta\ Normal(0, 1)$

When user-defined hyperprior values are supplied via the argument `prior` in the function `selection`, the elements of this list (see Arguments) must be vectors containing the user-provided distribution name and hyperprior values and must take specific names according to the parameters they are associated with. Specifically, the names for the parameters indexing the model which are accepted by **missingHE** are the following:

- auxiliary parameters σ: "sigma.prior.e"(effects) and/or "sigma.prior.c"(costs)
- covariate parameters α_j and β_j: "alpha.prior"(effects) and/or "beta.prior"(costs)
- covariate parameters in the missingness model γ_j (if covariate data provided): "gamma.prior.e"(effects) and/or "gamma.prior.c"(costs)
- mnar parameter δ: "delta.prior.e"(effects) and/or "delta.prior.c"(costs)

For simplicity, here we have assumed that the set of covariates X_j used in the models for the effects/costs and in the model of the missing effect/cost values is the same. However, it is possible to specify different sets of covariates for each model using the arguments in the function `selection` (see Arguments).

For each model, random effects can also be specified for each parameter by adding the term + (x | z) to each model formula, where x is the fixed regression coefficient for which also the random effects are desired and z is the clustering variable across which the random effects are specified (must be the name of a factor variable in the dataset). Multiple random effects can be specified using the notation + (x1 + x2 | site) for each covariate that was included in the fixed effects formula.

Value

An object of the class 'missingHE' containing the following elements

data_set A list containing the original data set provided in `data` (see Arguments). Additional information is also included about, among others, the number of observed and missing individuals, the total number of individuals by treatment arm and the indicator vectors for the missing values

model_output A list containing the output of a `JAGS` model generated from the functions `jags`, and the posterior samples for the main parameters of the model

cea A list containing the output of the economic evaluation performed using the function `bcea`

type A character variable that indicate which type of missing value mechanism used to run the model, either `MAR` or `MNAR` (see details)

data_format A character variable that indicates which type of analysis was conducted, either using a `wide` or `long` dataset

Author(s)

Andrea Gabrio

References

Daniels, MJ. Hogan, JW. *Missing Data in Longitudinal Studies: strategies for Bayesian modelling and sensitivity analysis*, CRC/Chapman Hall.

Baio, G.(2012). *Bayesian Methods in Health Economics*. CRC/Chapman Hall, London.

Gelman, A. Carlin, JB., Stern, HS. Rubin, DB.(2003). *Bayesian Data Analysis, 2nd edition*, CRC Press.

Plummer, M. *JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling.* (2003).

See Also

`jags`, `bcea`

Examples

```
# Quck example to run using subset of MenSS dataset
MenSS.subset <- MenSS[50:100, ]

# Run the model using the selection function assuming a SCAR mechanism
# Use only 100 iterations to run a quick check
model.selection <- selection(data = MenSS.subset, model.eff = e ~ trt, model.cost = c ~ trt,
   model.me = me ~ 1, model.mc = mc ~ 1, dist_e = "norm", dist_c = "norm",
   type = "MAR", n.chains = 2, n.iter = 100)

# Print the results of the JAGS model
print(model.selection)
#

# Use dic information criterion to assess model fit
pic.dic <- pic(model.selection, criterion = "dic", cases = "cc")
pic.dic
#

# Extract regression coefficient estimates
coef(model.selection)
#



# Assess model convergence using graphical tools
# Produce histograms of the posterior samples for the mean effects
diag.hist <- diagnostic(model.selection, type = "histogram", param = "mu.e")
#

# Compare observed effect data with imputations from the model
# using plots (posterior means and credible intervals)
p1 <- plot(model.selection, class = "scatter", outcome = "effects")
#

# Summarise the CEA information from the model
summary(model.selection)


# Further examples which take longer to run
model.selection <- selection(data = MenSS, model.eff = e ~ trt, model.cost = c ~ trt + e,
   model.se = me ~ age, model.mc = mc ~ age, dist_e = "norm", dist_c = "norm",
```

```
   type = "MAR", n.chains = 2, n.iter = 500)
#
# Print results for all imputed values
print(model.selection)

# Use looic to assess model fit
pic.looic <- pic(model.selection, criterion = "looic", cases = "cc")
pic.looic

# Show density plots for mean costs parameters
diag.den <- diagnostic(model.selection, type = "denplot", param = "mu.c")

# Plots of imputations for all data
p1 <- plot(model.selection, class = "scatter", outcome = "both")

# Summarise the CEA results
summary(model.selection)


#
#
```

`summary.missingHE` *Summary method for objects in the class* `missingHE`

Description

Produces a table printout with some summary results of the health economic evaluation probabilistic model run using the function `selection`, `pattern`, `hurdle` or `lmdm`.

Usage

```
## S3 method for class 'missingHE'
summary(object, incremental = FALSE, prob = c(0.025, 0.975), digits = 3, ...)
```

Arguments

`object`	A `missingHE` object containing the results of the Bayesian modelling and the economic evaluation
`incremental`	Logical. If `incremental` is `TRUE`, incremental CE results are printed.
`prob`	A numeric vector of probabilities within the range (0, 1), representing the upper and lower CI quantiles to be calculated and returned for the posterior estimates.
`digits`	Integer indicating the number of decimal places to be used for rounding (default = 3).
`...`	Additional arguments affecting the summary produced.

Value

Prints tables with information on the CE results based on a model fitted using the function `selection`, `pattern` or `hurdle`. Summary information on the main parameters of interests is provided.

Author(s)

Andrea Gabrio

References

Baio, G.(2012). *Bayesian Methods in Health Economcis*. CRC/Chapman Hall, London.

See Also

`selection` `pattern` `hurdle` `lmdm` `diagnostic` `plot.missingHE`

Examples

```
# For examples see the function selection, pattern or hurdle
#
#
```

`write_hurdle` *An internal function to select which type of hurdle model to execute. Alternatives vary depending on the type of distribution assumed for the effect and cost variables, type of structural value mechanism assumed and independence or joint modelling This function selects which type of model to execute.*

Description

An internal function to select which type of hurdle model to execute. Alternatives vary depending on the type of distribution assumed for the effect and cost variables, type of structural value mechanism assumed and independence or joint modelling This function selects which type of model to execute.

Usage

```
write_hurdle(dist_e, dist_c, type, model_txt_info)
```

Arguments

`dist_e`	Distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern')
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm')

`type`	Type of structural value mechanism assumed. Choices are Structural Completely At Random (SCAR) and Structural At Random (SAR).
`model_txt_info`	list containing model specification information used to write the txt file of the JAGS model.

Examples

```
#Internal function only
#No examples
#
#
```

`write_lmdm` *An internal function to select which type of longitudinal missing data model to execute. Alternatives vary depending on the type of distribution assumed for the effect and cost variables, type of missingness mechanism assumed and independence or joint modelling This function selects which type of model to execute.*

Description

An internal function to select which type of longitudinal missing data model to execute. Alternatives vary depending on the type of distribution assumed for the effect and cost variables, type of missingness mechanism assumed and independence or joint modelling This function selects which type of model to execute.

Usage

```
write_lmdm(dist_e, dist_c, type, model_txt_info)
```

Arguments

`dist_e`	Distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern')
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm')
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR)
`model_txt_info`	list containing model specification information used to write the txt file of the JAGS model.

Examples

```
#Internal function only
#No examples
#
#
```

`write_pattern` *An internal function to select which type of pattern mixture model to execute. Alternatives vary depending on the type of distribution assumed for the effect and cost variables, type of missingness mechanism assumed and independence or joint modelling This function selects which type of model to execute.*

Description

An internal function to select which type of pattern mixture model to execute. Alternatives vary depending on the type of distribution assumed for the effect and cost variables, type of missingness mechanism assumed and independence or joint modelling This function selects which type of model to execute.

Usage

```
write_pattern(dist_e, dist_c, type, model_txt_info)
```

Arguments

`dist_e`	Distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern')
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm')
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR)
`model_txt_info`	list containing model specification information used to write the txt file of the JAGS model.

Examples

```
#Internal function only
#No examples
#
#
```

`write_selection` *An internal function to select which type of selection model to execute. Alternatives vary depending on the type of distribution assumed for the effect and cost variables, type of missingness mechanism assumed and independence or joint modelling This function selects which type of model to execute.*

Description

An internal function to select which type of selection model to execute. Alternatives vary depending on the type of distribution assumed for the effect and cost variables, type of missingness mechanism assumed and independence or joint modelling This function selects which type of model to execute.

Usage

```
write_selection(dist_e, dist_c, type, model_txt_info)
```

Arguments

`dist_e`	Distribution assumed for the effects. Current available choices are: Normal ('norm'), Beta ('beta'), Gamma ('gamma'), Exponential ('exp'), Weibull ('weib'), Logistic ('logis'), Poisson ('pois'), Negative Binomial ('negbin') or Bernoulli ('bern')
`dist_c`	Distribution assumed for the costs. Current available choices are: Normal ('norm'), Gamma ('gamma') or LogNormal ('lnorm')
`type`	Type of missingness mechanism assumed. Choices are Missing At Random (MAR), Missing Not At Random for the effects (MNAR_eff), Missing Not At Random for the costs (MNAR_cost), and Missing Not At Random for both (MNAR)
`model_txt_info`	list containing model specification information used to write the txt file of the JAGS model.

Examples

```
#Internal function only
#No examples
#
#
```

Index