Package 'gFormulaMI'

July 3, 2025

Title G-Formula for Causal Inference via Multiple Imputation

Version 1.0.2

Description Implements the G-Formula method for causal inference with time-varying treatments and confounders using Bayesian multiple imputation methods, as described by Bartlett et al (2025) <doi:10.1177/09622802251316971>. It creates multiple synthetic imputed datasets under treatment regimes of interest using the 'mice' package. These can then be analysed using rules developed for analysing multiple synthetic datasets.

License GPL (>= 3)

Encoding UTF-8

Imports mice

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 2.10)

LazyData true

VignetteBuilder knitr

URL https://jwb133.github.io/gFormulaMI/

NeedsCompilation no

Author Jonathan Bartlett [aut, cre] (ORCID: <https://orcid.org/0000-0001-7117-0195>), Camila Olarte Parra [ctb] (ORCID: <https://orcid.org/0000-0003-0263-4392>), Erik van Zwet [ctb]

Maintainer Jonathan Bartlett <jonathan.bartlett1@lshtm.ac.uk>

Repository CRAN

Date/Publication 2025-07-03 14:30:10 UTC

Contents

`gFormulaImpute` *G-formula multiple imputation*

Description

`gFormulaImpute` creates multiple imputed synthetic datasets of longitudinal histories under specified treatment regimes of interest, based on the G-formula.

Usage

```
gFormulaImpute(
  data,
  M = 50,
  trtVars,
  trtRegimes,
  nSim = NULL,
  micePrintFlag = FALSE,
  silent = FALSE,
  method = NULL,
  predictorMatrix = NULL,
  missingDataCheck = TRUE
)
```

Arguments

`data`	The observed data frame
`M`	The number of imputed datasets to generate
`trtVars`	A vector of variable names indicating the time-varying treatment variables
`trtRegimes`	A vector specifying the treatment regime of interest, or a list of vectors specifying the treatment regimes of interest
`nSim`	The number of individuals to simulate in each imputed dataset. Defaults to number of individuals in observed data
`micePrintFlag`	TRUE/FALSE specifying whether the output from the call(s) to mice should be printed
`silent`	TRUE/FALSE indicating whether to print output to console (FALSE) or not (TRUE)
`method`	An optional method argument to pass to mice. If specified, this should be a named vector of methods, indicating which imputation method to use for each time-varying confounder variable and final outcome. If not specified, the default is to impute continuous variables using normal linear regression (norm), binary variables using logistic regression (logreg), polytomous regression for unordered factors and proportional odds model for ordered factors

`predictorMatrix`
: An optional predictor matrix to specify which variables to use as predictors in the imputation models. The default is to impute sequentially, i.e. impute using all variables to the left of the variable being imputed as covariates

`missingDataCheck`
: TRUE/FALSE indicating whether `gFormulaMI` checks, when passed a regular data frame, whether there any missing values.

Details

`gFormulaImpute` creates multiple imputed synthetic datasets of longitudinal histories under specified treatment regimes of interest, based on the G-formula, as described by Bartlett et al (2025) doi:10.1177/09622802251316971. Specifically, to the observed data frame, an additional `nSim` rows are added in which all variables are set to missing, except the time-varying treatment variables. The latter are set to the values as specified in the `trtRegimes` argument. If multiple treatment regimes are specified, `nSim` rows are added for each of the specified treatment regimes.

`gFormulaImpute` uses the `mice` package to impute the potential outcome values of the time-varying confounders and outcome in the synthetic datasets. Imputation is performed sequentially from left to right in the data frame. As such, the variables must be ordered in time in the input data frame, with the time-varying confounders at each time followed by the corresponding treatment variable at that time.

For the data argument, `gFormulaImpute` expects either a fully observed (complete) data frame, or else a set of multiple imputation stored in an object of class `mids` (from the `mice` package).

Unlike with Rubin's regular multiple imputation pooling rules, it is possible for the pooling rules developed by Raghunathan et al (2003) to give negative variance estimates. The probability of this occurring is reduced by increasing `M` and/or `nSim`.

`gFormulaImpute` returns an object of class `mids`. This can be analysed using the same methods that imputed datasets from `mice` can be analysed with (see examples). However, Rubin's standard pooling rules are not valid for analysis of the synthetic datasets. Instead, the synthetic variance estimator of Raghunathan et al (2003) must be used, as implemented in the syntheticPool function.

The development of the `gFormulaMI` package was supported by a grant from the UK Medical Research Council (MR/T023953/1).

Value

an S3 object of class mids (multiply imputed dataset)

Author(s)

Jonathan Bartlett <jonathan.bartlett1@lshtm.ac.uk>

References

Bartlett JW, Olarte Parra C, Granger E, Keogh RH., van Zwet EW and Daniel RM, 2025. G-formula with multiple imputation for causal inference with incomplete data. Statistical Methods in Medical Research.

Raghunathan TE, Reiter JP, Rubin DB. 2003. Multiple imputation for statistical disclosure limitation. Journal of Official Statistics, 19(1), p.1-16.

Examples

```
set.seed(7626)
#impute synthetic datasets under two regimes of interest
imps <- gFormulaImpute(data=simDataFullyObs,M=10,
                       trtVars=c("a0","a1","a2"),
                       trtRegimes=list(c(0,0,0),c(1,1,1)))
#fit linear model to final outcome with regime as covariate
fits <- with(imps, lm(y~factor(regime)))
#pool results using Raghunathan et al 2003 rules
syntheticPool(fits)
```

---

simDataFullyObs *Simulated fully observed data frame*

Description

A simulated observational study data frame with no missing data.

Usage

```
simDataFullyObs
```

Format

`simDataFullyObs`**:**

A data frame with 5000 rows and the following variables:

l0 Continuous baseline confounder

a0 Binary baseline treatment

l1 Continuous confounder at time 1

a1 Binary treatment at time 1

l2 Continuous confounder at time 2

a2 Binary treatment at time 2

y Continuous final outcome ...

`syntheticPool` *Pool estimates and variances obtained by analysing multiple synthetic datasets*

Description

This function pools estimates and variances which have been obtained by analysing multiple synthetic imputations (e.g. created used gFormulaImpute) using the method developed by Raghunathan et al 2003.

Usage

```
syntheticPool(fits)
```

Arguments

`fits`	Collection of model fits produced by a call of the form `with(imps, lm(y~regime))` where `imps` is a collection of imputed datasets of class `mids`.

Details

The only argument to `syntheticPool` is a set of model fits obtained by running an analysis on an imputed dataset collection of class `mids`, as created for example using the `mice` function in the `mice` package.

The function returns a table containing the overall parameter estimates, the within, between and total imputation variances, 95% confidence intervals, and p-values testing the null hypothesis that the corresponding parameters equal zero.

It is possible for the variance estimator developed by Raghunathan et al 2003 to be negative. In this case `syntheticPool` stops and informs you to re-impute using a larger number of imputations `M` and/or `nSim`.

The development of the `gFormulaMI` package was supported by a grant from the UK Medical Research Council (MR/T023953/1).

Value

A matrix containing the pooled results.

Author(s)

Jonathan Bartlett <jonathan.bartlett1@lshtm.ac.uk>

References

Raghunathan TE, Reiter JP, Rubin DB. 2003. Multiple imputation for statistical disclosure limitation. Journal of Official Statistics, 19(1), p.1-16.

Examples

```
set.seed(7626)
#impute synthetic datasets under two regimes of interest using gFormulaImpute
imps <- gFormulaImpute(data=simDataFullyObs,M=10,
                       trtVars=c("a0","a1","a2"),
                       trtRegimes=list(c(0,0,0),c(1,1,1)))
#fit linear model to final outcome with regime as covariate
fits <- with(imps, lm(y~factor(regime)))
#pool results using Raghunathan et al 2003 rules
syntheticPool(fits)
```

Index