Package 'StackImpute'

May 7, 2026

Title Tools for Analysis of Stacked Multiple Imputations

Version 0.1.0

Description Provides methods for inference using stacked multiple imputations augmented with weights. The vignette provides example R code for implementation in general multiple imputation settings. For additional details about the estimation algorithm, we refer the reader to Beesley, Lauren J and Taylor, Jeremy M G (2020) "A stacked approach for chained equations multiple imputation incorporating the substantive model" <doi:10.1111/biom.13372>, and Beesley, Lauren J and Taylor, Jeremy M G (2021) "Accounting for not-at-random missingness through imputation stacking" <doi:10.48550/arXiv.2101.07954>.

Depends R (>= 3.6.0)

License GPL-2

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.1.1

Imports sandwich, zoo, mice, dplyr, MASS, magrittr, boot

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Lauren Beesley [aut], Mike Kleinsasser [cre]

Maintainer Mike Kleinsasser <mkleinsa@umich.edu>

Repository CRAN

Date/Publication 2021-09-10 11:10:02 UTC

Contents

`Bootstrap_Variance` *Bootstrap_Variance*

Description

This function takes a dataset with stacked multiple imputation and a model fit and applies bootstrap to estimate the covariance matrix accounting for imputation uncertainty.

Usage

```
Bootstrap_Variance(fit, stack, M, n_boot = 100)
```

Arguments

`fit`	object with corresponding vcov method (e.g. glm, coxph, survreg, etc.) from fitting to the (weighted) stacked dataset
`stack`	data frame containing stacked dataset across multiple imputations. Could have 1 or M rows for each subject with complete data. Should have M rows for each subject with imputed data. Must contain the following named columns: (1) stack$.id, which correspond to a unique identifier for each subject. This column can be easily output from MICE. (2) stack$wt, which corresponds to weights assigned to each row. Standard analysis of stacked multiple imputations should set these weights to 1 over the number of times the subject appears in the stack. (3) stack$.imp, which indicates the multiply imputed dataset (from 1 to M). This column can be easily output from MICE.
`M`	number of multiple imputations
`n_boot`	number of bootstrap samples

Details

This function implements the bootstrap-based estimation method for stacked multiple imputations proposed by Dr. Paul Bernhardt in “A Comparison of Stacked and Pooled Multiple Imputation" at the Joint Statistical Meetings, 2019.

Value

Variance, estimated covariance matrix accounting for within and between imputation variation

Examples

```
data(stackExample)

fit = stackExample$fit
stack = stackExample$stack

bootcovar = Bootstrap_Variance(fit, stack, M = 5, n_boot = 10)
VARIANCE_boot = diag(bootcovar)
```

`func.boot` *func.boot*

Description

This function is called internal to Bootstrap_Variance and re-estimates glm model parameters

Usage

```
func.boot(data, indices)
```

Arguments

`data`	matrix with indices of possible imputed datasets to sample
`indices`	sampled indices

Value

numeric vector of parameter coefficients

`func.jack` *func.jack*

Description

This function is internal to Jackknife_Variance. This estimates model parameters using a subset of the stacked data.

Usage

```
func.jack(leaveout, stack)
```

Arguments

leaveout	indexes the multiple imputation being excluded from estimation
stack	data frame containing stacked dataset across multiple imputations. Could have 1 or M rows for each subject with complete data. Should have M rows for each subject with imputed data. Must contain the following named columns: (1) stack$.id, which correspond to a unique identifier for each subject. This column can be easily output from MICE. (2) stack$wt, which corresponds to weights assigned to each row. Standard analysis of stacked multiple imputations should set these weights to 1 over the number of times the subject appears in the stack. (3) stack$.imp, which indicates the multiply imputed dataset (from 1 to M). This column can be easily output from MICE.

Value

numeric vector of parameter coefficients

---

glm.weighted.dispersion *glm.weighted.dispersion*

Description

The goal of this function is to estimate the glm dispersion parameter using data across imputed datasets while correctly accounting for the weights.

Usage

```
glm.weighted.dispersion(fit)
```

Arguments

fit	an object of class glm

Value

an estimate of the glm dispersion parameter

Examples

```
data(stackExample)
glm.weighted.dispersion(stackExample$fit)
```

Jackknife_Variance *Jackknife_Variance*

Description

This function takes a dataset with stacked multiple imputation and a model fit and applies jackknife to estimate the covariance matrix accounting for imputation uncertainty.

Usage

```
Jackknife_Variance(fit, stack, M)
```

Arguments

fit	object with corresponding vcov method (e.g. glm, coxph, survreg, etc.) from fitting to the (weighted) stacked dataset
stack	data frame containing stacked dataset across multiple imputations. Could have 1 or M rows for each subject with complete data. Should have M rows for each subject with imputed data. Must contain the following named columns: (1) stack$.id, which correspond to a unique identifier for each subject. This column can be easily output from MICE. (2) stack$wt, which corresponds to weights assigned to each row. Standard analysis of stacked multiple imputations should set these weights to 1 over the number of times the subject appears in the stack. (3) stack$.imp, which indicates the multiply imputed dataset (from 1 to M). This column can be easily output from MICE.
M	number of multiple imputations

Details

This function implements the jackknife-based estimation method for stacked multiple imputations proposed by Beesley and Taylor (2021).

Value

Variance, estimated covariance matrix accounting for within and between imputation variation

Examples

```
data(stackExample)

fit = stackExample$fit
stack = stackExample$stack

jackcovar = Jackknife_Variance(fit, stack, M = 5)
VARIANCE_jack = diag(jackcovar)
```

Louis_Information *Louis_Information*

Description

This function takes a dataset with stacked multiple imputations and a glm or coxph fit and estimates the corresponding information matrix accounting for the imputation uncertainty.

Usage

```
Louis_Information(fit, stack, M, IMPUTED = NULL)
```

Arguments

fit	object of class glm or coxph from fitting to the (weighted) stacked dataset
stack	data frame containing stacked dataset across multiple imputations. Could have 1 or M rows for each subject with complete data. Should have M rows for each subject with imputed data. Must contain the following named columns: (1) stack$.id, which correspond to a unique identifier for each subject. This column can be easily output from MICE. (2) stack$wt, which corresponds to weights assigned to each row. Standard analysis of stacked multiple imputations should set these weights to 1 over the number of times the subject appears in the stack.
M	number of multiple imputations
IMPUTED	deprecated parameter, not used in current version

Details

This function uses the observed information matrix principle proposed in Louis (1982) and applied to imputations in Wei and Tanner (1990). This estimator is a further extension specifically designed for analyzing stacks of multiply imputed data as proposed in Beesley and Taylor (2019) https://arxiv.org/abs/1910.04625.

Value

Info, estimated information matrix accounting for within and between imputation variation

Examples

```
data(stackExample)
Info = Louis_Information(stackExample$fit, stackExample$stack, M = 50)
VARIANCE = diag(solve(Info))
```

Louis_Information_Custom

Louis_Information_Custom

Description

This function takes a dataset with stacked multiple imputations and a score matrix and covariance matrix from stacked and weighted analysis as inputs to estimates the corresponding information matrix accounting for the imputation uncertainty.

Usage

```
Louis_Information_Custom(score, covariance_weighted, stack, M)
```

Arguments

score	n x p matrix containing the contribution to the outcome model score matrix for each subject (n rows) and each model parameter (p columns).
covariance_weighted	p x p matrix containing the estimated covariance matrix from fitting the desired model to the stacked and weighted multiple imputations. Note: For GLM models, use summary(fit)$cov.unscaled*StackImpute::glm.weighted.dispersion(fit) as the default dispersion parameter will be incorrect.
stack	data frame containing stacked dataset across multiple imputations. Could have 1 or M rows for each subject with complete data. Should have M rows for each subject with imputed data. Must contain the following named columns: (1) stack$.id, which correspond to a unique identifier for each subject. This column can be easily output from MICE. (2) stack$wt, which corresponds to weights assigned to each row. Standard analysis of stacked multiple imputations should set these weights to 1 over the number of times the subject appears in the stack.
M	number of multiple imputations

Details

This function uses the observed information matrix principle proposed in Louis (1982) and applied to imputations in Wei and Tanner (1990). This estimator is a further extension specifically designed for analyzing stacks of multiply imputed data as proposed in Beesley and Taylor (2019) https://arxiv.org/abs/1910.04625.

Value

Info, estimated information matrix accounting for within and between imputation variation

Examples

```
data(stackExample)

fit = stackExample$fit
stack = stackExample$stack

covariates = as.matrix(cbind(1, stack$X, stack$B))
score = sweep(covariates, 1, stack$Y - covariates %*%
         matrix(coef(fit)), '*') / glm.weighted.dispersion(fit)
covariance_weighted = summary(fit)$cov.unscaled * glm.weighted.dispersion(fit)
Info = Louis_Information_Custom(score, covariance_weighted, stack, M = 50)
VARIANCE_custom = diag(solve(Info))
```

`my_update` *my_update*

Description

Function for updating a model fit using either new data or a new model structure

Usage

```
my_update(mod, formula = NULL, data = NULL, weights = NULL)
```

Arguments

`mod`	object of class 'glm' or 'coxph'
`formula`	formula for updated model fit, default = no change
`data`	data used for updated model fit, default = no change
`weights`	weights used for updated model fit, default = no change

Value

the updated model fit object of the same class as the given model

`stackExample` *Example data for Louis_Information()*

Description

Example data set for Louis_Information()

Format

a list with

- fit glm fit from vignette example
- stack stacked imputed data sets from vignette example

Index