Package ‘mvnimpute’

October 13, 2022

Type Package

Title Simultaneously Impute the Missing and Censored Values

Version 1.0.1

Depends R (>= 3.4.0)

Author Hesen Li

Maintainer Hesen Li <li.hesen.21@gmail.com>

URL https://github.com/hli226/mvnimpute

BugReports https://github.com/hli226/mvnimpute/issues

Description Implementing a multiple imputation algorithm for multivariate data with missing and censored values under a coarsening at random assumption (Heitjan and Rubin, 1991<doi:10.1214/aos/1176348396>). The multiple imputation algorithm is based on the data augmentation algorithm proposed by Tanner and Wong (1987)<doi:10.1080/01621459.1987.10478458>. The Gibbs sampling algorithm is adopted to to update the model parameters and draw imputations of the coarse data.

License GPL-2 | GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

LinkingTo Rcpp, RcppArmadillo, RcppDist

Imports ggplot2, reshape2, LaplacesDemon, rlang, Rcpp, MASS, truncnorm

Suggests mice, clusterGeneration

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-07-06 09:40:02 UTC

R topics documented:

mvnimpute-package *mvnimpute: Multiple imputation for multivariate data with missing and censored values*

Description

The mvnimpute package implements multiple imputation for simultaneously imputing missing and censored values based on the joint normal model assumption.

Author(s)

Hesen Li

acf.calc *Autocorrelation function*

Description

Calculates the autocorrelation function and draws the plots.

Usage

```
acf.calc(data.mat, lag = 50, plot = TRUE, title = NULL, details = FALSE)
```

Arguments

`data.mat`	matrix including the variables of which autocorrelations are calculated.
`lag`	lag at which the autocorrelation is calculated, default is set as 50.
`plot`	logical variable to specify whether the plot is generated, default is set to TRUE.
`title`	title of each generated autocorrelation plot.
`details`	boolean variable to specify whether the autocorrelation values are returned, default is set to FALSE.

Details

This function calculates the autocorrelations of all the variables on a column by column base. The default value of `lag` is set as 50, the maximum number of lag should not exceed the number of rows of the dataset, which reflects the corresponding number of iteration of running the multiple imputation.

Value

If `details` = TRUE, a matrix containing the calculated autocorrelations of all the variables in the dataset will be returned. If `plot` = TRUE, the autocorrelation plots of all the variables will be drawn.

Examples

```
### generate some data
dat <- MASS::mvrnorm(n = 1000, mu = c(1, 2, 3, 4), Sigma = diag(4))

### ACF plots
acf.calc(data.mat = dat, title = paste0("Var ", 1:nrow(dat)))
```

`avg.plot` *Averaged simulated values plot function*

Description

Calculates the average simulated values of all parameters and generates plots.

Usage

```
avg.plot(
  data.mat,
  start,
  end,
  x.lab = "Iteration number",
  y.lab = "Average of simulated values",
  title = NULL,
  details = FALSE
)
```

Arguments

`data.mat`	data matrix including the simulated values for plot.
`start`	the number of cycle to start.
`end`	the number of cycle to end.
`x.lab`	label of the x axis in the generated plot, default is set to "Iteration number".

`y.lab` label of the y axis in the generated plot, default is set to "Average of simulated values".

`title` title of each generated plot.

`details` logical variable to specify whether the average simulated values are returned, default is set to FALSE.

Details

This function calculates the average simulated values across simulations. `iter` can be any number of iterations you want to draw, the corresponding number of rows of the data should be `iter` + 1.

Value

The plot of averaged values across iterations. If `details` = TRUE, a matrix containing the averaged values of all the variables across iterations will be returned.

Examples

```
### generate some normal data
dat <- MASS::mvrnorm(n = 1000, mu = c(1, 2, 3, 4), Sigma = diag(4))

### set column names
colnames(dat) <- paste0("Var ", 1:ncol(dat))

### average values plot: take sample from 500 to 1000 rows
avg.plot(data.mat = dat[500:1000, ], start = 500, end = 1000, title = "Random Variables")
```

`conv.plot` *Convergence plot function*

Description

Draws convergence plot for the simulated parameter values of all variables.

Usage

```
conv.plot(
  data.mat,
  start,
  end,
  x.lab = "Iteration number",
  y.lab = "Simulated values",
  title = NULL
)
```

Arguments

`data.mat` data matrix including the simulated values.

`start` the number of cycle to start.

`end` the number of cycle to end.

`x.lab` label of the x axis in the generated plot, default is set to "Iteration number".

`y.lab` label of the y axis in the generated plot, default is set to "Simulated values".

`title` title of each generated plot.

Details

The function generates the trace plot of simulated values across iterations. `iter` can be any number of iterations you want to draw, the corresponding number of rows of the data is `iter` + 1.

Value

The plot of simulated values across iterations.

Examples

```
### generate some data
dat <- MASS::mvrnorm(n = 1000, mu = c(1, 2, 3, 4), Sigma = diag(4))

### set column names
colnames(dat) <- paste0("Var ", 1:ncol(dat))

### convergence plot: select samples from 500 to 1000 rows
conv.plot(data.mat = dat[500:1000, ], start = 500, end = 1000, title = "Random Variables")
```

`data.generation` *Data generation function*

Description

Simulates multivariate normal data with missing and censored values. In this function, missing values will be generated first in the multivariate data, then censored values will be generated for the non-missing data.

Usage

```
data.generation(
  num_ind = 2000,
  mean_vec = rnorm(5),
  cov_mat = diag(5),
  miss_var = c(2, 3),
  miss_mech = "MCAR",
  miss_prob = c(0.2, 0.4),
  censor_var = 4,
  censor_type = "interval",
  censor_param = 0.1
)
```

Arguments

`num_ind` number of subjects.

`mean_vec` mean vectors.

`cov_mat` covariance matrix.

`miss_var` variables that have missing values.

`miss_mech` missing mechanism. "MCAR" or "MAR". Default "MCAR".

`miss_prob` missing data probability when missing data is MCAR.

`censor_var` variables that have censored values.

`censor_type` type of censoring. "interval", "right" or "left. Default "interval".

`censor_param` rate parameter of the exponential distribution that the censoring times come from.

Value

A list containing the fully observed data, the observed data, the bounds information of the observed data and the data type indicator matrix.

Examples

```
### generate a multivariate normal dataset of 2000 sample size
### using the default arguments
data.generation()
```

marg.plot *Marginal density plots function*

Description

Draws marginal density plots for all variables

Usage

```
marg.plot(data.mat, title = NULL)
```

Arguments

`data.mat` data matrix including all the variables.

`title` title of each generated plot.

Value

Marginal density plot for each variable in the dataset.

Examples

```
### generate some data
dat <- MASS::mvrnorm(n = 1000, mu = c(1, 2, 3, 4), Sigma = diag(4))

### set column names
colnames(dat) <- paste0("Var ", 1:ncol(dat))

### marginal plots
marg.plot(data.mat = dat, title = paste0("Var", 1:nrow(dat)))
```

`multiple.imputation` *Multiple imputation function*

Description

Multiply imputes the missing and censored values in multivariate data.

Usage

```
multiple.imputation(data, prior.params, initial.values, iter, verbose = TRUE)
```

Arguments

`data`	a list of data containing the lower and upper bounds information for the missing and censored values.
`prior.params`	list of prior parameter specifications.
`initial.values`	list of initial values.
`iter`	number of rounds for doing multiple imputation.
`verbose`	boolean variable indicating whether the running status is printed in the console. Default is set to TRUE.

Details

A multivariate normal model is assumed on the data, the sweep operator is adopted to calculate the parameters of the conditional models. The implemented multiple imputation algorithm is based on the data augmentation algorithm proposed by Tanner and Wong (1987). The Gibbs sampling algorithm is adopted to update the model parameters and draw imputations of the coarse data. Output is a list including the parameters of the normal models and the imputed data across different iterations of multiple imputation.

Value

A list including the simulated mean and variance values of the assumed normal model, the covariance matrix, the imputed data, and the conditional model parameters across different iterations of multiple imputation.

References

Goodnight, J. H. (1979). A tutorial on the SWEEP operator. *The American Statistician*, **33(3)**, 149-158.

Tanner, M., & Wong, W. (1987). The Calculation of Posterior Distributions by Data Augmentation. *Journal of the American Statistical Association*, **82(398)**, 528-540.

Examples

```
## Not run:
### data and indicator
miss.dat <- simulated.dat[[1]]
data.ind <- simulated.dat[[2]]

### number of observations and variables
n <- nrow(miss.dat); p <- ncol(miss.dat)

#### bound matrices
b1 <- b2 <- matrix(nrow = nrow(data.ind), ncol = ncol(data.ind))

for (i in 1:nrow(b1)) {
  for (j in 1:ncol(b1)) {
    b1[i, j] <- ifelse(data.ind[i, j] != 1, NA,
                       miss.dat[i, j])
    b2[i, j] <- ifelse(data.ind[i, j] == 0, NA, miss.dat[i, j])
  }
}
colnames(b1) <- colnames(b2) <- colnames(miss.dat)

#### create a matrix for including the lower and upper bounds
bounds <- list()
bounds[[1]] <- b1; bounds[[2]] <- b2

### prior specifications
prior.param <- list(
  mu.0 = rep(0, p),
  Lambda.0 = diag(100, p),
  kappa.0 = 2,
  nu.0 = p * (p + 1) / 2
)

### starting values
start.vals <- list(
  mu = rep(0, p),
  sigma = diag(100, p)
)
```

```
### imputation
sim.res <- multiple.imputation(
   data = bounds,
   prior.params = prior.param,
   initial.values = start.vals,
   iter = 500,
   verbose = FALSE
)

## End(Not run)
```

`NHANES.dat` *Combined NHANES dataset from 1999-2004 NHANES study*

Description

A dataset including the age, gender and diastolic blood pressure, body mass index and 24 PCB measurements.

Usage

```
NHANES.dat
```

Format

A list including data frame with 5874 rows and 24 variables and associated indicator matrix:

BPXDAR Diastolic blood pressure

RIAGENDR Gender, 1 = male, 2 = female

RIDAGEYR Age in years

BMXBMI Body mass index

Details

The dataset is combined from the NHANES release cycles 1999-2000, 2001-2002, and 2003-2004. Almost all PCB have both the missing and censored values as falling below the limits of detection (LODs). The dataset include two components, the first component is the observed NHANES data where the censored PCB measurements are replaced by the LODs dividing the square root of 2. The second component is a data frame including the censoring indicators of the data, in that data frame, 0 indicates an observed PCB measurement, 1 indicates a censored PCB measurement, and ‘NA‘ indicates a missing PCB measurement.

Note

The subset provided here was selected to demonstrate the functionality of the mvnimpute package, no clinical conclusions should be derived from it.

Source

https://www.cdc.gov/nchs/nhanes/index.htm

simulated.dat *Simulated continuous data with missing and censored values*

Description

A dataset including simulated data with missing and censored values

Usage

```
simulated.dat
```

Format

A list including data matrix with 200 rows and 4 variables and associated indicator matrix:

y Outcome variable to be used in the regression model after imputation

x1 First covariate variable subject to MAR missing and non-informative censored values

x2 Second covariate variable subject to MAR missing and non-informative censored values

x3 Third covariate variable that is fully observed

Details

A simulated dataset and its associated indicator matrix are included into a list. In the indicator matrix, 0 stands for the missing values, 1 stands for the observed values, and 3 stands for the left censored values.

visual.plot *Draws percentage plot for different type of values*

Description

Draws plot that graphically shows the percentages of the missing, censored and observed data. It supports generating plots for all major types of censoring including left, right and interval censoring.

Usage

```
visual.plot(data.indicator, title = "Percentages of different data type")
```

Arguments

`data.indicator` matrix including the data type indicators of the original data.

`title` title of the generated plot, default is set to "Percentages of different data type".

Details

The function draws the plot that graphically shows the percentages of the missing, censored and observed data in the dataset. `data.indicator` should be a matrix containing the data type indicators as generated in the data preparation step. 0 for missing values, 1 for observed values, and 2 for right censored values, 3 for left censored values, and 4 for interval censored values. `title` is the title of the generated plot.

Value

The plot that shows the details of the different type of data in the dataset.

Examples

```
data.ind <- simulated.dat[[2]]
visual.plot(data.ind)
```

Index