Package 'mi'

October 2, 2014

Version 0.09-19

Date 2014-10-02

Title Missing Data Imputation and Model Checking

Author Andrew Gelman <gelman@stat.columbia.edu>,Jennifer Hill <jh1030@columbia.edu>,Yu-Sung Su <suyusung@tsinghua.edu.cn>,Masanao Yajima <my2167@columbia.edu>,Maria Grazia Pittau <grazia@stat.columbia.edu>

Maintainer Yu-Sung Su <suyusung@tsinghua.edu.cn>

Depends arm (>= 1.6-07)

Imports abind, car, foreign, lme4 (>= 0.999999-2), MASS, Matrix (>= 1.0), methods, nnet, stats, R2WinBUGS

Description Missing-data imputation and model checking

URL http://www.stat.columbia.edu/~gelman/

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2014-10-02 08:02:16

R topics documented:

`CHAIN` *Subset of variables from the CHAIN project, a longitudinal cohort study of people living with HIV in New York City.*

Description

The CHAIN cohort was recruited in 1994 from a large number of medical care and social service agencies serving HIV in New York City. Cohort members were interviewed up to 8 times through 2002. A total of 532 CHAIN participants completed at least one interview at either the 6th, 7th or 8th, and 508, 444, 388 interviews were completed respectively at rounds 6,7 and 8th.

Usage

```
data(CHAIN)
```

Format

A data frame with 532 observations on the following 8 variables.

`h39b.W1` log of self reported viral load level at round 6th (0 represents undetectable level).

`age.W1` age at time of interview at round 6th.

`c28.W1` family annual income. Values range from under \$5,000 to \$70,000 or over at round 6th.

`pcs.W1` a continuous scale of physical health with a theoretical range between 0 and 100 (better health is associated with higher scale values) at round 6th.

`mcs37.W1` a binary measure of poor mental health (1=Yes, 0=No) at round 6th.

`b05.W1` ordered interval for the CD4 count (indicator of how much damage HIV has caused to the immune system) at round 6th.

haartadhere.W1 a three-level ordered variable: 0=Not currently taking HAART (highly Active antiretroviral therapy), 1=taking HAART nonadherent, 2=taking HAART adheremt at round 6th.

Details

A missing value in the virus load level (h39b) was assigned to individuals who either could not recall their viral load level, did not have a viral load test in the six month preceding the interview, or reported their viral loads as "good" or "bad".

Source

http://cchps.columbia.edu/research.cfm

References

Messeri P, Lee G, Abramson DA, Aidala A, Chiasson MA, Jones JD. (2003). "Antiretroviral therapy and declining AIDS mortality in New York City". *Medical Care* 41:512–521.

See Also

mi

---

convergence.plot *Convergence Plot of mi Object*

Description

Function to plot trace of mi iterative samples.

Usage

```
convergence.plot ( mi.object, ... )
conv.plot ( mi.object, ... )
```

Arguments

mi.object	mi object generated from mi function
...	Other options for traceplot function.

Details

Convergence plot plots the convergence of the means and the standard deviations of each variable for the different imputations.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, M. Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Examples

```
# NOT RUN
#=======================================================
#  data(CHAIN)
#  CHAIN <- mi.preprocess(CHAIN)
#  imp.CHAIN <- mi(CHAIN, n.iter=6, add.noise=noise.control(post.run.iter=0))
#  convergence.plot(imp.CHAIN,mfrow=c(2,4))
#=======================================================
```

---

mi *Multiple Iterative Regression Imputation*

Description

Generate a multiply imputed matrix applying the elementary functions iteratively to the variables with missingness in the data randomly imputing each variable and looping through until approximate convergence.

Usage

```
## S4 method for signature 'data.frame'
mi(object, info,  n.imp = 3, n.iter = 30,
    R.hat = 1.1, max.minutes = 20, rand.imp.method = "bootstrap",
    run.past.convergence = FALSE,
    seed = NA, check.coef.convergence = FALSE,
    add.noise = noise.control())

## S4 method for signature 'mi.preprocessed'
mi(object, n.imp = 3, n.iter = 30,
    R.hat = 1.1, max.minutes = 20, rand.imp.method = "bootstrap",
    run.past.convergence = FALSE,
    seed = NA, check.coef.convergence = FALSE,
    add.noise = noise.control())


## S4 method for signature 'mi'
mi(object, n.iter = 30,
    R.hat = 1.1, max.minutes = 20, rand.imp.method = "bootstrap",
    run.past.convergence = FALSE,  seed = NA)
```

Arguments

object	A data frame or an mi object that contains an incomplete data. mi identifies NAs as the missing data.
info	The mi.info object.
n.imp	The number of multiple imputations. Default is 3 chains.
n.iter	The maximum number of imputation iterations. Default is 30 iterations.
R.hat	The value of the R.hat statistic used as a convergence criterion. Default is 1.1.
max.minutes	The maximum minutes to operate the whole imputation process. Default is 20 minutes.
rand.imp.method	The methods for random imputation. Currently, mi implements only the boostrap method.
run.past.convergence	Default is FALSE. If the value is set to be TRUE, mi will run until the values of either n.iter or max.minutes are reached even if the imputation is converged.
seed	The random number seed.
check.coef.convergence	Default is FALSE. If the value is set to be TRUE, mi will check the convergence of the coefficients of imputation models.
add.noise	A list of parameters for controlling the process of adding noise to mi via noise.control.

Details

Generate multiple imputations for incomplete data using iterative regression imputation. If the variables with missingness are a matrix Y with columns Y(1), . . . , Y(K) and the fully observed predictors are X, this entails first imputing all the missing Y values using some crude approach (for example, choosing imputed values for each variable by randomly selecting from the observed outcomes of that variable); and then imputing Y(1) given Y(2), . . . , Y(K) and X; imputing Y(2) given Y(1), Y(3), . . . , Y(K) and X (using the newly imputed values for Y(1)), and so forth, randomly imputing each variable and looping through until approximate convergence.

Value

A list of object of class mi, which stands for "multiple imputation".

Each object is itself a list of 10 elements.

call	Theimputation model.
data	The original data frame.
m	The number of imputations.
mi.info	Information matrix of the mi.
imp	A list of length(m) of imputations.
mcmc	A mcmc list that stores lists of means and sds of the imputed data.
converged	Binary variable to indicate if the mi has converged.

`coef.mcmc` A mcmc list that stores lists of means of regression coefficients of the conditonal models.

`coef.converged` Binary variable to indicate if the coefs of `mi` model have converged, return `NULL` if `check.coef.convergence = FALSE`

`preprocess` Binary variable to indicate if `preprocess=TRUE` in the `mi` process

`mi.info.preprocessed`
Information matrix that actually used in the `mi` if `preprocess=TRUE`.

Each `imp[[m]]` is itself a list containg k variable lists of 3 objects:

`imp[[m]][[k]]@model`
the specified models used for imputing missing values

`imp[[m]][[k]]@expected`
a list of vectors of length n-n.mis (number of complete observed data), specifying the estimated values of the models

`imp[[m]][[k]]@random`
a list of vectors of length n.mis (number of NAs), specifying the random predicted values for imputing missing data

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <suyusung@tsinghua.edu.cn>, M. Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Kobi Abayomi, Andrew Gelman and Marc Levy. (2008). "Diagnostics for multivariate imputations". *Applied Statistics* 57, Part 3: 273–291.

Andrew Gelman and Jennifer Hill. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

See Also

`mi.completed`, `mi.data.frame`, `mi.continuous`, `mi.binary`, `mi.count`, `mi.categorical`, `mi.polr`, `typecast`, `mi.info`, `mi.preprocess`

Examples

```
# simulate fake data
set.seed(100)
n <- 100
u1 <- rbinom(n, 1, .5)
v1 <- log(rnorm(n, 5, 1))
x1 <- u1*exp(v1)
u2 <- rbinom(n, 1, .5)
v2 <- log(rnorm(n, 5, 1))
x2 <- u2*exp(v2)
```

```
x3 <- rbinom(n, 1, prob=0.45)
x4 <- ordered(rep(seq(1, 5),100)[sample(1:n, n)])
x5 <- rep(letters[1:10],10)[sample(1:n, n)]
x6 <- trunc(runif(n, 1, 10))
x7 <- rnorm(n)
x8 <- factor(rep(seq(1,10),10)[sample(1:n, n)])
x9 <- runif(n, 0.1, .99)
x10 <- rpois(n, 4)
y <- x1 + x2 + x7 + x9 + rnorm(n)
fakedata <- cbind.data.frame(y, x1, x2, x3, x4, x5, x6, x7, x8, x9, x10)

# randomly create missing values
dat <- mi:::.create.missing(fakedata, pct.mis=30)

# get information matrix of the data
inf <- mi.info(dat)

# update the variable type of a specific variable to mi.info
inf <- update(inf, "type", list(x10="count"))

# run the imputation without data transformation
#IMP <- mi(dat, info=inf, check.coef.convergence=TRUE,
#  add.noise=noise.control(post.run.iter=10))

# run the imputation with data transformation
dat.transformed <- mi.preprocess(dat, inf)
#IMP <- mi(dat.transformed, n.iter=6, check.coef.convergence=TRUE,
#  add.noise=noise.control(post.run.iter=6))

IMP <- mi(dat.transformed, n.iter=6, add.noise=FALSE)


# no noise
# IMP <- mi(dat, info=inf, n.iter=6, add.noise=FALSE) ## NOT RUN

# pick up where you left off
# IMP <- mi(IMP, n.iter = 6)

## this is the suggested (defautl) way of running mi
# IMP <- mi(dat, info=inf) ## NOT RUN

# convergence checking
converged(IMP, check = "data")  ## You should get FALSE here because only n.iter is small
#converged(IMP, check = "coefs")
IMP.bugs1 <- bugs.mi(IMP, check = "data")    ## BUGS object to look at the R hat statistics
#IMP.bugs2 <- bugs.mi(IMP, check = "coefs")   ## BUGS object to look at the R hat statistics
plot(IMP.bugs1)  ## visually check R.hat

# visually check the imputation
plot(IMP)
```

`mi.binary` *Elementary function: Bayesian logistic regression to impute a binary variable.*

Description

Imputes univariate missing data using bayesglm, an R functions for generalized linear modeling with independent normal, t, or Cauchy prior distribution for the coefficients.

Usage

```
mi.binary(formula, data = NULL, start = NULL, n.iter = 100,
  draw.from.beta = TRUE, missing.index = NULL, ...)
## S4 method for signature 'mi.binary'
resid(object, y)
## S4 method for signature 'mi.binary'
residuals(object, y)
## S4 method for signature 'mi.binary,ANY'
plot( x, y, main=deparse( substitute( y ) ), gray.scale = FALSE, ...)
```

Arguments

`formula`	an object of class '"formula"' (or one that can be coerced to that class): a symbolic description of the model to be fitted. See bayesglm 'formula' for details.
`data`	A data frame containing the incomplete data and the matrix of the complete predictors.
`start`	Starting value for bayesglm.
`n.iter`	Maximum number of iteration for bayesglm. The default is 100.
`draw.from.beta`	Draws from posterior distribution of the betas to add randomness.
`missing.index`	The index of missing units of the outcome variable.
`...`	Currently not used.
`object`	`mi.binary` object.
`x`	`mi.binary` object.
`y`	Observed values.
`main`	main title of the plot.
`gray.scale`	When set to TRUE, makes the plot into gray scale with predefined color and line type.

Details

In bayesglm default the prior distribution is Cauchy with center 0 and scale 2.5 for all coefficients (except for the intercept, which has a prior scale of 10). See also glm for other details.

Value

model	A summary of the bayesian fitted model.
expected	The expected values estimated by the model.
random	Vector of length n.mis of random predicted values predicted by using the binomial distribution.

Note

see also http://www.stat.columbia.edu/~gelman/standardize/

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <suyusung@tsinghua.edu.cn>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Andrew Gelman and Jennifer Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Cambridge University Press, 2007.

See Also

mi.info, mi.method, mi

Examples

```
# true data
x <- rnorm(100,0,1) # N(0,1)
y <- rbinom(100,1,invlogit(1+2*x)) # y ~ Bin(1,invlogit(1 + 2*x)
# create artificial missingness on y
y[seq(1,100,10)]<-NA
dat.xy <- data.frame(x,y)
# imputation
mi.binary(y~x, data = dat.xy)
```

---

mi.categorical *Elementary function: multinomial log-linear models to impute a categorical variable.*

Description

Imputes missing data in a categorical variable using multinomial Log-linear Models.

Usage

```
  mi.categorical( formula, data = NULL, n.iter = 100,
                  MaxNWts = 1500, missing.index = NULL, ...)
  ## S4 method for signature 'mi.categorical'
residuals(object, y)
  ## S4 method for signature 'mi.categorical,ANY'
plot( x, y, main=deparse( substitute( y ) ), gray.scale = FALSE, ...)
```

Arguments

formula	a formula expression as for regression models, of the form `response ~ predictors`. The response should be a factor or a matrix with K columns, which will be interpreted as counts for each of K classes. A log-linear model is fitted, with coefficients zero for the first class. An offset can be included: it should be a numeric matrix with K columns if the response is either a matrix with K columns or a factor with K > 2 classes, or a numeric vector for a response factor with 2 levels. See the documentation of `formula()` for other details.
data	A data frame containing the incomplete data and the matrix of the complete predictors.
n.iter	Maximum number of iteration.
MaxNWts	The maximum allowable number of weights. See nnet for detail.
missing.index	The index of missing units of the outcome variable
object	`mi.categorical` object.
x	`mi.categorical` object.
y	Observed values.
main	main title of the plot.
gray.scale	When set to TRUE, makes the plot into gray scale with predefined color and line type.
...	Currently not used.

Details

`multinom` calls the library **nnet**. See `multinom` for other details.

Value

model	A summary of the multinomial fitted model.
expected	The expected values estimated by the model.
random	Vector of length n.mis of random predicted values predicted by using the multinomial distribution.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <suyusung@tsinghua.edu.cn>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Andrew Gelman and Jennifer Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Cambridge University Press, 2007.

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

See Also

mi.info, mi.method, mi

Examples

```
x <-rnorm(100,0,1)
y <- x+4
y <- round(y)
y[y<0] <- 0
# create artificial missingness on y
y[seq(1,100,10)] <- NA
dat.xy <- data.frame(x,y)
mi.categorical(formula = y ~ x, data = dat.xy)
```

---

mi.completed *Multiply Imputed Dataframes*

Description

Function to return completed data set from result of mi program.

Usage

```
## S4 method for signature 'mi'
mi.completed(object)
## S4 method for signature 'mi'
mi.data.frame(object, m = 1)
```

Arguments

object	mi object containing a multiply imputed data set. The mi object is generated by the mi function.
m	Index of the imputed data set. The default is 1.

Value

A data set or a list of datasets with the missing data imputed.

Author(s)

Yu-Sung Su <suyusung@tsinghua.edu.cn> Masanao Yajima <yajima@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. *Forthcoming*. "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software*.

Andrew Gelman and Jennifer Hill. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

See Also

mi

Examples

```
#data(CHAIN)
#IMP <- mi (CHAIN, n.iter=6, add.noise=FALSE)
### get all imputed dataset
#imputed.matrix <- mi.completed (IMP)
### get the 3rd chain of the imputed dataset
#imputed.data.frame <- mi.data.frame(IMP, m=3)
```

---

mi.continuous *Elementary function: linear regression to impute a continuous variable.*

Description

Imputes univariate missing data using linear regression.

Usage

```
mi.continuous(formula, data = NULL, start = NULL, n.iter = 100,
    draw.from.beta = TRUE, missing.index = NULL, ...)
```

Arguments

formula	an object of class '"formula"' (or one that can be coerced to that class): a symbolic description of the model to be fitted. See bayesglm 'formula' for details.
data	A data frame containing the incomplete data and the matrix of the complete predictors.
start	Starting value for bayesglm.
n.iter	Maximum number of iteration for bayesglm. The default is 100.
draw.from.beta	Draws from posterior distribution of the betas to add randomness.
missing.index	The index of missing units of the outcome variable
...	Currently not used.

Details

see bayesglm

Value

model	A summary of the fitted model.
expected	The expected values estimated by the model.
random	Vector of length n.mis of random predicted values predicted by using the normal distribution.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <suyusung@tsinghua.edu.cn>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Andrew Gelman and Jennifer Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Cambridge University Press, 2006.

See Also

mi.info, mi.method, mi

Examples

```
# true data
x<-rnorm(100,0,1) # N(0,1)
y<-rnorm(100,(1+2*x),1.2) # y ~ 1 + 2*x + N(0,1.2)
# create artificial missingness on y
y[seq(1,100,10)]<-NA
dat.xy <- data.frame(x,y)
# imputation
mi.continuous(y~x, data = dat.xy)
```

---

mi.count *Elementary function: Bayesian overdispersed poisson regression to impute a count variable.*

Description

Imputes univariate missing data using bayesglm, an R functions for generalized linear modeling with independent normal, t, or Cauchy prior distribution for the coefficients.

Usage

```
mi.count(formula, data = NULL, start = NULL, n.iter = 100,
  draw.from.beta = TRUE, missing.index = NULL, ...)
## S4 method for signature 'mi.count'
resid(object, y)
## S4 method for signature 'mi.count'
residuals(object, y)
```

Arguments

formula	an object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. See bayesglm formula for details.
data	A data frame containing the incomplete data and the matrix of the complete predictors.
start	Starting value for bayesglm.
n.iter	Maximum number of iteration for bayesglm. The default is 100.
draw.from.beta	Draws from posterior distribution of the betas to add randomness.
missing.index	The index of missing units of the outcome variable
...	Currently not used.
object	mi.countr object.
y	Observed values.

Details

In bayesglm default the prior distribution is Cauchy with center 0 and scale 2.5 for all coefficients (except for the intercept, which has a prior scale of 10). See also glm for other details.

Value

model	A summary of the bayesian fitted model.
expected	The expected values estimated by the model.
random	Vector of length n.mis of random predicted values predicted by using the binomial distribution.

Note

see also http://www.stat.columbia.edu/~gelman/standardize/

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <suyusung@tsinghua.edu.cn>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Andrew Gelman and Jennifer Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Cambridge University Press, 2007.

See Also

mi.info, mi.method, mi

Examples

```
# true data
x <- rnorm(100,0,1)
y <- rpois(100,40)
# create artificial missingness on y
y [seq(1,100,10)] <- NA
dat.xy <- data.frame(x,y)
# imputation
mi.count(y ~ x, data = dat.xy)
```

`mi.fixed` *Elementary function: imputation of constant variable.*

Description

Imputes univariate constant missing data.

Usage

```
mi.fixed( formula, data = NULL, missing.index = NULL, ... )
mi.copy(Y, X, missing.index = NULL, ...)
```

Arguments

`formula`	an object of class '"formula"' (or one that can be coerced to that class): a symbolic description of the model to be fitted. See bayesglm 'formula' for details.
`data`	A data frame containing the incomplete data and the matrix of the complete predictors.
`missing.index`	The index of missing units of the outcome variable
`Y`	A variable that is collinear with X.
`X`	A variable that is colliear with Y.
`...`	Currently not used

Value

model	A summary of the fitted model.
expected	The expected values estimated by the model.
random	Vector of length n.mis of random predicted values predicted by using the normal distribution.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Andrew Gelman and Jennifer Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Cambridge University Press, 2006.

See Also

mi.info, mi.method, mi

Examples

```
# fake data
n <- 100
x1 <- rbinom(n, 1, .45)
x2 <- 2*x1
x1[c(1, 3, 5, 20, 26)] <- NA

# impute data
mi.copy(x1, x2)
```

---

mi.hist *Multiple Imputation Histogram*

Description

A function for plotting the histogram of each variable and of its observed and imputed values.

Usage

```
  mi.hist( object, Yobs, ...)
## S4 method for signature 'mi.method,ANY'
mi.hist( object, Yobs, ...)
## S4 method for signature 'mi.categorical,ANY'
mi.hist( object, Yobs, ...)
## S4 method for signature 'mi.binary,ANY'
```

```
mi.hist( object, Yobs,...)
## S4 method for signature 'mi.polr,ANY'
mi.hist( object, Yobs, ...)
## S4 method for signature 'mi.pmm,ANY'
mi.hist( object, Yobs, ...)
```

Arguments

Yobs	observed values.
object	imputed values or member object of mi.method object family.
...	Other options for plot function.

Value

The histogram (in black) of the complete variable, the histogram (in blue) of the observed values and the histogram (in red) of the imputed values.

Note

The histogram of the completed values (observed plus imputed) is in black, the histogram of the imputed values in red, while the one of the observed values in blue.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <suyusung@tsinghua.edu.cn>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Kobi Abayomi, Andrew Gelman and Marc Levy. (2008). "Diagnostics for multivariate imputations". *Applied Statistics* 57, Part 3: 273–291.

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Andrew Gelman and Jennifer Hill. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

See Also

plot.mi, hist

Examples

```
# true data
x<-rnorm(100,0,1) # N(0,1)
y<-rnorm(100,(1+2*x),1.2) # y ~ 1 + 2*x + N(0,1.2)
# create artificial missingness on y
y[seq(2,100,10)]<-NA
dat.xy <- data.frame(x,y)
# imputation
```

```
dat.cont.mi <- mi.continuous(y~x, data = dat.xy)
mi.hist( dat.cont.mi, y)

# imputation
#dat.mi <- mi(dat.xy)
#mi.hist( imp(dat.mi,1)[["y"]], y)
```

`mi.info` *Function to create information matrix for missing data imputation*

Description

Produces matrix of information needed to impute the missing data. After the information is extracted user has the option of changing the default.

Usage

```
  mi.info(data, threshold  = 0.99999)
  ## S4 method for signature 'mi.info'
print(x, ...)
  ## S4 method for signature 'mi.info'
show(object)
```

Arguments

`data`	dataframe or matrix of dataset with missing data coded as NAs.
`threshold`	Threshold value for correlation to be considered a problem.
`x`	An object of a class `mi.info`.
`object`	An object of a class `mi.info`.
`...`	Currently not used.

Value

`info` information matrix

-name:	Name of variable
-imp.order:	Imputation Order
-nmis:	Number of missing
-type:	Type of variable
-var.class:	Class of input variable
-level:	Levels in the input varialbe
-include:	Include in the imputation process or not
-is.ID:	Is ID variable or not
-all.missing:	All observation missing or not
-collinear:	Collineared variables
-determ.pred:	Deterministic predictor
-imp.formula:	Imputation formula

-params:	Parameters for the imputation model
-other:	Currently not used

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

See Also

mi

Examples

```
data(CHAIN)
info.CHAIN <- mi.info(CHAIN)

info.CHAIN$imp.order # imputation order

info.CHAIN$imp.formula # imputation formula
info.CHAIN[["age.W1"]]$imp.formula  #imputation formula for specific variable
```

---

mi.info.update *function to update mi.info object to use for multiple imputation*

Description

This function is internal function to update the mi.info object.

Usage

```
## S3 method for class 'mi.info'
update(object, target, list, ...)
mi.info.update.type(object, list)
mi.info.update.level(object, list)
mi.info.update.include(object, list)
mi.info.update.is.ID(object, list)
mi.info.update.collinear(object, list)

mi.info.update.imp.order(object, list)
mi.info.update.determ.pred(object, list)
mi.info.update.params(object, list)
mi.info.update.imp.formula(object, list)
mi.info.update.other(object, list)
```

Arguments

object	mi.info object that is result of mi.info function.
target	which part of mi.info object to modify.
list	list that has same length as the number of variables in the mi.info object. Element which are left NULL will not be updated .
...	currently no function.

Value

info	updated mi.info object.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <suyusung@tsinghua.edu.cn>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Andrew Gelman and Jennifer Hill. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

See Also

mi.completed

Examples

```
data(CHAIN)
CHAIN.info <- mi.info(CHAIN)

# You can update the mi.info object in the below two ways
CHAIN.info <- update(CHAIN.info, "type",
  list(age.W1="continuous", b05.W1="unordered-categorical"))
CHAIN.info <- mi.info.update.type(CHAIN.info,
  list(age.W1="continuous", b05.W1="unordered-categorical"))
```

---

mi.method *Virtual class for all mi classes.*

Description

Imputes univariate missing data using linear regression.

Usage

```
  ## S4 method for signature 'mi.method'
imputed(object,y)
  ## S4 method for signature 'mi.categorical'
imputed(object,y)
  ## S4 method for signature 'mi.polr'
imputed(object,y)
  ## S4 method for signature 'mi.method'
coef(object)
  ## S4 method for signature 'mi.method'
coefficients(object)
  ## S4 method for signature 'mi.method'
sigma.hat(object)
  ## S4 method for signature 'mi.method'
fitted(object)
  ## S4 method for signature 'mi.method'
resid(object, y)
  ## S4 method for signature 'mi.method'
residuals(object, y)
  ## S4 method for signature 'mi.method'
print(x, ...)
  ## S4 method for signature 'mi.method,ANY'
plot(x, y, main=deparse( substitute( y ) ), gray.scale = FALSE, ...)
```

Arguments

object	mi.method object.
...	Currently not used.
x	mi.method object.
y	Observed values.
main	main title of the plot.
gray.scale	When set to TRUE, makes the plot into gray scale with predefined color and line type.

Details

mi.method is a virtual class for all the mi classes. Basically all the necessary functions are defined under mi.method class, thus most of the mi classes that do not have specific method defined for them inherits their methods from this class. For some special class as mi.nonnegative these methods are extended to tailor to the needs.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

See Also

mi

`mi.pmm` *Elementary function: Predictive Mean Matching for imputation.*

Description

Imputes univariate missing data using bayesglm and predictive mean matching.

Usage

```
mi.pmm(formula, data = NULL, start = NULL, n.iter = 100, missing.index = NULL, ...)
```

Arguments

`formula`	an object of class '"formula"' (or one that can be coerced to that class): a symbolic description of the model to be fitted. See bayesglm 'formula' for details.
`data`	A data frame containing the incomplete data and the matrix of the complete predictors.
`start`	Starting value for bayesglm.
`n.iter`	Maximum number of iteration for bayesglm. The default is 100.
`missing.index`	The index of missing units of the outcome variable
`...`	Currently not used.

Details

In bayesglm default the prior distribution is Cauchy with center 0 and scale 2.5 for all coefficients (except for the intercept, which has a prior scale of 10). See also glm for other details.

Value

`model`	A summary of the bayesian fitted model.
`expected`	The expected values estimated by the model.
`random`	Vector of length n.mis of random predicted values predicted by using the binomial distribution.

Note

see also http://www.stat.columbia.edu/~gelman/standardize/

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Andrew Gelman and Jennifer Hill, Data Analysis Using Regression and Multilevel/Hierarchical Models, Cambridge University Press, 2007.

Van Buuren, S. and Oudshoorn, C.G.M. (2000). Multivariate Imputation by Chained Equations: MICE V1.0 User's manual. Report PG/VGZ/00.038, TNO Prevention and Health, Leiden.

Rubin, D.B. (1987). Multiple imputation for nonresponse in surveys. New York: Wiley.

See Also

mi.info, mi.method, mi

`mi.polr` *Elementary function: multinomial log-linear models to impute a ordered categorical variable.*

Description

Imputes missing data in a categorical variable using multinomial Log-linear Models.

Usage

```
  mi.polr(formula, data = NULL, drop.unused.levels = TRUE, start = NULL,
    n.iter = 100, missing.index = NULL, ...)
  ## S4 method for signature 'mi.polr'
residuals(object, y)
  ## S4 method for signature 'mi.polr,ANY'
plot( x, y, main=deparse( substitute( y ) ), gray.scale = FALSE, ... )
```

Arguments

`formula`	a formula expression as for regression models, of the form `response ~ predictors`. The response should be a factor (preferably an ordered factor), which will be interpreted as an ordinal response, with levels ordered as in the factor. The model must have an intercept: attempts to remove one will lead to a warning and be ignored. An offset may be used. See the documentation of 'formula' for other details.
`data`	A data frame containing the incomplete data and the matrix of the complete predictors.
`drop.unused.levels`	Drops unused levels.
`start`	Starting value for bayespolr.

n.iter	Maximum number of iteration for bayespolr. The default is 100.
missing.index	The index of missing units of the outcome variable.
...	Currently not used.
object	mi.polr object.
x	mi.polr object.
y	Observed values.
main	main title of the plot.
gray.scale	When set to TRUE, makes the plot into gray scale with predefined color and line type.

Details

multinom calls the library **nnet**. See multinom for other details.

Value

model	A summary of the multinomial fitted model
expected	The expected values estimated by the model
random	Vector of length n.mis of random predicted values predicted by using the multinomial distribution
residual	The residual vector of length same as y

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <ys463@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Andrew Gelman and Jennifer Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Cambridge University Press, 2007.

See Also

mi.info, mi.method, mi

Examples

```
# true data
x<-rnorm(100,0,1) # N(0,1)
y<-(1+2*x)+rnorm(100,0,1)
y<-round(y)
y[y<0]<-0
# create artificial missingness on y
y[seq(1,100,10)]<-NA
```

```
dat.xy <- data.frame(x,y)
# imputation
polr.imp <- mi.polr(y~x, data = dat.xy)
```

`mi.pooled` *Modeling Functions for Multiply Imputed Dataset*

Description

Modeling Function that pulls together the estimates from multiply imputed dataset.

Usage

```
mi.pooled(coef, se)
lm.mi(formula, mi.object, ...)
glm.mi(formula, mi.object, family = gaussian, ...)
bayesglm.mi(formula, mi.object, family = gaussian, ...)
polr.mi(formula, mi.object, ...)
bayespolr.mi(formula, mi.object, ...)
lmer.mi(formula, mi.object, rescale=FALSE, ...)
glmer.mi(formula, mi.object, family = gaussian, rescale=FALSE, ...)
## S3 method for class 'mi.pooled'
print(x, ...)
## S4 method for signature 'mi.pooled'
coef(object)
## S4 method for signature 'mi.pooled'
se.coef(object)
## S4 method for signature 'mi.pooled'
display(object, digits=2)
```

Arguments

`coef`	list of coefficients
`se`	list of standard errors
`formula`	See `lm`, `glm`, `polr`, `lmer`for detail.
`mi.object`	`mi` object
`family`	See `glm`, `polr`, `lmer`for detail.
`rescale`	default is `FALSE`, see `rescale` for detail.
`x`	`mi.pooled` object.
`object`	`mi.pooled` object.
`digits`	number of significant digits to display, default=2.
`...`	Any option to pass on to `lm`, `glm`, `bayesglm`, `bayespolr`, `polr`, and `lmer` functions

Value

call	the matched call.
mi.pooled	pulled estimates from the multiple dataset.
mi.fit	estimates from each dataset.

Author(s)

Yu-Sung Su <suyusung@tsinghua.edu.cn>,

References

Andrew Gelman and Jennifer Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Cambridge University Press, 2007.

See Also

lm, glm, bayesglm, bayespolr, polr, and lmer

Examples

```
# true data
n <- 100
x <- rbinom(n,1,.45)
z <- ordered(rep(seq(1, 5),n)[sample(1:n, n)])
y <- rnorm(n)
group <- rep(1:10, 10)

# create artificial missingness
dat.xy <- data.frame(x, y, z)
dat.xy <- mi:::.create.missing(dat.xy, pct.mis=10)

# imputation
IMP <- mi(dat.xy, n.iter=6, add.noise=FALSE)

# fit models
M1 <- lm.mi(y ~ x + z, IMP)
display(M1)
coef(M1)
se.coef(M1)

M2 <- glm.mi(x ~ y , IMP, family = binomial(link="logit"))
display(M2)
coef(M2)
se.coef(M2)

M3 <- bayesglm.mi(x ~ y , IMP, family = binomial(link="logit"))
display(M3)
coef(M3)
se.coef(M3)

M4 <- polr.mi(ordered(z) ~  y, IMP)
```

```
display(M4)
coef(M4)
se.coef(M4)

M5 <- bayespolr.mi(ordered(z) ~  y, IMP)
display(M5)
coef(M5)
se.coef(M5)

M6 <- lmer.mi(y ~ x  + (1|group), IMP)
display(M6)
coef(M6)
se.coef(M6)

M7 <- glmer.mi(x ~ y  + (1|group), IMP, family = binomial(link="logit"))
display(M7)
coef(M7)
se.coef(M7)
```

`mi.preprocess` *Preproessing and Postprocessing mi data object*

Description

Function for propressing and postprocessing `nonnegative`, and `positive-continuous` variable types in mi data object

Usage

```
mi.preprocess(data, info)
mi.postprocess(mi.data, info)
```

Arguments

`data`	the data.frame to be imputed.
`info`	the information matrix, see `mi.info`.
`mi.data`	the imputed data list, obtained from `mi.completed`

Details

`mi.proprocess` will transform the `nonnegative` and `positive-continuous` variable types. If the variable is of `nonnegative` type, the function transforms the variable into two variables: an indicator indicates whether the value is postive or not and a transformed variable that takes on all positive value and is transformed either by taking a log; 0 and NA will be treated as missing for such a variable. If the variable is of `positive-continuous` type, it will be transformed by taking a log.

`mi.postprocess` will transform the imputed dataset back to its original form. The imputed dataset is obtained from `mi.completed` function.

Value

data	a data.frame or a list of dataframe
mi.info	a mi.info matrix

Author(s)

Yu-Sung Su <suyusung@tsinghua.edu.cn>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

See Also

mi.completed

---

mi.scatterplot *Multiple Imputation Scatterplot*

Description

A function for plotting observed and imputed values for a variable .

Usage

```
mi.scatterplot( Yobs, Yimp, X = NULL, xlab = NULL, ylab = NULL,
                            main = "Imputed Variable Scatter Plot",
                             display.zero = TRUE, gray.scale = FALSE,
                              obs.col = rgb( 0, 0, 1 ),
                              imp.col = rgb( 1, 0, 0 ),
                              obs.pch = 20 , imp.pch = 20,
                              obs.cex = 0.3, imp.cex = 0.3,
                              obs.lty = 1  , imp.lty = 1,
                              obs.lwd = 2.5, imp.lwd = 2.5, ... )
marginal.scatterplot ( data, object, use.imputed.X = FALSE, ...  )
```

Arguments

Yobs	observed values.
Yimp	imputed values.
X	variable to plot on the x axis.
xlab	label on the x axis.
ylab	label on the y axis.
display.zero	if set to FALSE zeros will not be displayed. Default is TRUE.

main	main title of the plot.
gray.scale	When set to TRUE, makes the plot into gray scale with predefined color and line type.
obs.col	color for the observed variable. Default is "blue".
imp.col	color for the imputed variable. Default is "red".
obs.pch	data symbol for observed variable. Default is 20.
imp.pch	data symbol for imputed variable. Default is 20.
obs.cex	text size for observed variable. Default is 0.3.
imp.cex	text size for imputed variable. Default is 0.3.
obs.lty	line type for observed variable. Default is 1.
imp.lty	line type for imputed variable. Default is 1.
obs.lwd	line width for observed variable. Default is 2.5.
imp.lwd	line width for imputed variable. Default is 2.5.
...	Other options for 'plot' function.
data	missing data.
object	mi object.
use.imputed.X	If you want to use the imputed X. Default is FALSE.

Details

Since several data points can have the same data values, especially in discrete variables, small random number is added to each value so that points do not fall on top of each other. See help on jitter for more details. Lowess line is fitted to both imputed and observed data.

Value

A scatterplot with the observed and the imputed values plotted against a chosen variable.

Note

By default imputed values are in red, while the observed values are in blue.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Kobi Abayomi, Andrew Gelman and Marc Levy. (2008). "Diagnostics for multivariate imputations". *Applied Statistics* 57, Part 3: 273–291.

Andrew Gelman and Jennifer Hill. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

See Also

mi, plot

Examples

```
# true data
x<-rnorm(100,0,1) # N(0,1)
y<-rnorm(100,(1+2*x),1.2) # y ~ 1 + 2*x + N(0,1.2)
# create artificial missingness on y
y[seq(1,100,10)]<-NA
dat.xy <- data.frame(x,y)
# imputation
imp.cont<-mi.continuous(y~x, data = dat.xy)
mi.scatterplot(y,imputed(imp.cont,y))
```

---

missing.pattern.plot *Missing Pattern Plot*

Description

Function to plot a missing pattern plot.

Usage

```
missing.pattern.plot ( data, y.order = FALSE, x.order = FALSE, clustered = TRUE,
                                    xlab = "Index", ylab = "Variable",
                                    main = NULL, gray.scale = FALSE,
                                    obs.col = "blue", mis.col = "red", ... )
```

Arguments

data	data.frame or matrix of data with missing data coded as "NA".
y.order	if TRUE, orders the variable by number of missing value. Default is FALSE.
x.order	if TRUE, orders the data by number of missing value. Default is FALSE.
clustered	if TRUE, data are grouped together with similiar missingness patterns.
xlab	a title for the x axis: see 'title'.
ylab	a title for the y axis: see 'title'.
main	an overall title for the plot: see 'title'.
gray.scale	if TRUE, makes the plot into black and white. This option overwrites the color specification.
obs.col	color used for observed values. Default is "blue".
mis.col	color used for missing values. Default is "red".
...	additional parameters passed to 'image' function.

Details

Color image with different color for missing and observed value in the dataset is plotted. By default the observed is in "blue" and missing is in "red".

Value

Plot to visualize pattern of missingness in the data.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. *Forthcoming*. "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software*.

Kobi Abayomi, Andrew Gelman and Marc Levy. (2008). "Diagnostics for multivariate imputations". *Applied Statistics* 57, Part 3: 273–291.

Andrew Gelman and Jennifer Hill. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

See Also

mi, plot

Examples

```
data(CHAIN)
missing.pattern.plot(CHAIN)
```

---

noise.control *Auxiliary for Adding Priors to Missing Data Imputation*

Description

Auxiliary function as user interface for adding noise for mi procedure. Typically only used when calling mi.

Usage

```
noise.control(method=c("reshuffling", "fading"), pct.aug=10, K=1, post.run.iter=20)
```

Arguments

method	two methods are implemented: reshuffling and fading.
pct.aug	percent of N being add into the exisitng data, where N is the number of observation of the completed data set, default is 10.
K	the cooling parameter, default is 1.
post.run.iter	number of iterations after a imputation, default is 20. This is to mitigate the influence of a imputation with the noise.

Details

If reshuffling method is used, the imputation will randomly switch between randomly imputing data from marginal (imputing data from the observed values) and drawing from the modeled based values. And with each iteration (s), the probability of cooling (q) decreases by number of iterations, such that q = K/s.

If fading method is used, the imputation will augment

Author(s)

Yu-Sung Su <yusung@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

See Also

mi

Examples

```
### NOT RUN
#============================================================
# add fadding empirical noise by augmenting 10% of the data
#============================================================
## data(CHAIN)
## IMP <- mi(CHAIN, add.noise=noise.control(method="fading", pct.aug=10, post.run.iter=20))
#================================================
# add noise by randomly drawing from the marginal
#================================================
## This is the default setting
## IMP <- mi(CHAIN, add.noise=noise.control(method="reshuffling", K=1, post.run.iter=20))
#================================================
# add no noise
#================================================
## IMP <- mi(CHAIN, add.noise=FALSE)
#================================================
# add noise but no post.run
#================================================
## IMP <- mi(CHAIN, add.noise=noise.control(post.run.iter=0))
```

plot.mi *Diagnostic Plots for multiple imputation object*

Description

Diagnostic plots for testing the fit of the imputation method to the observed data.

Usage

```
## S4 method for signature 'mi,ANY'
plot(x, y, ...)

plot.mi(x, y, m = 1, vrb = NULL, vrb.name = "Variable Score",
                    gray.scale = FALSE, mfrow=c( 1, 3 ), ... )
```

Arguments

x	mi object generated by the mi function.
y	currently not used.
...	Arguments for other methods, not used.
m	The m-th imputation. By default is 1.
vrb	A chosen variable for the scatter plot.
vrb.name	A name of the vrb variable.
gray.scale	When set to TRUE, makes the plot into gray scale with predefined color and line type.
mfrow	See "par" for details.

Details

For each variable, observed values are in blue, the imputed values are in red. In the scatterplot the observed and the imputed are plotted versus a variable the users can choose. By default the values are plotted against an index number but it strongly recommended to use a variable containing more information. Fitted lowess lines are also plotted for both observed and imputed data. A small amount of random noise (jittering) is added to the points so that they do not fall on top of each other.

Value

Histograms, scatterplots, and residual plots of the fit of the imputation models. Binned residual plots are for each binary variable.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, Yu-Sung Su <suyusung@tsinghua.edu.cn>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

Kobi Abayomi, Andrew Gelman and Marc Levy. (2008). "Diagnostics for multivariate imputations". *Applied Statistics* 57, Part 3: 273–291.

Andrew Gelman and Jennifer Hill. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

See Also

mi, mi.scatterplot, mi.hist

Examples

```
### NOT RUN
#=========================================================
#  data(CHAIN)
#  imp.CHAIN <- mi(CHAIN, n.iter=6, add.noise=FALSE)
#  plot(imp.CHAIN)
#=========================================================
```

---

random.imp *Random Imputation of Missing Data*

Description

Simple random imputation of missing values in given data set.

Usage

```
random.imp(data, imp.method = c( "bootstrap", "pca" ) , ...)
```

Arguments

data	A vector, matrix, or data frame with missing data.
imp.method	Character to specify which method of random imputation to use. Default is "bootstrap". Note: pca is not implemented in the current version.
...	Unused

Details

Impute missing values based on the observed data for the variable.

Value

Data with its missing values imputed using the specified method.

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Andrew Gelman and Jennifer Hill. (2006). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

See Also

mi

Examples

```
data(CHAIN)
data.imp <- random.imp(CHAIN)
```

---

type.models *Functions to identify types of the models of the mi object*

Description

The function to select the model based on the variable type

Usage

```
type.models(type)
mi.types()
```

Arguments

type	Nine types: continuous, log-continuous, count, ordered-categorical, unordered-categorical, binary, positive-continuous, proportion, predictive-mean-matching nonnegative, fixed

Author(s)

Yu-Sung Su <suyusung@tsinghua.edu.cn>, Masanao Yajima <yajima@stat.columbia.edu>, Andrew Gelman <gelman@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

See Also

mi.info

---

typecast *Variables type*

Description

Function for determinig the variable type.

Usage

```
  typecast( object )
  ## S4 method for signature 'ANY'
typecast( object )
  ## S4 method for signature 'matrix'
typecast( object )
  ## S4 method for signature 'data.frame'
typecast( object )
  ## S4 method for signature 'list'
typecast( object )
```

Arguments

object	Vector, matrix, or data frame of data to determin the type of.

Details

The variable type of a vector, or vector of variable types for each variable in the dataset.

Value

fixed	varaibles that contain only one value.
binary	variables that contain two values.
ordered-categorical	variables that contain 3 to 5 postive values.
unorderd-categorial	variables that contain characters or more than 5 postive levels
positive-continuous	variables that contain more than 5 postive values, NOT including 0s.
nonnegative	variables that contain more than 5 postive values, including 0s.
continuous	variables that are not belong to any of the above types.
log-continuous	log-scaled continuous variable

Author(s)

Yu-Sung Su <suyusung@tsinghua.edu.cn>, Masanao Yajima <yajima@stat.columbia.edu>, M.Grazia Pittau <grazia@stat.columbia.edu>

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

See Also

mi

Examples

```
data(CHAIN)
class(CHAIN)

typecast(CHAIN[,1]) # for vector

typecast(as.matrix(CHAIN))# for matrix

typecast(CHAIN) # for data.frame
```

`write.mi` *Writes mi impuations to file*

Description

Writes the imputed datasets to file for the `mi` object into the `csv`, `dta`, and `table` format.

Usage

```
write.mi(object, format = c("csv", "dta", "table"), ...)
```

Arguments

`object`	`mi` object
`format`	output format, only "csv", "dta" and "table" format are supported.
`...`	further arguments for `write` functions

Details

`write.mi` write each imputed dataset to a file in one of the three formats: `csv`, `dta` and `table`, using `write.csv`, `write.dta` and `write.table` repectively.

The output files should be

```
midata1.csv
midata2.csv
omidata3.csv
...
```

Author(s)

Yu-Sung Su <suyusung@tsinghua.edu.cn>,

References

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. (2011). "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software* 45(2).

See Also

`write.csv`, `write.table`, `write.dta`

Examples

```
# data(CHAIN)
# IMP <- mi(CHAIN)
# write.mi(IMP)
```

Index