The mice Package

June 25, 2007

Version 1.16

Date 2007-06-25

Title Multivariate Imputation by Chained Equations

Author S. Van Buuren & C.G.M. Oudshoorn

Maintainer Roel de Jong <dejongroel@gmail.com>

Depends MASS, nnet

Description Multivariate Imputation by Chained Equations

License GPL (version 2)


R topics documented:

  complete                             2
  glm.mids                            3
  lm.mids                             5
  md.pattern                          6
  mice.internal                       7
  mice                               8
  mice.impute.lda                    11
  mice.impute.logreg                 12
  mice.impute.logreg2               13
  mice.impute.mean                   15
  mice.impute.norm                   16
  mice.impute.norm.improper          17
  mice.impute.passive                18
  mice.impute.pmm                    19
  mice.impute.polyreg                20
  mice.impute.sample                 21
  mice.mids                          22
  mids                               23
complete

Description

Takes an object of type mids, fills in the missing data, and returns the completed data in a specified format.

Usage

complete(x, action=1)

Arguments

x
An object of class 'mids' (created by the function mice()).

action
If action is a scalar between 1 and x$m, the function returns the data with the action’s imputation filled in. Thus, action=1 returns the first completed data set. The can also be one of the following strings: "long", "broad", "repeated". This has the following meaning:

action="long" produces a long matrix with n*m rows, containing all imputed data plus two additional variables "_ID_" (containing the row names) and "_IMP_" (containing the imputation number).

action="broad" produces a broad matrix with m times the number of columns in the original data. The first ncol(x$data) columns contain the first imputed data matrix. Column names are changed to reflect the imputation number.

action="repeated" produces a broad matrix with m times ncol(x$data) columns. The first m columns give the filled-in first variable. Column names are changed to reflect the imputation number.

Value

A data frame with the imputed values filled in.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000
glm.mids

References

See Also
mice, mids

Examples

data(nhanes)
imp <- mice(nhanes)  # do default multiple imputation on a numeric matrix
mat <- complete(im)  # fills in the first imputation
mat <- complete(im, 3) # fills in the third imputation
mat <- complete(im, "long")  # produces a long matrix with stacked complete data
mat <- complete(im, "b")  # a broad matrix
cor(mat)  # for numeric mat, produces a blocked correlation matrix, where
          # each m*m block contains of the same variable pair over different
          # multiple imputations.

glm.mids

Generalized Linear Regression on Multiply Imputed Data

Description
Performs repeated glm on a multiply imputed data set

Usage
glm.mids(formula, data, ...)

Arguments
formula a formula expression as for other regression models, of the form response predictors. See the documentation of lm and formula for details.
data An object of type mids, which stands for ’multiply imputed data set’, typically created by function mice().
... Additional parameters passed to glm.

Details
see glm

Value
An objects of class mira, which stands for ’multiply imputed repeated analysis’. This object contains in glm.objects, plus some descriptive information.
**Author(s)**

Stef van Buuren, Karin Oudshoorn, 2000

**References**


**See Also**

`glm`, `mids`, `mira`

**Examples**

data(nhanes)
imp <- mice(nhanes)  # do default multiple imputation on a numeric matrix
glm.mids((hyp==2)~bmi+chl, data=imp)
  # fit
  # $call:
  # glm.mids(formula = (hyp == 2) ~ bmi + chl, data = imp)
  # $call1:
  # mice(data = nhanes)
  # $nmis:
  # age bmi hyp chl
  # 0 9 8 10
  # $analyses:
  # $analyses[[1]]:
  # Call:
  # glm(formula = formula, data = data.i)
  # $ Coefficients:
  # (Intercept) bmi chl
  # -0.4746337 -0.01565534 0.005417846
  # Degrees of Freedom: 25 Total; 22 Residual
  # Residual Deviance: 2.323886
  # $analyses[[2]]:
  # Call:
  # glm(formula = formula, data = data.i)
  # $ Coefficients:
  # (Intercept) bmi chl
  # -0.1184695 -0.02885779 0.006090282
  # Degrees of Freedom: 25 Total; 22 Residual
  # Residual Deviance: 3.647927
  # $analyses[[3]]:
Description

Performs repeated linear regression on multiply imputed data set

Usage

lm.mids(formula, data, ...)

lm.mids

Linear Regression on Multiply Imputed Data
md.pattern

Arguments

formula a formula object, with the response on the left of a \ operator, and the terms, separated by + operators, on the right.
data An object of type 'mids', which stands for 'multiply imputed data set', typically created by function mice().
... Additional parameters passed to \texttt{lm}

Value

An objects of class 'mira', which stands for 'multiply imputed repeated analysis'. This object contains m \texttt{lm} objects, plus some descriptive information.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000

References


See Also

\texttt{lm}, \texttt{mids}, \texttt{mira}

Examples

data(nhanes)
imp <- mice(nhanes)  # do default multiple imputation on a numeric matrix
fit <- lm.mids(bmi~hyp+chl,data=imp)

\begin{verbatim}
md.pattern # Missing Data Pattern

\end{verbatim}

Description

Display missing-data patterns.

Usage

\texttt{md.pattern(x)}

Arguments

\x A data frame or a matrix containing the incomplete data. Missing values are coded as NA's.
Details

This function is useful for investigating any structure of missing observation in the data. In specific case, the missing data pattern could be (nearly) monotone. Monotonicity can be used to simplify the imputation model. See Schafer (1997) for details. Also, the missing pattern could suggest which variables could potentially be useful for imputation of missing entries.

Value

A matrix with \( \text{ncol}(x)+1 \) columns, in which each row corresponds to a missing data pattern (1=observed, 0=missing). Rows and columns are sorted in increasing amounts of missing information. The last column and row contain row and column counts, respectively.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000

References


Examples

data(nhanes)
md.pattern(nhanes)
# age hyp bmi chl
# 13 1 1 1 1 0
# 1 1 1 0 1 1
# 3 1 1 1 0 1
# 1 1 0 0 1 2
# 7 1 0 0 0 3
# 0 8 9 10 27

---

mice.internal  internal mice functions

Description

Internal functions for package mice.

Usage

check.imputationMethod(imputationMethod, defaultImputationMethod, visitSequence, data)
check.predictorMatrix(predictorMatrix, nmis, nvar)
check.visitSequence(visitSequence, nmis, nvar)
data.frame.to.matrix(x)
is.mids(data)
is.mipo(data)
is.mira(data)
is.passive(string)
logitreg(x, y, wt = rep(1, length(y)), intercept = TRUE, start, trace = TRUE, ...)
.norm.draw(y, ry, x)
.padModel(data, imputationMethod, predictorMatrix, visitSequence, nmis, nvar)
.pmm.match(z, yhat = yhat, y = y)
sampler(p, data, m, imp, r, visitSequence, maxit, printFlag)

Arguments

imputationMethod, defaultImputationMethod, visitSequence, data, nmis, nvar, predictorMatrix, ...

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000

mice

Multivariate Imputation by Chained Equations

Description

Produces an object of class "mids", which stands for 'multiply imputed data set'.

Usage

mice(data, m = 5,
imputationMethod = vector("character", length=ncol(data)),
predictorMatrix = (1 - diag(1, ncol(data))),
visitSequence = (1:ncol(data))[apply(is.na(data),2,any)],
defaultImputationMethod=c("pmm","logreg","polyreg"),
maxit = 5,
diagnostics = TRUE,
printFlag = TRUE,
seed = NA)

Arguments

data A data frame or a matrix containing the incomplete data. Missing values are coded as NA's.
m Number of multiple imputations. If omitted, m=5 is used.
imputationMethod Can be either a string, or a vector of strings with length ncol(data), specifying the elementary imputation method to be used for each column in data. If specified as a single string, the given method will be used for all columns. The default imputation method (when no argument is specified) depends on the measurement level of the target column and are specified by the defaultImputationMethod argument. Columns that need not be imputed have method "". See details for more information.
predictorMatrix
A square matrix of size ncol(data) containing 0/1 data specifying the set of predictors to be used for each target column. Rows correspond to target variables (i.e. variables to be imputed), in the sequence as they appear in data. A value of '1' means that the column variable is used as a predictor for the target variable (in the rows). The diagonal of predictorMatrix must be zero. The default for predictorMatrix is that all other columns are used as predictors (sometimes called massive imputation).

visitSequence
A vector of integers of arbitrary length, specifying the column indices of the visiting sequence. The visiting sequence is the column order that is used to impute the data during one iteration of the algorithm. A column may be visited more than once. All incomplete columns that are used as predictors should be visited, or else the function will stop with an error. The default sequence 1:ncol(data) implies that columns are imputed from left to right.

defaultImputationMethod
A vector of three strings containing the default imputation methods for numerical columns, factor columns with 2 levels, and factor columns with more than two levels, respectively. If nothing is specified, the following defaults will be used: pmm, predictive mean matching (numeric data); logreg, logistic regression imputation (binary data, factor with 2 levels); polyreg, polytomous regression imputation categorical data (factor >= 2 levels).

maxit
A scalar giving the number of iterations. The default is 5.

diagnostics
A Boolean flag. If TRUE, diagnostic information will be appended to the value of the function. If FALSE, only the imputed data are saved. The default is TRUE.

printFlag
An integer between 0 and 1000 that is used by the set.seed function for offsetting the random number generator. Default is to leave the random number generator alone.

Details
Generates multiple imputations for incomplete multivariate data by Gibbs Sampling. Missing data can occur anywhere in the data. The algorithm imputes an incomplete column (the target column) by generating appropriate imputation values given other columns in the data. Each incomplete column must act as a target column, and has its own specific set of predictors. The default predictor set consists of all other columns in the data. For predictors that are incomplete themselves, the most recently generated imputations are used to complete the predictors prior to imputation of the target column.

A separate univariate imputation model can be specified for each column. The default imputation method depends on the measurement level of the target column. In addition to these, several other methods are provided. Users may also write their own imputation functions, and call these from within the algorithm.

In some cases, an imputation model may need transformed data in addition to the original data (e.g. log or quadratic transforms). In order to maintain consistency among different transformations of the same data, the function has a special built-in method using the ~ mechanism. This method can
be used to ensure that a data transform always depends on the most recently generated imputations in the untransformed (active) column.

The data may contain categorical variables that are used in a regressions on other variables. The algorithm creates dummy variables for the categories of these variables, and imputes these from the corresponding categorical variable.

Built-in imputation methods are:

- **norm**  Bayesian linear regression (Numeric)
- **pmm**  Predictive mean matching (Numeric)
- **mean**  Unconditional mean imputation (Numeric)
- **logreg**  Logistic regression (2 categories)
- **logreg2**  Logistic regression (direct minimization) (2 categories)
- **polyreg**  Polytomous logistic regression (>= 2 categories)
- **lda**  Linear discriminant analysis (>= 2 categories)
- **sample**  Random sample from the observed values (Any)

**Special method**  If the first character of the elementary method is a ~, then the string is interpreted as the formula argument in a call to `model.frame(formula, data[!r[,j]],)`. This provides a simple mechanism for specifying a large variety of dependencies among the variables. For example transformed versions of imputed variables, recodes, interactions, sum scores, and so on, that may themselves be needed in other parts of the algorithm, can be specified in this way. Note that the ~ mechanism works only on those entries which have missing values in the target column. The user should make sure that the combined observed and imputed parts of the target column make sense. One easy way to create consistency is by coding all entries in the target as NA, but for large data sets, this could be inefficient. Moreover, this will not work in S-Plus 4.5. Though not strictly needed, it is often useful to specify `visitSequence` such that the column that is imputed by the ~ mechanism is visited each time after one of its predictors was visited. In that way, deterministic relation between columns will always be synchronized.

For example, for the j'th column, the `impute.norm` function that implements the Bayesian linear regression method can be called by specifying the string "norm" as the j'th entry in the vector of strings.

The user can write his or her own imputation function, say `impute.myfunc`, and call it for all columns by specifying `imputationMethod="myfunc"`, or for specific columns by specifying `imputationMethod=c("norm","myfunc",...).

**side effects:** Some elementary imputation method require access to the nnet or MASS libraries of Venables & Ripley. Where needed, these libraries will be attached.

**Value**

An object of class mids, which stands for 'multiply imputed data set'. For a description of the object, see the documentation on `mids`.

**Author(s)**

Stef van Buuren, Karin Oudshoorn, 2000
mice.impute.lda

References


See Also

complete, mids, lm.mids, set.seed

Examples

data(nhanes)
imp <- mice(nhanes) # do default multiple imputation on a numeric matrix
imp
imp$imputations$bmi # and list the actual imputations
complete(imp) # show the first completed data matrix
lm.mids(chl~age+bmi+hyp, imp) # repeated linear regression on imputed data

data(nhanes2)
mice(nhanes2,im=c("sample","pmm","logreg","norm")) # imputation on mixed data with a different method per column

mice.impute.lda

Elementary Imputation Method: Linear Discriminant Analysis

Description

Imputes univariate missing data using linear discriminant analysis

Usage

mice.impute.lda(y, ry, x)

Arguments

y
Incomplete data vector of length n

ry
Vector of missing data pattern (FALSE=missing, TRUE=observed)

x
Matrix (n x p) of complete covariates.
Details

Imputation of categorical response variables by linear discriminant analysis. This function uses the Venables/Ripley functions lda and predict.lda to compute posterior probabilities for each incomplete case, and draws the imputations from this posterior.

Value

A vector of length nmis with imputations.

Warning

The function does not incorporate the variability of the discriminant weight, so it is not 'proper' in the sense of Rubin. For small samples and rare categories in the y, variability of the mice.imputed data could therefore be somewhat underestimated.

Note

This function can be called from within the Gibbs sampler by specifying 'lda' in the imputation-Method argument. This method is usually faster and uses less resources than mice.impute.polyreg.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000

References


See Also

mice, lda, predict.lda

mice.impute.logreg  Elementary Imputation Method: Logistic Regression

Description

Imputes univariate missing data using logistic regression.

Usage

mice.impute.logreg(y, ry, x)
Arguments

- **y**: Incomplete data vector of length n
- **ry**: Vector of missing data pattern of length n (FALSE=missing, TRUE=observed)
- **x**: Matrix (n x p) of complete covariates.

Details

Imputation for binary response variables by the Bayesian logistic regression model. See Rubin (1987, p. 169-170) for a description of the method. The method consists of the following steps:

1. Fit a logit, and find \( (\hat{b}, V(\hat{b})) \)
2. Draw BETA from \( N(\hat{b}, V(\hat{b})) \)
3. Compute predicted scores for m.d., i.e. \( \text{logit}^{-1}(X \text{BETA}) \)
4. Compare the score to a random \((0,1)\) deviate, and mice.impute.

The method relies on the standard \text{glm.fit} function.

Value

- **imp**: A vector of length \( n_{\text{mis}} \) with imputations (0 or 1).

Note

An alternative is mice.impute.logreg2.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000

References


See Also

- \text{mice.glm}, \text{glm.fit}, \text{mice.impute.logreg2}
mice.impute.logreg2

Elementary Imputation Method: Logistic Regression

Description
Imputes univariate missing data using logistic regression.

Usage
imp <- mice.impute.logreg2(y, ry, x)

Arguments
- y: Incomplete data vector of length n
- ry: Vector of missing data pattern of length n (FALSE=missing, TRUE=observed)
- x: Matrix (n x p) of complete covariates.

Details
Imputation for binary response variables by the Bayesian logistic regression model. See Rubin (1987, p. 169-170) for a description of the method. The method consists of the following steps:

1. Fit a logit, and find (bhat, V(bhat))
2. Draw BETA from N(bhat, V(bhat))
3. Compute predicted scores for m.d., i.e. logit-1(X BETA)
4. Compare the score to a random (0,1) deviate, and mice.impute.

This method uses direct minimization of the likelihood function by means of V&R function logitreg (V&R, 2nd ed, p. 293).

Value
- imp: A vector of length nmis with imputations (0 or 1).

Note
An alternative is mice.impute.logreg.

Author(s)
Stef van Buuren, Karin Oudshoorn, 2000
**mice.impute.mean**

**Elementary Imputation Method: Simple Mean Imputation**

**References**


**See Also**
mice, logitreg, mice.impute.logreg

**Description**
Imputes the arithmetic mean of the observed data

**Usage**
mice.impute.mean(y, ry, x=NULL)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Incomplete data vector of length n</td>
</tr>
<tr>
<td>ry</td>
<td>Vector of missing data pattern (FALSE=missing, TRUE=observed)</td>
</tr>
<tr>
<td>x</td>
<td>Matrix (n x p) of complete covariates.</td>
</tr>
</tbody>
</table>

**Value**
A vector of length nmis with imputations.

**Warning**
Imputing the mean of a variable rarely produces appropriate inferences. See Little and Rubin (1987).

**Author(s)**
Stef van Buuren, Karin Oudshoorn, 2000
References

See Also
mice, mean

mice.impute.norm Elementary Imputation Method: Linear Regression Analysis

Description
Imputes univariate missing data using linear regression analysis

Usage
mice.impute.norm(y, ry, x)

Arguments
y Incomplete data vector of length n
ry Vector of missing data pattern (FALSE=missing, TRUE=observed)
x Matrix (n x p) of complete covariates.

Details
Draws values of beta and sigma for Bayesian linear regression imputation of y given x according to Rubin p. 167.

Value
A vector of length nmis with imputations.

Note
Using mice.impute.norm for all columns gives results similar to Schafer’s norm method (Schafer, 1997), though much slower.

Author(s)
Stef van Buuren, Karin Oudshoorn, 2000
mice.impute.norm.improper

Elementary Imputation Method: Linear Regression Analysis (improper)

Description

Imputes univariate missing data using linear regression analysis (improper version)

Usage

mice.impute.norm.improper(y, ry, x)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Incomplete data vector of length n</td>
</tr>
<tr>
<td>ry</td>
<td>Vector of missing data pattern (FALSE=missing, TRUE=observed)</td>
</tr>
<tr>
<td>x</td>
<td>Matrix (n x p) of complete covariates.</td>
</tr>
</tbody>
</table>

Details

This creates imputation using the spread around the fitted linear regression line of y given x, as fitted on the observed data.

Value

A vector of length nmis with imputations.

Warning

The function does not incorporate the variability of the regression weights, so it is not ‘proper’ in the sense of Rubin. For small samples, variability of the mice.imputed data is therefore somewhat underestimated.

Note

This function is provided mainly to allow comparison between proper and improper norm methods.

References


mice.impute.passive

Elementary Imputation Method: Passive Imputation

Description
Derive a new variable based on the mice.imputed data

Usage
mice.impute.passive(data, func)

Arguments
  data     A data frame
  func     A formula specifying the transformations on data

Details
This is a special imputation function for so-called passive imputation. Using this function, the user can specify, at any point in the mice Gibbs sampling algorithm, a function on the (mice.imputed) data. This is useful, for example, to compute a cubic version of a variable, a transformation like \( Q = W/H^2 \) based on two variables, or a mean variable like \( (x_1 + x_2 + x_3)/3 \). The so derived variables might be used in other places in the imputation model. The function allows to dynamically derive virtually any function of the mice.imputed data at virtually any time.

Value
  t        The tranformed data.

Author(s)
Stef van Buuren, Karin Oudshoorn, 2000

References

See Also
mice, mice.impute.norm
References

See Also
mice

Elementary Imputation Method: Linear Regression Analysis

Description
Imputes univariate missing data using predictive mean matching

Usage
mice.impute.pmm(y, ry, x)

Arguments
y Incomplete data vector of length n
ry Vector of missing data pattern (FALSE=missing, TRUE=observed)
x Matrix (n x p) of complete covariates.

Details
Imputation of y by predictive mean matching, based on Rubin (p. 168, formulas a and b). The procedure is as follows:

1. Draw beta and sigma from the proper posterior
2. Compute predicted values for yobs and ymis
3. For each ymis, find the observation with closest predicted value, and take its observed y as the imputation.

The matching is on yhat, NOT on y, which deviates from formula b.

Value
imp A vector of length nmis with imputations.

Author(s)
Stef van Buuren, Karin Oudshoorn, 2000
References

mice.impute.polyreg

Elementary Imputation Method: Polytomous Regression

Description
Imputes missing data in a categorical variable using polytomous regression

Usage
mice.impute.polyreg(y, ry, x)

Arguments
- y: Incomplete data vector of length n
- ry: Vector of missing data pattern (FALSE=missing, TRUE=observed)
- x: Matrix (n x p) of complete covariates.

Details
Imputation for categorical response variables by the Bayesian polytomous regression model. See J.P.L. Brand (1999), Chapter 4, Appendix B.

The method consists of the following steps:
1. Fit categorical response as a multinomial model
2. Compute predicted categories
3. Add appropriate noise to predictions.

This algorithm uses the function multinom from the libraries nnet and MASS (Venables and Ripley).

Value
A vector of length nmis with imputations.

Author(s)
Stef van Buuren, Karin Oudshoorn, 2000
References


See Also

mice, multinom

mice.impute.sample  Elementary Imputation Method: Simple Random Sample

Description

Imputes a random sample from the observed y data

Usage

mice.impute.sample(y, ry, x=NULL)

Arguments

y  Incomplete data vector of length n
ry  Vector of missing data pattern (FALSE=missing, TRUE=observed)
x  Matrix (n x p) of complete covariates.

Details

This function takes a simple random sample from the observed values in y, and returns these as imputations.

Value

A vector of length nmis with imputations.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000

References

Description

Takes a "mids"-object, and produces an new object of class "mids".

Usage

mice.mids(obj, maxit=1, diagnostics=TRUE, printFlag=TRUE)

Arguments

- **obj**: An object of class "mids", typically produces by a previous call to mice() or mice.mids()
- **maxit**: The number of additional Gibbs sampling iterations.
- **diagnostics**: A Boolean flag. If TRUE, diagnostic information will be appended to the value of the function. If FALSE, only the imputed data are saved. The default is TRUE.
- **printFlag**: A Boolean flag. If TRUE, diagnostic information during the Gibbs sampling iterations will be written to the command window. The default is TRUE.

Details

This function enables the user to split up the computations of the Gibbs sampler into smaller parts. This is useful for the following reasons:

- RAM memory may become easily exhausted if the number of iterations is large. Returning to prompt/session level may alleviate these problems.
- The user can compute customized convergence statistics at specific points, e.g. after each iteration, for monitoring convergence. - For computing a 'few extra iterations'.

Note: The imputation model itself is specified in the mice() function and cannot be changed with mice.mids. The state of the random generator is saved with the mids-object.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000

References


See Also
Examples

data(nhanes)
imp1 <- mice(nhanes,maxit=1)
imp2 <- mice.mids(imp1)

# yields the same result as
imp <- mice(nhanes,maxit=2)

# for example:
#
# > imp$imp$bmi[1,]
# 1  2  3  4  5
# 1 30.1 35.3 33.2 35.3 27.5
# > imp2$imp$bmi[1,]
# 1  2  3  4  5
# 1 30.1 35.3 33.2 35.3 27.5
#

mids Multiply Imputed Data Set

Description

An object containing a multiply imputed data set. The "mids" object is generated by the mice and mice.mids functions. The "mids" class of objects has methods for the following generic functions: print, summary, plot

Usage

## S3 method for class 'mids':
print(x, ...)
## S3 method for class 'mids':
summary(object, ...)
## S3 method for class 'mids':
plot(x, y, ...)

Arguments

x A mids object.

object A mids object.

y Not used.

... Not used.
**Value**

- **call**: The call that created the object.
- **data**: A copy of the incomplete data set.
- **m**: The number of imputations.
- **nmis**: An array containing the number of missing observations per column.
- **imp**: A list of nvar components with the generated multiple imputations. Each part of the list is a nmis[j] by m matrix of imputed values for variable j.
- **imputationMethod**: A vector of strings of length(nvar) specifying the elementary imputation method per column.
- **predictorMatrix**: A square matrix of size ncol(data) containing 0/1 data specifying the predictor set.
- **visitSequence**: The sequence in which columns are visited.
- **seed**: The seed value of the solution.
- **iteration**: Last Gibbs sampling iteration number.
- **lastSeedValue**: The most recent seed value.
- **chainMean**: A list of m components. Each component is a length(visitSequence) by maxit matrix containing the mean of the generated multiple imputations. The array can be used for monitoring convergence. Note that observed data are not present in this mean.
- **chainCov**: A list with similar structure of itermean, containing the covariances of the imputed values.
- **pad**: A list containing various settings of the padded imputation model, i.e. the imputation model after creating dummy variables. Normally, this array is only useful for error checking.

**Author(s)**

Stef van Buuren, Karin Oudshoorn, 2000

**References**

Description

The "mipo" object is generated by the `lm.mids` and `glm.mids` functions. The "mipo" class of objects has methods for the following generic functions: print, summary.

Usage

```r
print.mipo(x,...)
summary.mipo(object,...)
```

Arguments

- `x`, `object`  An object containing the m fit objects of a complete data analysis, plus some additional information.
- `...`  not used.

Value

- `call`  The call that created the mipo object.
- `call1`  The call that created the mira object that was used in 'call'.
- `call2`  The call that created the mids object that was used in 'call1'.
- `nmis`  An array containing the number of missing observations per column.
- `m`  Number of multiple imputations.
- `qhat`  An m x npar matrix containing the complete data estimates for the npar parameters of the m complete data analyses.
- `u`  An m x npar x npar array containing the variance-covariance matrices of the m complete data analyses.
- `qbar`  The average of complete data estimates.
- `ubar`  The average of the variance-covariance matrix of the complete data estimates.
- `b`  The between imputation variance-covariance matrix.
- `t`  The total variance-covariance matrix.
- `r`  Relative increases in variance due to missing data.
- `df`  Degrees of freedom associated with the t-statistics.
- `f`  Fractions of missing information.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000
mira

Multiply Imputed Repeated Analysis

Description

The "mira" object is generated by the lm.mids and glm.mids functions. The "mira" class of objects has methods for the following generic functions: print, summary.

Usage

print.mira(x,...)
summary.mira(object, correlation, ...)

Arguments

x, object  An object containing the m fit objects of a complete data analysis, plus some additional information.
correlation
...  not used

Value

call  The call that created the object.
call1  The call that created the mids object that was used in 'call'.
nmis  An array containing the number of missing observations per column.
analyses  A list of m components containing the individual fit objects from each of the m complete data analyses.

Author(s)

Stef van Buuren, Karin Oudshoorn, 2000

References

nhanes  

nhanes  
nhanes data set

Description

Usage
data(nhanes)

Format
A data frame with 25 observations on the following 4 variables.

age  a numeric vector
bmi  a numeric vector
hyp  a numeric vector
chl  a numeric vector

Source

nhanes2  
nhanes2 data set

Description

Usage
data(nhanes2)

Format
A data frame with 25 observations on the following 4 variables.

age  a factor with levels 1 2 3
bmi  a numeric vector
hyp  a factor with levels 1 2
chl  a numeric vector

Source
Multiple Imputation Pooling

Description
Pools the results of m repeated complete data analysis

Usage
pool(object, method="smallsample")

Arguments
- object: An object of class 'mira', produced by functions like lm.mids or glm.mids.
- method: A string describing the method to compute the degrees of freedom. The default value is "smallsample", which specifies the is Barnard-Rubin adjusted degrees of freedom (Barnard & Rubin, 1999) for small samples. Specifying a different string produces the conventional degrees of freedom as in Rubin (1987).

Details
The function averages the estimates of the complete data model, computes the total variance over the repeated analyses, and computes the relative increase in variance due to nonresponse and the fraction of missing information. The function relies on the availability of:

1. the estimates of the model, typically present as 'coefficients' in the fit object
2. an appropriate estimate of the variance-covariance matrix of the estimates per analyses.

R-Specific: The original use of Varcov has been removed to vcov (VR MASS).

Value
An object of class 'mipo', which stands for 'multiple imputation pooled'.

Author(s)
Stef van Buuren, Karin Oudshoorn, 2000

References
See Also

lm.mids, glm.mids, vcov, print.mira, summary.mira

Examples

data(nhanes)
imp <- mice(nhanes)
fit <- lm.mids(bmi ~ hyp + chl, data = imp)
pool(fit)
# Call: pool(object = fit)
# Pooled coefficients:
# (Intercept)    hyp    chl
# 21.29782 -1.751721 0.04085703
#
# Fraction of information about the coefficients missing due to nonresponse:
# (Intercept)    hyp    chl
# 0.1592247 0.1738868 0.3117452
#
> summary(pool(fit))
# est       se      t      df     Pr(>|t|)
# (Intercept) 21.29781702 4.33668150 4.9110863 16.95890 0.0001329371
# hyp -1.75172102 2.30620984 -0.7595671 16.39701 0.4582953905
# chl 0.04085703 0.02532914 1.6130442 11.50642 0.1338044664
# lo 95  hi 95   missing     fmi
# (Intercept) 12.14652927 30.4491048 NA 0.1592247
# hyp -6.63106456 3.1276225 8 0.1738868
# chl -0.01459414 0.0963082 10 0.3117452
### Index

#### *Topic datasets*
- nhanes, 27
- nhanes2, 27

#### *Topic misc*
- complete, 1
- glm.mids, 3
- lm.mids, 5
- md.pattern, 6
- mice, 8
- mice.impute.lda, 11
- mice.impute.logreg, 12
- mice.impute.logreg2, 13
- mice.impute.mean, 15
- mice.impute.norm, 16
- mice.impute.norm.improper, 17
- mice.impute.passive, 18
- mice.impute.pmm, 19
- mice.impute.polyreg, 20
- mice.impute.sample, 21
- mice.internal, 7
- mice.mids, 22
- mids, 23
- mipo, 25
- mira, 26
- pool, 28
- .norm.draw (mice.internal), 7
- .pmm.match (mice.internal), 7

- check.imputationMethod
  (mice.internal), 7
- check.predictorMatrix
  (mice.internal), 7
- check.visitSequence
  (mice.internal), 7
- complete, 1, 11

- data.frame.to.matrix
  (mice.internal), 7

- formula, 3

- glm, 3, 13
- glm.fit, 13
- glm.mids, 3, 25, 29

- is.mids (mice.internal), 7
- is.mipo (mice.internal), 7
- is.mira (mice.internal), 7
- is.passive (mice.internal), 7

- lda, 12
- lm, 3, 5, 6
- lm.mids, 5, 11, 25, 29
- logitreg, 14
- logitreg (mice.internal), 7

- md.pattern, 6
- mean, 15

- mice, 2, 8, 12–15, 18, 19, 21
- mice.impute.lda, 11
- mice.impute.logreg, 12, 14
- mice.impute.logreg2, 13, 13
- mice.impute.mean, 15
- mice.impute.norm, 16, 18
- mice.impute.norm.improper, 17
- mice.impute.passive, 18
- mice.impute.pmm, 19
- mice.impute.polyreg, 12, 20
- mice.impute.sample, 21
- mice.internal, 7
- mice.mids, 22
- mids, 2, 3, 6, 10, 11, 23
- mipo, 25
- mipo (mice.internal), 7
- mira, 3, 6, 26
- multinom, 21

- nhanes, 27
- nhanes2, 27

- padModel (mice.internal), 7
- plot.mids (mids), 23
INDEX

pool, 28
predict.lda, 12
print.mids(mids), 23
print.mipo(mipo), 25
print.mira, 29
print.mira(mira), 26

sampler(mice.internal), 7
set.seed, 11
summary.mids(mids), 23
summary.mipo(mipo), 25
summary.mira, 29
summary.mira(mira), 26

vcov, 29