Package ‘imi’

September 26, 2017

Type Package

Title What the Package Does (Title Case)

Version 0.1.0

Author Who wrote it

Maintainer The package maintainer `<yourself@somewhere.net>`

Imports mice, Amelia, MKmisc

Description More about what it does (maybe more than one line)
Use four spaces when indenting paragraphs within the Description.

License What license is it under?

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

R topics documented:

combine.mi *combine.mi*

Description

This function combines vectors of estimated parameters and theircovariance matrices from different imputed sets of data using Rubin's rule (Rubin, 1978).

Usage

```
combine.mi(mi.param.est,mi.param.cov)
```

Arguments

mi.param.est	A matrix of size number of imputed datasets by number of parameters in the model. Each row of this matrix should present the estimated parameters from each imputed dataset.
mi.param.cov	A list of length number of imputed datasets. Each element of this list is a covariance matrix of the size of the number of parameters in the model.

Value

param.est	The combined vector of estimated parameters.
param.cov	The combined covariance matrix of the estimated parameters.
between.cov	The between-imputation covariance matrix.
within.cov	The within-imputation covariance matrix.

References

Rubin, Donald B. Multiple imputation for nonresponse in surveys. John Wiley & Sons, 1978.

imi.glm *imi.glm*

Description

This function fits a generalized linear model for a given set of predictors and a response variable to an incomplete dataset using multiple imputation and determines the sufficient number of imputed datasets using iterative multiple imputation (imi) procedure.

Usage

```
imi.glm(data.miss,family=binomial(link='logit'),M0=5,max.M=500,epsilon,method='pmm',
          resp,regressors,conv.plot=TRUE,dis.method='mahalanobis',
          mah.scale='combined',successive.valid=3,max.iter.glm=1000)
```

Arguments

`data.miss` A data frame with the variable in the model as its columns. Note that the missing values should be indicated by NA.

`family` Indicating the error distribution and the link function (see documentation of function glm in R stats package).

`M0` The initial number of imputations, it should be an integer with minimum 2. It also can take the string value 'auto' which lets the user to decide about the initial number of imputation after the software reports the time it takes to impute the data and fit the model to it twice.

`max.M` The maximum number of iterations which the algorithm should terminate afterwards in case of non-covergence.

`epsilon` The threshold for difference between two iterations.

`method` Specifying string value 'mvn' would impute the data using a multivariate normal predictive model in R package amelia2, any other specification will impute the data using fully conditional specification approach in R package mice. One can see the method in documentation of function mice in R package mice. Specifying 'auto' will selected the predictive model based on the measurement level of each variable.

`resp` A string value input with the name of the response variable. Note that this should match the name of one of the columns in the data.miss.

`regressors` A vector of string values with the names of the predictors. Note that they should match the names of the variables in data.miss.

`conv.plot` A logitical value, if TRUE then a convergence plot will be generated, if FALSE no plot will be provided.

`dis.method` A string takes its values among 'euclidean', 'inf.norm', and 'mahalanobis' which specifies the distance measure between two iterations. Note that our suggestion is to use 'mahalanobis', other options are provided for research purposes.

`mah.scale` A string takes its values among 'within', 'between', and 'combined' which specifies the scale matrix in Mahalanobis distance. Note that our suggestion is to use 'combined', other options are provided for research purposes.

`successive.valid`
An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.

`max.iter.glm` The maximum number of iterations for the glm algorithm.

`print.progress` A logical variable, if TRUE it prints the progress of imputation.

Value

`mi.param` A list with the final MI-based estimated model parameters, their covariance matrix, as well as the within and between imputation covariance matrices.

`data.imp` A list with imputed datasets as its components.

`dis.steps` A vector with computed distance between iterations.

`conv.status` If 1 then convergence is achieved, if 0 with max.M iterations, still the convergence could not be achieved.

`M` The selected number of imputed datasets.

References

https://www.rdocumentation.org/packages/mice/versions/2.30/topics/mice

https://cran.r-project.org/web/packages/Amelia/

Examples

```
> # specifying sample size and number of predictors
> sample.size=100
> num.var.log=2
> # creating a correlated set of predictors
> x.orig=matrix(rnorm(num.var.log*sample.size),sample.size,num.var.log)
> x=cbind(scale(x.orig,
+               center=TRUE,scale=FALSE))
> # creating the compound-symmetry structured covariance matrix
> sigma2=4
> tau=1
> cov.mat=diag(sigma2,num.var.log)+tau
> # making the data correlated
> chol.cov=chol(cov.mat)
> for (i in 1:sample.size){
+   x[i,]=t(chol.cov)
+ }
> x=t(t(x)+apply(x.orig,2,mean))
> # specifying model parameters
> beta=c( 0.2, -1.0,  0.5)
> z=beta[1]+(x
> # computing inverse logit transformation
> pr = 1/(1+exp(-z))
> # generating response variable
> y = rbinom(sample.size,1,pr)
> # creating complete the data
> data = data.frame(y,x)
> # specifying the regressors and predictors
> resp='y'
> regressors=c('X1','X2')
> # creating missing values in the dataset
> require('mice')
> x.miss=ampute(x,prop=0.1,mech='MAR')$amp
> data.miss=data.frame(y,x.miss)
> # Determining number of imputations, impute the incomplete data and fit the model to it
> out.glm=imi.glm (data.miss,family=binomial(link='logit'),M0='manual',max.M=500,
+          epsilon=0.05,method='mvn',resp,regressors, conv.plot=TRUE,
+          dis.method='mahalanobis',mah.scale='within',
+          successive.valid='manual',max.iter.glm=1000)
[1] "The time it takes (in seconds) to imput the data two times and fit the model to them is: 0.02"
What is your choice of initial number of imputations?2
What is your choice for successive steps validation?3
We are working on the initial imputation.
We are working on imputed dataset number  3
We are working on imputed dataset number  4
We are working on imputed dataset number  5
We are working on imputed dataset number  6
We are working on imputed dataset number  7
We are working on imputed dataset number  8
We are working on imputed dataset number  9
We are working on imputed dataset number  10
We are working on imputed dataset number  11
We are working on imputed dataset number  12
We are working on imputed dataset number  13
We are working on imputed dataset number  14
We are done! The convergence is acheived and the sufficient number of imputations is  14
```

---

imi.glm.conv *imi.glm.conv*

Description

This function uses iterative multiple imputation idea to compute the distance between estimated parameters and their precisions from imputed datasets when fitting a generalized linear model to see whether the already generated sets of imputed data are sufficient for such analysis or not.

Usage

```
imi.glm.conv(data.imp,family=binomial(link='logit'),epsilon,resp,regressors,
                         conv.plot=TRUE, dis.method='mahalanobis',
                     mah.scale='combined',successive.valid=3,max.iter.glm=1000)
```

Arguments

`data.imp` A list imputed sets of data as its components.

`family` Indicating the error distribution and the link function (see documentation of function glm in R stats package.)

`epsilon` The threshold for difference between two iterations.

`resp` A string value input with the name of the response variable. Note that this should match the name of one of the columns in the data.miss.

`regressors` A vector of string values with the names of the predictors. Note that they should match the names of the variables in data.miss.

`conv.plot` A logitical value, if 'TRUE' then a convergence plot will be generated, if 'FALSE' no plot will be provided.

`dis.method` A string takes its values among 'euclidean', 'inf.norm', and 'mahalanobis' which specifies the distance measure between two iterations. Note that our suggestion is to use 'mahalanobis', other options are provided for research purposes.

`mah.scale` A string takes its values among 'within', 'between', and 'combined' which specifies the scale matrix in Mahalanobis distance. Note that our suggestion is to use 'combined', other options are provided for research purposes.

`successive.valid` An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.

`max.iter.glm` The maximum number of iterations for the glm algorithm.

Value

`dis.steps` A vector with computed distance between iterations.

`sufficient.M` An integer indicating the minumum number of sufficent imputed datasets, in case of insufficiency it will take the value 'Not sufficient!'

See Also

imi.glm

Examples

```
# In the setting of the example for imi.glm, we have:
> imi.glm.conv(out.glm$data.imp,family=binomial(link='logit'),epsilon=0.05,resp,regressors,
+             conv.plot=TRUE,
+         dis.method='mahalanobis',mah.scale='within',successive.valid=3,max.iter.glm=1000)
$dis.steps
 [1] 0.46355012 0.20284904 0.14844815 0.02788952 0.09077971 0.12970693 0.04556116 0.02601013 0.05598301
[10] 0.06236743 0.04832095 0.02365007 0.01836882

$sufficient.M
[1] 12

> imi.glm.conv(out.glm$data.imp,family=binomial(link='logit'),epsilon=0.01,resp,regressors,
+             conv.plot=TRUE,
+         dis.method='mahalanobis',mah.scale='within',successive.valid=3,max.iter.glm=1000)
$dis.steps
 [1] 0.46355012 0.20284904 0.14844815 0.02788952 0.09077971 0.12970693 0.04556116 0.02601013 0.05598301
[10] 0.06236743 0.04832095 0.02365007 0.01836882

$sufficient.M
[1] "Not sufficient!"

>
```

imi.glm.more *imi.glm.more*

Description

This function fits a generalized linear model for a given set of predictors and a response variable to an incomplete dataset using multiple imputation and determines the sufficient number of imputed datasets using iterative multiple imputation (imi) procedure when a set of already imputed datasets are available.

Usage

```
imi.glm.more(data.miss,data.imp0,family=binomial(link='logit'),max.M=500,epsilon,
          method='pmm',resp,regressors,conv.plot=TRUE,dis.method='mahalanobis',
             mah.scale='combined',successive.valid=3,max.iter.glm=1000)
```

Arguments

`data.miss` A data frame with the variable in the model as its columns. Note that the missing values should be indicated by NA.

`family` Indicating the error distribution and the link function (see documentation of function glm in R stats package).

`data.imp0` A list with already imputed sets of data as its components.

`max.M` The maximum number of iterations which the algorithm should terminate afterwards in case of non-covergence.

`epsilon` The threshold for difference between two iterations.

`method` Specifying string value 'mvn' would impute the data using a multivariate normal predictive model in R package amelia2, any other specification will impute the data using fully conditional specification approach in R package mice. One can see the method in documentation of function mice in R package mice. Specifying 'auto' will selected the predictive model based on the measurement level of each variable.

`resp` A string value input with the name of the response variable. Note that this should match the name of one of the columns in the data.miss.

`regressors` A vector of string values with the names of the predictors. Note that they should match the names of the variables in data.miss.

`conv.plot` A logitical value, if TRUE then a convergence plot will be generated, if FALSE no plot will be provided.

`dis.method` A string takes its values among 'euclidean', 'inf.norm', and 'mahalanobis' which specifies the distance measure between two iterations. Note that our suggestion is to use 'mahalanobis', other options are provided for research purposes.

`mah.scale` A string takes its values among 'within', 'between', and 'combined' which specifies the scale matrix in Mahalanobis distance. Note that our suggestion is to use 'combined', other options are provided for research purposes.

`successive.valid`
An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.

`max.iter.glm` The maximum number of iterations for the glm algorithm.

`print.progress` A logical variable, if TRUE it prints the progress of imputation.

Value

`mi.param` A list with the final MI-based estimated model parameters, their covariance matrix, as well as the within and between imputation covariance matrices.

`data.imp` A list with imputed datasets as its components.

`dis.steps` A vector with computed distance between iterations.

`conv.status` If 1 then convergence is achieved, if 0 with max.M iterations, still the convergence could not be achieved.

`M` The selected number of imputed datasets.

References

https://www.rdocumentation.org/packages/mice/versions/2.30/topics/mice

https://cran.r-project.org/web/packages/Amelia/

Examples

```
# specifying sample size and number of predictors
sample.size=100
num.var.log=2
# creating a correlated set of predictors
x.orig=matrix(rnorm(num.var.log*sample.size),sample.size,num.var.log)
```

```
x=cbind(scale(x.orig,
              center=TRUE,scale=FALSE))
# creating the compound-symmetry structured covariance matrix
sigma2=4
tau=1
cov.mat=diag(sigma2,num.var.log)+tau
# making the data correlated
chol.cov=chol(cov.mat)
for (i in 1:sample.size){
  x[i,]=t(chol.cov)
}
x=t(t(x)+apply(x.orig,2,mean))
# specifying model parameters
beta=c( 0.2, -1.0,  0.5)
z=beta[1]+(x
# computing inverse logit transformation
pr = 1/(1+exp(-z))
# generating response variable
y = rbinom(n.samp,1,pr)
# creating complete the data
data = data.frame(y,x)
# specifying the regressors and predictors
resp='y'
regressors=c('X1','X2')
# creating missing values in the dataset
require('mice')
x.miss=ampute(x,prop=0.1,mech='MAR')$amp
data.miss=data.frame(y,x.miss)
# Determining number of imputations, impute the incomplete data and fit the model to it
out.glm=imi.glm (data.miss,family=binomial(link='logit'),M0='manual',max.M=500,epsilon=0.05,
                 method='mvn',resp,regressors, conv.plot=TRUE, dis.method='mahalanobis',
                  mah.scale='within',successive.valid='manual',max.iter.glm=1000)
-- Imputation 1 --

  1  2  3  4

-- Imputation 2 --

  1  2  3  4  5  6

[1] "The time it takes (in seconds) to imput the data two times and fit the model to them is: 0.04"
What is your choice of initial number of imputations?2
What is your choice for successive steps validation?3
-- Imputation 1 --

  1  2  3  4

-- Imputation 1 --

  1  2  3  4

-- Imputation 1 --

  1  2  3

-- Imputation 1 --
```

```
  1  2  3  4

-- Imputation 1 --

  1  2  3  4

-- Imputation 1 --

  1  2  3  4

-- Imputation 1 --

  1  2  3  4

-- Imputation 1 --

  1  2  3

-- Imputation 1 --

  1  2  3  4

-- Imputation 1 --

  1  2  3  4

-- Imputation 1 --

  1  2  3  4

-- Imputation 1 --

  1  2  3

> names(out.glm)
[1] "mi.param"    "data.imp"    "dis.steps"   "conv.status" "M"
```

imi.lm *imi.lm*

Description

This function fits a linear model for a given set of predictors and a response variable to an incomplete dataset using multiple imputation and determines the sufficient number of imputed datasets using iterative multiple imputation (imi) procedure.

Usage

```
imi.lm(data.miss,M0 = 5,max.M = 500,epsilon,method = 'pmm',resp,regressors,
          conv.plot = TRUE, dis.method = 'mahalanobis', mah.scale = 'combined',
           successive.valid = 3)
```

Arguments

`data.miss` A data frame with the variable in the model as its columns. Note that the missing values should be indicated by NA.

`M0` The initial number of imputations, it should be an integer with minimum 2. It also can take the string value 'auto' which lets the user to decide about the initial number of imputation after the software reports the time it takes to impute the data and fit the model to it twice.

`max.M` The maximum number of iterations which the algorithm should terminate afterwards in case of non-covergence.

`epsilon` The threshold for difference between two iterations.

`method` Specifying string value 'mvn' would impute the data using a multivariate normal predictive model in R package amelia2, any other specification will impute the data using fully conditional specification approach in R package mice. One can see the method in documentation of function mice in R package mice. Specifying 'auto' will selected the predictive model based on the measurement level of each variable.

`resp` A string value input with the name of the response variable. Note that this should match the name of one of the columns in the data.miss.

`regressors` A vector of string values with the names of the predictors. Note that they should match the names of the variables in data.miss.

`conv.plot` A logitical value, if TRUE then a convergence plot will be generated, if FALSE no plot will be provided.

`dis.method` A string takes its values among 'euclidean', 'inf.norm', and 'mahalanobis' which specifies the distance measure between two iterations. Note that our suggestion is to use 'mahalanobis', other options are provided for research purposes.

`mah.scale` A string takes its values among 'within', 'between', and 'combined' which specifies the scale matrix in Mahalanobis distance. Note that our suggestion is to use 'combined', other options are provided for research purposes.

`successive.valid`
An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.

`print.progress` A logical variable, if TRUE it prints the progress of imputation.

Value

`mi.param` A list with the final MI-based estimated model parameters, their covariance matrix, as well as the within and between imputation covariance matrices.

`data.imp` A list with imputed datasets as its components.

`dis.steps` A vector with computed distance between iterations.

`conv.status` If 1 then convergence is achieved, if 0 with max.M iterations, still the convergence could not be achieved.

`M` The selected number of imputed datasets.

References

https://www.rdocumentation.org/packages/mice/versions/2.30/topics/mice

https://cran.r-project.org/web/packages/Amelia/

Examples

```
> sample.size=100
> num.var=2
> # creating a correlated set of predictors
> x.orig=matrix(rnorm(num.var*sample.size),sample.size,num.var)
> x=cbind(scale(x.orig,
+               center=TRUE,scale=FALSE))
> # creating the compound-symmetry structured covariance matrix
> sigma2=4
> tau=1
> cov.mat=diag(sigma2,num.var)+tau
> # making the data correlated
> chol.cov=chol(cov.mat)
> for (i in 1:sample.size){
+   x[i,]=t(chol.cov)
+ }
> x=t(t(x)+apply(x.orig,2,mean))
> # specifying model parameters
> beta=c( 0.2, -1.0,  0.5)
> y=beta[1]+(x
> # creating complete the data
> data = data.frame(y,x)
> # specifying the regressors and predictors
> resp='y'
> regressors=c('X1','X2')
> # creating missing values in the dataset
> require('mice')
> x.miss=ampute(x,prop=0.1,mech='MAR')$amp
> data.miss=data.frame(y,x.miss)
> # Determining number of imputations, impute the incomplete data and fit the model to it
> out.lm=imi.lm (data.miss,M0='manual',max.M=500,epsilon=0.05,method='mvn',resp,regressors,
+                  conv.plot=TRUE,
+                 dis.method='mahalanobis',mah.scale='within',successive.valid='manual')
-- Imputation 1 --

  1  2

-- Imputation 2 --

  1  2

[1] "The time it takes (in seconds) to imput the data two times and fit the model to them is: 0.04"
What is your choice of initial number of imputations?2
What is your choice for successive steps validation?3
-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2
```

```
-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

> names(out.lm)
[1] "mi.param"    "data.imp"    "dis.steps"   "conv.status" "M"
```

imi.lm.conv *imi.lm.conv*

Description

This function uses iterative multiple imputation idea to compute the distance between estimated parameters and their precisions imputed datasets when fitting a linear models to see whether the already generated sets of imputed data are sufficient for such analysis or not.

Usage

```
imi.lm.conv(data.imp,epsilon,resp,regressors, conv.plot=TRUE,
                 dis.method='mahalanobis',mah.scale='combined',successive.valid=3)
```

Arguments

`data.imp` A list imputed sets of data as its components.

`epsilon` The threshold for difference between two iterations.

`resp` A string value input with the name of the response variable. Note that this should match the name of one of the columns in the data.miss.

`regressors` A vector of string values with the names of the predictors. Note that they should match the names of the variables in data.miss.

`conv.plot` A logitical value, if 'TRUE' then a convergence plot will be generated, if 'FALSE' no plot will be provided.

`dis.method` A string takes its values among 'euclidean', 'inf.norm', and 'mahalanobis' which specifies the distance measure between two iterations. Note that our suggestion is to use 'mahalanobis', other options are provided for research purposes.

`mah.scale` A string takes its values among 'within', 'between', and 'combined' which specifies the scale matrix in Mahalanobis distance. Note that our suggestion is to use 'combined', other options are provided for research purposes.

`successive.valid` An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.

Value

`dis.steps` A vector with computed distance between iterations.

`sufficient.M` An integer indicating the minumum number of sufficent imputed datasets, in case of insufficiency it will take the value 'Not sufficient!'

See Also

imi.lm

Examples

```
# In the setting of the example for imi.lm, we have:
imi.lm.conv(out.lm$data.imp,epsilon=0.05,resp,regressors,
+              conv.plot=TRUE,
+              dis.method='mahalanobis',mah.scale='within',successive.valid=3)
$dis.steps
 [1] 0.62518742 0.26078660 0.23218647 0.21404833 0.05736930 0.05001030 0.02659284
 [8] 0.08570338 0.03877358 0.06810190 0.04897386 0.02438733 0.04324450

$sufficient.M
[1] 14

>
> imi.lm.conv(out.lm$data.imp,epsilon=0.01,resp,regressors,
+              conv.plot=TRUE,
+              dis.method='mahalanobis',mah.scale='within',successive.valid=3)
$dis.steps
 [1] 0.62518742 0.26078660 0.23218647 0.21404833 0.05736930 0.05001030 0.02659284
 [8] 0.08570338 0.03877358 0.06810190 0.04897386 0.02438733 0.04324450

$sufficient.M
[1] "Not sufficient!"
```

imi.lm.more *imi.lm.more*

Description

This function fits a linear model for a given set of predictors and a response variable to an incomplete dataset using multiple imputation and determines the sufficient number of imputations using iterative multiple imputation (imi) procedure when a set of already imputed datasets are available.

Usage

```
imi.lm(data.miss,data.imp0,max.M = 500,epsilon,method = 'pmm',resp,regressors,
          conv.plot = TRUE, dis.method = 'mahalanobis', mah.scale = 'combined',
           successive.valid = 3)
```

Arguments

data.miss	A data frame with the variable in the model as its columns. Note that the missing values should be indicated by NA.
data.imp0	A list with already imputed sets of data as its components.
max.M	The maximum number of iterations which the algorithm should terminate afterwards in case of non-covergence.
epsilon	The threshold for difference between two iterations.
method	Specifying string value 'mvn' would impute the data using a multivariate normal predictive model in R package amelia2, any other specification will impute the data using fully conditional specification approach in R package mice. One can see the method in documentation of function mice in R package mice. Specifying 'auto' will selected the predictive model based on the measurement level of each variable.
resp	A string value input with the name of the response variable. Note that this should match the name of one of the columns in the data.miss.
regressors	A vector of string values with the names of the predictors. Note that they should match the names of the variables in data.miss.
conv.plot	A logitical value, if TRUE then a convergence plot will be generated, if FALSE no plot will be provided.
dis.method	A string takes its values among 'euclidean', 'inf.norm', and 'mahalanobis' which specifies the distance measure between two iterations. Note that our suggestion is to use 'mahalanobis', other options are provided for research purposes.
mah.scale	A string takes its values among 'within', 'between', and 'combined' which specifies the scale matrix in Mahalanobis distance. Note that our suggestion is to use 'combined', other options are provided for research purposes.
successive.valid	An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.
print.progress	A logical variable, if TRUE it prints the progress of imputation.

Value

`mi.param` A list with the final MI-based estimated model parameters, their covariance matrix, as well as the within and between imputation covariance matrices.

`data.imp` A list with imputed datasets as its components.

`dis.steps` A vector with computed distance between iterations.

`conv.status` If 1 then convergence is achieved, if 0 with max.M iterations, still the convergence could not be achieved.

`M` The selected number of imputed datasets.

References

https://www.rdocumentation.org/packages/mice/versions/2.30/topics/mice

Examples

```
> sample.size=100
> num.var=2
> # creating a correlated set of predictors
> x.orig=matrix(rnorm(num.var*sample.size),sample.size,num.var)
> x=cbind(scale(x.orig,
+               center=TRUE,scale=FALSE))
> # creating the compound-symmetry structured covariance matrix
> sigma2=4
> tau=1
> cov.mat=diag(sigma2,num.var)+tau
> # making the data correlated
> chol.cov=chol(cov.mat)
> for (i in 1:sample.size){
+   x[i,]=t(chol.cov)
+ }
> x=t(t(x)+apply(x.orig,2,mean))
> # specifying model parameters
> beta=c( 0.2, -1.0,  0.5)
> y=beta[1]+(x
> # creating complete the data
> data = data.frame(y,x)
> # specifying the regressors and predictors
> resp='y'
> regressors=c('X1','X2')
> # creating missing values in the dataset
> require('mice')
> x.miss=ampute(x,prop=0.1,mech='MAR')$amp
> data.miss=data.frame(y,x.miss)
> # Determining number of imputations, impute the incomplete data and fit the model to it
> out.lm=imi.lm (data.miss,M0='manual',max.M=500,epsilon=0.05,method='mvn',resp,regressors,
+                 conv.plot=TRUE,
+                dis.method='mahalanobis',mah.scale='within',successive.valid='manual')
-- Imputation 1 --

  1  2

-- Imputation 2 --

  1  2
```

```
[1] "The time it takes (in seconds) to imput the data two times and fit the model to them is: 0.04"
What is your choice of initial number of imputations?2
What is your choice for successive steps validation?3
-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

-- Imputation 1 --

  1  2

> names(out.lm)
[1] "mi.param"    "data.imp"    "dis.steps"   "conv.status" "M"
```

imi.make.plots *imi.make.plots*

Description

This function take the simulation output available via data of type list with the names: sim.ss, sim.sm, sim.sl, sim.ms, sim.mm, sim.ml, sim.ls, sim.lm, and sim.ll as the input and produce the convergence plots for estimated parameters as well as the p-values of the t-tests. These are similar to the Figure 1 and Figure 2 in the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets'

Usage

```
imi.make.plots(x)
```

Arguments

x	A list which should be selected from the available datasets in the package: sim.ss, sim.sm, sim.sl, sim.ms, sim.mm, sim.ml, sim.ls, sim.lm, and sim.ll

See Also

sim.ss, sim.sm, sim.sl, sim.ms, sim.mm, sim.ml, sim.ls, sim.lm, and sim.ll

Examples

```
imi.make.plots(sim.ss)
```

imi.t.test *imi.t.test*

Description

This function uses the function mi.t.test in R package MKmisc to perform one or two sample Student's t-test on an incomplete dataset using multiple imputation and determines the sufficient number of imputed datasets using iterative multiple imputation (imi) procedure.

Usage

```
imi.t.test(data.miss,M0=2,max.M=500,epsilon,method='pmm',
                          x, y = NULL, alternative='two.sided',mu,paired = FALSE,
                 var.equal = FALSE, conf.level = 0.95,conv.plot=TRUE,successive.valid=3)
```

Arguments

`data.miss` A data frame with the variable in the model as its columns. Note that the missing values should be indicated by NA.

`M0` The initial number of imputations, it should be an integer with minimum 2. It also can take the string value 'auto' which lets the user to decide about the initial number of imputation after the software reports the time it takes to impute the data and fit the model to it twice.

`max.M` The maximum number of iterations which the algorithm should terminate afterwards in case of non-covergence.

`epsilon` The threshold for difference between two iterations.

`method` Specifying string value 'mvn' would impute the data using a multivariate normal predictive model in R package amelia2, any other specification will impute the data using fully conditional specification approach in R package mice. One can see the method in documentation of function mice in R package mice. Specifying 'auto' will selected the predictive model based on the measurement level of each variable.

`x` Name of the data column which the t-test should be performed on it.

`y` In case of two sample t-test this input variable specifiess the name of the second column for paired-test, or an indicator variable showing different populations for indepdendent two sample t-test.

`alternative` A character string specifying the alternative hypothesis, it takes the values "two.sided" (default), "greater" or "less".

`mu` A real number specifying the test value.

`paired` A logical value specifying whether a paired t-test should be performed or not.

`var.equal` A logical value specifying whether variance equality should be assumed or not.

`conf.level` Confidence level of the test.

`conv.plot` A logitical value, if TRUE then a convergence plot will be generated, if FALSE no plot will be provided.

`successive.valid`
An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.

`print.progress` A logical variable, if TRUE it prints the progress of imputation.

Value

`test.result` A list with the final MI-based outcome of the t-test.

`data.imp` A list with imputed datasets as its components.

`dis.steps` A vector with computed distance between iterations.

`conv.status` If 1 then convergence is achieved, if 0 with max.M iterations, still the convergence could not be achieved.

`M` The selected number of imputed datasets.

References

https://stat.ethz.ch/R-manual/R-devel/library/stats/html/t.test.html

Examples

```
# To illustrate the use of this function we use the cholestrol dataset in R package norm2.
> library(norm2)
> data(cholesterol)
>
> chol.ttest=imi.t.test (cholesterol,M0='manual',max.M=500,epsilon=0.05/10,method='mvn',
+                      x=names(cholesterol)[3], y = NULL, alternative='two.sided',mu=220,
+                           paired = FALSE, var.equal = FALSE,
+                           conf.level = 0.95,conv.plot=TRUE,successive.valid='manual')
-- Imputation 1 --

  1  2  3  4  5  6

[1] "The time it takes (in seconds) to imput the data once and fit the model to it is: 0.01"
What is your choice of initial number of imputations?2
What is your choice for successive steps validation?3
-- Imputation 1 --

  1  2  3  4  5  6  7  8

-- Imputation 1 --

  1  2  3  4  5  6  7

# it continues till the convergence happens.
```

`imi.t.test.conv` *imi.t.test.conv*

Description

This function uses iterative multiple imputation idea to compute the distance between p-values of t-tests from imputed datasets to see whether the already generated sets of imputed data are sufficient for such analysis or not.

Usage

```
imi.glm.conv(data.imp,epsilon,x, y = NULL, alternative='two.sided',
               mu,paired = FALSE, var.equal = FALSE, conf.level = 0.95,
               conv.plot=TRUE,successive.valid=3)
```

Arguments

`data.imp`	A list imputed sets of data as its components.
`epsilon`	The threshold for difference between two iterations.
`x`	Name of the data column which the t-test should be performed on it.
`y`	In case of two sample t-test this input variable specifiess the name of the second column for paired-test, or an indicator variable showing different populations for indepdendent two sample t-test.
`alternative`	A character string specifying the alternative hypothesis, it takes the values "two.sided" (default), "greater" or "less".

`mu` A real number specifying the test value.

`paired` A logical value specifying whether a paired t-test should be performed or not.

`var.equal` A logical value specifying whether variance equality should be assumed or not.

`conf.level` Confidence level of the test.

`conv.plot` A logitical value, if TRUE then a convergence plot will be generated, if FALSE no plot will be provided.

`successive.valid`
An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.

Value

`dis.steps` A vector with computed distance between iterations.

`sufficient.M` An integer indicating the minumum number of sufficent imputed datasets, in case of insufficiency it will take the value 'Not sufficient!'

See Also

imi.t.test

Examples

```
# In th setting of the example of imi.t.test
imi.t.test.conv (chol.ttest$data.imp,epsilon=0.05/10,x=names(cholesterol)[3],
+     y = NULL, alternative='two.sided',mu,paired = FALSE, var.equal = FALSE,
+                       conf.level = 0.95,conv.plot=TRUE,successive.valid=3)
$dis.steps
 [1] 0.3117110189 0.1578016324 0.0295781152 0.0487635084 0.0288667328 0.0318036141 0.0173372375
 [8] 0.0090263689 0.0575828782
[10] 0.0007220647 0.0145497285 0.0417895357 0.0044211378 0.0124385165 0.0203312525
[16] 0.0050927153 0.0058802488 0.0117815975
[19] 0.0022407713 0.0026592932 0.0007328110

$sufficient.M
[1] 22

>
> imi.t.test.conv (chol.ttest$data.imp,epsilon=0.05/100,x=names(cholesterol)[3],
+        y = NULL, alternative='two.sided',mu,paired = FALSE, var.equal = FALSE,
+                  conf.level = 0.95,conv.plot=TRUE,successive.valid=3)
$dis.steps
 [1] 0.3117110189 0.1578016324 0.0295781152 0.0487635084 0.0288667328 0.0318036141 0.0173372375
 [8] 0.0090263689 0.0575828782
[10] 0.0007220647 0.0145497285 0.0417895357 0.0044211378 0.0124385165 0.0203312525 0.0050927153
[17] 0.0058802488 0.0117815975
[19] 0.0022407713 0.0026592932 0.0007328110

$sufficient.M
[1] "Not sufficient!"
```

imi.t.test.more *imi.t.test.more*

Description

This function uses the function mi.t.test in R package MKmisc to perform one or two sample Student's t-test on an incomplete dataset using multiple imputation and determines the sufficient number of imputations using iterative multiple imputation (imi) procedure.

Usage

```
imi.t.test(data.miss,data.imp0,max.M=500,epsilon,method='pmm',
               x, y = NULL, alternative='two.sided',mu,paired = FALSE, var.equal = FALSE,
                      conf.level = 0.95,conv.plot=TRUE,successive.valid=3)
```

Arguments

`data.miss` A data frame with the variable in the model as its columns. Note that the missing values should be indicated by NA.

`data.imp0` A list with already imputed sets of data as its components.

`max.M` The maximum number of iterations which the algorithm should terminate afterwards in case of non-covergence.

`epsilon` The threshold for difference between two iterations.

`method` Specifying string value 'mvn' would impute the data using a multivariate normal predictive model in R package amelia2, any other specification will impute the data using fully conditional specification approach in R package mice. One can see the method in documentation of function mice in R package mice. Specifying 'auto' will selected the predictive model based on the measurement level of each variable.

`x` Name of the data column which the t-test should be performed on it.

`y` In case of two sample t-test this input variable specifiess the name of the second column for paired-test, or an indicator variable showing different populations for indepdendent two sample t-test.

`alternative` A character string specifying the alternative hypothesis, it takes the values "two.sided" (default), "greater" or "less".

`mu` A real number specifying the test value.

`paired` A logical value specifying whether a paired t-test should be performed or not.

`var.equal` A logical value specifying whether variance equality should be assumed or not.

`conf.level` Confidence level of the test.

`conv.plot` A logitical value, if TRUE then a convergence plot will be generated, if FALSE no plot will be provided.

`successive.valid` An integer with minimum 1 which specifies the number of successive steps the stopping rule should be validated so the procedure could terminate.

`print.progress` A logical variable, if TRUE it prints the progress of imputation.

Value

`test.result` A list with the final MI-based outcome of the t-test.

`data.imp` A list with imputed datasets as its components.

`dis.steps` A vector with computed distance between iterations.

`conv.status` If 1 then convergence is achieved, if 0 with max.M iterations, still the convergence could not be achieved.

`M` The selected number of imputed datasets.

References

https://stat.ethz.ch/R-manual/R-devel/library/stats/html/t.test.html

Examples

```
# To illustrate the use of this function we use the cholestrol dataset in R package norm2.
> library(norm2)
> data(cholesterol)
>
> chol.ttest=imi.t.test (cholesterol,M0='manual',max.M=500,epsilon=0.05/10,method='mvn',
+                      x=names(cholesterol)[3], y = NULL, alternative='two.sided',mu=220,
+                           paired = FALSE, var.equal = FALSE,
+                           conf.level = 0.95,conv.plot=TRUE,successive.valid='manual')
-- Imputation 1 --

  1  2  3  4  5  6

[1] "The time it takes (in seconds) to imput the data once and fit the model to it is: 0.01"
What is your choice of initial number of imputations?2
What is your choice for successive steps validation?3
-- Imputation 1 --

  1  2  3  4  5  6  7  8

-- Imputation 1 --

  1  2  3  4  5  6  7

# it continues till the convergence happens.
```

`sim.ll` *sim.ll*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 64$ and $\rho = 0.8$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.ll")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Examples

```
data(sim.ll)
```

`sim.lm` *sim.lm*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 64$ and $\rho = 0.5$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.lm")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Examples

```
data(sim.lm)
```

`sim.ls` *sim.ls*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 64$ and $\rho = 0.2$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.ls")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Examples

```
data(sim.ls)
```

sim.ml *sim.ml*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 16$ and $\rho = 0.8$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.ml")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Examples

```
data(sim.ml)
```

sim.mm *sim.mm*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 16$ and $\rho = 0.5$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.mm")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Examples

```
data(sim.mm)
```

---

sim.ms *sim.ms*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 16$ and $\rho = 0.2$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.ms")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Examples

```
data(sim.ms)
```

---

sim.sl *sim.sl*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 1$ and $\rho = 0.8$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.sl")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Examples

```
data(sim.sl)
```

sim.sm *sim.sm*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 1$ and $\rho = 0.5$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.sm")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Examples

```
data(sim.sm)
```

sim.ss *sim.ss*

Description

This dataset is a list contains the simulation results from the paper 'Iterative multiple imputation: a framework to determine the number of imputed datasets' for the scenario where $\sigma^2 = 1$ and $\rho = 0.2$. It contains the averaged distance between two steps (over 100 replications) for $M = 500$. Also the selected final M for different ϵ's and k_0's.

Usage

```
data("sim.ss")
```

Format

The format is: List of 3 lists: the first list contains the averaged simulation results over 100 replications for parameter estimates, the second list contains the same but for the p-values of one sample and paired t-test, and the third list contains the final selected M for eahc scenario.

Details

One can use the function imi.make.plots to generate similar plots as in the paper (see Figure 1 and Figure 2).

Examples

```
data(sim.ss)
imi.make.plots(sim.ss)
```

Index