

Package ‘mimR’

April 17, 2009

Version 2.6.1

Title A package for graphical modelling in R

Author Søren Højsgaard <sorenh@agrsci.dk>

Maintainer Søren Højsgaard <sorenh@agrsci.dk>

Description An R interface to MIM for graphical modelling in R

URL <http://gbi.agrsci.dk/~shd/Public/mimR/>

License GPL (>= 2)

Depends methods,gRbase,graph,rcom,MASS

Suggests Rgraphviz

Encoding latin1

OS_type windows

Repository CRAN

Date/Publication 2008-11-08 09:16:10

R topics documented:

cellCounts	2
cliques	3
display	4
edges-methods	5
editMIM	6
fitted.mim	7
fixEdges	8
imputeMissing	9
mim-class	10
MIMbase	11
MIMfit	12

MIMmodels	13
mimR	15
mimRprint-MIM	16
modelTest	17
nthOrderModel	18
plot	19
retrieveData	19
stepwise	20
toMIM	21
variableType	22

Index	23
--------------	-----------

cellCounts	<i>Convenience functions for specifying sufficient statistics for purely discrete and purely continuous graphical models</i>
------------	--

Description

These functions provide a way of specifying a contingency table by a vector of counts and for specifying the sufficient statistics for a covariance selection model as a covariance matrix.

Usage

```
cellCounts(varNames, nLevels = NULL, valueLabels = NULL, observations)
empCov(S, counts = NULL, sd = NULL, mean = rep(0, ncol(S)))
```

Arguments

varNames	~~Describe varNames here~~
nLevels	~~Describe nLevels here~~
valueLabels	~~Describe valueLabels here~~
observations	~~Describe observations here~~
S	~~Describe S here~~
counts	~~Describe counts here~~
sd	~~Describe sd here~~
mean	~~Describe mean here~~

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

```
comp1      Description of 'comp1'
comp2      Description of 'comp2'
...
```

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, An Introduction to Graphical Modelling, Springer Verlag, 2002

See Also

~~objects to See Also as [help](#), ~~~

Examples

```
x <- cellCounts(varNames=c("aa", "bb"),
  valueLabels=list(aa=c("a1", "a2"), bb=c("b100", "b200")),
  observations=c(1,2,3,4))

as.gmData(x)

S <- structure(c(305.77, 127.22, 101.58, 106.27, 117.4, 127.22, 172.84,
85.16, 94.67, 99.01, 101.58, 85.16, 112.89, 112.11, 121.87, 106.27,
94.67, 112.11, 220.38, 155.54, 117.4, 99.01, 121.87, 155.54,
297.76), .Dim = c(5L, 5L), .Dimnames = list(c("me", "ve", "al",
"an", "st"), c("me", "ve", "al", "an", "st")))

x <- empCov (S, 88)

as.gmData(x)
```

`cliques`*Get cliques, edges etc of MIM model objects*

Description

Get cliques, edges etc of MIM model objects

Usage

```
## S4 method for signature 'mim':  
cliques(object, which)
```

Arguments

<code>object</code>	A MIM model object
<code>which</code>	Not used

Value

A list

Note

Before using `mimR`, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, *An Introduction to Graphical Modelling*, Springer Verlag, 2002

Examples

```
data(carass)  
gmd.carc <- as.gmData(carass)  
m.sat <- fit(mim("../", data=gmd.carc))  
cliques(m.sat)  
nodes(m.sat)  
edges(m.sat)
```

`display`*Display a 'mim' model graphically*

Description

Displays a model graphically. This function is highly experimental and requires that the packages 'Rgraphviz' and 'graph' are loaded

Usage

```
display(x)
## S4 method for signature 'mim':
display(x)
```

Arguments

`x` A mim model object

Details

None

Value

The graph object.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

See Also

[mim](#)

Examples

```
data(rats)
gd <- as.gmData(rats)
m12 <- mim("Sex:Drug/Sex:Drug:W1+Sex:Drug:W2/W1+W2", data=gd)
plot(m12)
```

edges-methods *Retrieve edges and nodes of a mim model*

Description

Retrieve edges and nodes of a mim model

Methods

edges Return the edges of a mim model object

nodes Return the nodes (vertices) of a mim model object

editMIM *Editing a mim model object*

Description

Editing a mim model object by adding/deleting edges.

Usage

```
editmim(object, deleteEdge=NULL, addEdge=NULL, haddEdge=NULL,
        deleteTerm=NULL, addTerm=NULL, fit=object$fit)
## S3 method for class 'mim':
update(object, deleteEdge=NULL, addEdge=NULL, haddEdge=NULL,
        deleteTerm=NULL, addTerm=NULL, fit=object$fit, ...)
testdelete(edge, object, arg=NULL)
```

Arguments

object	A mim model object
deleteEdge	Edges to be deleted
addEdge	Edges to be added
haddEdge	Edges to be added (homogeneously)
deleteTerm	Terms to be deleted
addTerm	Terms to be added
...	Additional arguments to update. Currently not used.
fit	Should the updated model be fitted
edge	Edge to be tested
arg	Additional arguments to MIM specifying the tests

Details

retrieve is used for retrieving a model (as a mim object) manually from MIM after e.g. altering a model in the MIM program directly

Value

A new mim model object

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, An Introduction to Graphical Modelling, Second Edition, Springer Verlag, 2000

See Also

See Also as [modelTest](#)

Examples

```
data(rats)
gmd.rats <- as.gmData(rats)
m.main <- mim(".", data=gmd.rats)
m2 <- editmim(m.main, addEdge=c("Sex:Drug", "Sex:W2"))
m3 <- editmim(m.main, addEdge=c("Sex:Drug", "Sex:W2"), haddEdge="Drug:W1:W2")

summary(m2)
summary(m3)

data(housing)

housingTab <- xtabs(Freq ~ Sat + Infl + Type + Cont, data = housing)
ht <- as.gmData(housingTab)

marg1 <- mim("Sat:Infl:Cont//", data=ht)
testdelete("Sat:Cont", marg1)

## Try deleting an edge not in the model:

testdelete("Sat:Type", marg1)
testdelete("Sat:Cont", marg1, arg="MJ")
testdelete("Sat:Cont", marg1, arg="M")
```

`fitted.mim`*Extract fitted values (parameter estimates)*

Description

Extracts fitted values (parameter estimates) from a mim model.

Usage

```
## S3 method for class 'mim':  
fitted(object, data.frame=FALSE, ...)  
modelInfo(object, slot=NULL)
```

Arguments

<code>object</code>	A mim model object
<code>data.frame</code>	If the model is discrete (a log linear model) then fitted values are returned as a table by default. Setting <code>data.frame=TRUE</code> implies that fitted values are returned as a dataframe with the column 'Freq' containing the fitted values.
<code>...</code>	Other arguments
<code>slot</code>	A specific slot of the modelInfo list. If <code>NULL</code> , the entire list is returned.

Value

A data frame

Note

Before using `mimR`, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, An Introduction to Graphical Modelling, Second Edition, Springer Verlag, 2000

See Also

[simulate](#)

Examples

```
data(rats)
gmd.rats <- as.gmData(rats)
m2 <- mim("../", data=gmd.rats)
mf2 <- fit(m2)
parms <- fitted(mf2)
parms
```

`fixEdges`*Fixing edges and sets in a mim model*

Description

Fix edges and sets in a mim model.

Usage

```
fixEdges(v=NULL, mim=NULL)
fixSet(v=NULL, mim=NULL)
```

Arguments

<code>v</code>	An edge/vertex set, given as "a:b+c:d" or "a:b,c:d"
<code>mim</code>	A mim object

Details

If `v=NULL` then all fixes are removed. Only in this case the `mim` argument needs not to be given.

Value

None

Note

Before using `mimR`, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, An Introduction to Graphical Modelling, Springer Verlag, 2002

See Also

See Also as [stepwise.mim](#)

`imputeMissing` *Impute missing values*

Description

Imputes missing values in a data set in MIM after a model has been fitted.

Usage

```
imputeMissing()
```

Value

None

Note

Before using `mimR`, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, *An Introduction to Graphical Modelling*, Springer Verlag, 2002

See Also

[fit.mim](#)

`mim-class` *Class "mim" – mim model objects*

Description

A `mim` model object holds a `mim` model which includes log-linear models for contingency tables, covariance selection models for the multivariate normal distribution and mixed interaction models (a combination of the two former).

Objects from the Class

A virtual Class: No objects may be created from it.

Extends

Class "`oldClass`", directly.

Methods

No methods defined with class "bayesnet" in the signature.

Note

~~further notes~~

Author(s)

~~who you are~~

References

~put references to the literature/web site here ~

Examples

```
##---- Should be DIRECTLY executable !! ----
```

MIMbase

Submit commands to MIM and get the result back in R

Description

mim.cmd is the core function for communicating with MIM from within R. MIM commands are given as text strings and the results from submitting these commands in MIM are returned (as a vector of strings). Several functions exist for processing these results in sensible ways. See below.

Invoking mcm() gives a direct interface to MIM such that MIM commands can be entered directly. The output printed by MIM is printed the same way in the R console.

Usage

```
mim.cmd(cmd, look.nice = TRUE, return.look.nice=FALSE, version='R')
mcm()
helpmim()
```

Arguments

cmd	A (vector of) strings of MIM commands
look.nice	When TRUE the result of the last MIM command is printed on the screen and returned as a list of strings, one string per line of output. When FALSE the result is returned as a vector of strings (thereby enabling processing of the results in R).
return.look.nice	When TRUE the result of the last MIM command is returned as a list of strings, one string per line of output.
version	If set to 'S' then the function also should work with Splus

Details

To exit the `mcm()` function, type `quit`, `end`, `exit`, `q` or `e` at the prompt. Note that this will not terminate the MIM program. The `helpmim()` function invokes the help pages of the MIM program.

Value

`mim.cmd` returns a vector or a string, whereas `mcm` returns `NULL`

Note

Before using `mimR`, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, *An Introduction to Graphical Modelling*, Second Edition, Springer Verlag, 2000

See Also

[mim](#)

Examples

```
mim.cmd("fact a2 b2 c2; statread abc")
mim.cmd("25 2 17 8 14 9 6 8 !")
mim.cmd("mod ab,ac,bc; fit")
##mcm()
```

MIMfit

Function to fit model in MIM

Description

Fits model in MIM either directly or using an EM–algorithm in the case of incomplete data

Usage

```
## S3 method for class 'mim':
fit(m, arg=NULL, ...)
```

Arguments

<code>m</code>	A mim model object
<code>arg</code>	Additional MIM arguments controlling the fitting algorithm
<code>...</code>	Additional arguments, currently not used.

Details

The S option enables the user to supply start values for the missing data: as with the R option these are used to calculate the sufficient statistics, and thence the initial parameter estimates. First start values are entered, and then these are overwritten with an asterix (missing value). For an example of how to do this, see the examples below.

The start values can be entered using EditData in MIM: first enter the desired value, and then overwrite this with an asterix (missing value). Check using Print E in MIM that the values have been correctly entered.

Value

Returns a fitted mim object

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, *An Introduction to Graphical Modelling*, Springer Verlag, 2002

Examples

```
data(math)
math$L <- factor(NA, levels=1:2)
gmd.math <- as.gmData(math)
latent(gmd.math) <- "L"
m1 <- mim("..", data=gmd.math, fit=FALSE)
m2 <- editmim(m1, deleteEdge=paste(names(math)[1:5], collapse=':'))
m2f <- fit(m2, "er")
imputeMissing()
d.imp <- retrieveData()
```

Description

Create undirected MIM models

Usage

```
mim(mimFormula, data, fit=TRUE, marginal=data$name)
```

Arguments

<code>mimFormula</code>	A model formula following the MIM syntax. Long variable names are allowed however. See 'details'. The formula can be given either with a tilde or as a string
<code>fit</code>	Should the model be fitted if possible
<code>data</code>	A <code>gmData</code> object
<code>marginal</code>	Can be used for specifying only a subset of the variables in connection with a main effects, a saturated and a homogeneous saturated model

Details

A `mim.formula` can be "Sex+Drug/Sex:W1+Drug:W1+Sex:W2+Drug:W2/Sex:W1:W2+Drug:W1:W2".
 A `mimFormula` can also be "." (the main effects (the independence) model), ".." (the saturated model) or "..h" (the homogeneous saturated model). See 'examples'.

Value

A `mim` model object

Note

Before using `mimR`, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, An Introduction to Graphical Modelling, Springer Verlag, 2002

See Also

[as.gmData](#)

Examples

```
data(rats)
gmd.rats <- as.gmData(rats)
valueLabels(gmd.rats)
observations(gmd.rats)

m1 <- mim("Sex:Drug/Sex:Drug:W1+Sex:Drug:W2/W1:W2", data=gmd.rats)
summary(m1)
mlf <- fit(m1)
summary(mlf)

m.main <- mim(".", data=gmd.rats)
m.sat <- mim("..", data=gmd.rats)
m.hsats <- mim("..h", data=gmd.rats)
```

```

summary(m.main);
summary(m.sat);
summary(m.hsar)

edges(m.hsar)
nodes(m.hsar)

m.main <- mim(".", data=gmd.rats, marginal=c("Sex", "Drug", "W1"))
m.sat <- mim("..", data=gmd.rats, marginal=c("Sex", "Drug", "W1"))
m.hsar <- mim"..h", data=gmd.rats, marginal=c("Sex", "Drug", "W1"))

plot(m.hsar)

m.main <- fit(mim(".", data=gmd.rats))
m.sat <- fit(mim("..", data=gmd.rats))
m.hsar <- fit(mim"..h", data=gmd.rats))

summary(m.main);
summary(m.sat);
summary(m.hsar)

# To generate an nth order hierarchical log-linear model for discrete
# data you can do

data(HairEyeColor)
mim(nthOrderModel(names(dimnames(HairEyeColor)),order=2),data=as.gmData(HairEyeColor))

```

Description

This package provides an interface to the MIM program for inference in mixed graphical models.

Details

- mimR is available on Windows platforms only.
- mimR requires that the MIM program is installed on the computer.
- MIM can be downloaded from <http://www.hypergraph.dk>.
- mimR requires that the R package 'RDCOMClient' is installed.
- For information about required versions of R and MIM, please see the information presented when mimR is loaded.
- mimR will automatically start the MIM program if not already running. However, mimR sometimes runs more smoothly if the user starts MIM manually.
- The mimR package comes with a small users manual.
- To start using mimR you must do `library(mimR)`

The package is intended as a contribution to the gR-project described by Lauritzen (2002).

Authors

Søren Højsgaard, Biometry Research Unit, Danish Institute of Agricultural Sciences, DK-8830 Tjele, Denmark

Acknowledgements

Thanks to David Edwards, the creator of the MIM program.

References

Lauritzen, S. L. (2002). gRaphical Models in R. *R News*, 3(2)39.

mimRprint-MIM *Print methods for objects in mimR*

Description

Print methods for objects in mimR

Usage

```
## S3 method for class 'mim':  
print(x, ...)  
properties(object)
```

Arguments

x	An object of appropriate class
object	a mim object
...	Has no effect

Value

x

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, An Introduction to Graphical Modelling, Second Edition, Springer Verlag, 2000

modelTest	<i>Compare two (nested) mim models</i>
-----------	--

Description

Compare two nested mim models or test a mim model against saturated model.

Usage

```
## S3 method for class 'mim':  
modelTest(m1, m2 = NULL)
```

Arguments

m1	mim model object
m2	mim model object

Value

A list of test statistics

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, An Introduction to Graphical Modelling, Second Edition, Springer Verlag, 2000

See Also

[testdelete](#)

nthOrderModel	<i>Create generating class with nth order interactions for log-linear model</i>
---------------	---

Description

Creates generating class with nth order interactions for log-linear model

Usage

```
nthOrderModel(variables, order = 2)
```

Arguments

variables	A vector of variables
order	The maximum order of interactions in the model

Value

A string

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, An Introduction to Graphical Modelling, Springer Verlag, 2002

Examples

```
nthOrderModel(c("ss", "uu", "ii"), 2)
```

plot	<i>Plot MIM model</i>
------	-----------------------

Description

Plot MIM model.

Usage

```
## S3 method for class 'mim':  
plot(x, ...)
```

Arguments

x	A MIM model
...	Not used

Author(s)

~~who you are~~

See Also

[mim](#)

retrieveData	<i>Retrieve data from MIM engine</i>
--------------	--------------------------------------

Description

This function can be used for retrieving data from the MIM engine. It is useful in connection with latent variable models

Usage

```
retrieveData(arg="c")
```

Arguments

arg	"c": the raw data (using value labels), "d": the raw data (using levels) , "e": the raw data (showing missing values)
-----	---

Value

A data frame

Note

Before using `mimR`, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, *An Introduction to Graphical Modelling*, Springer Verlag, 2002

See Also

See Also as [fit.mim](#)

stepwise

Stepwise model selection in MIM

Description

Functions to do stepwise model selection in MIM to achieve a new model object.

Usage

```
## S3 method for class 'mim':
stepwise(object, arg = NULL, critlevel=NULL, infconstant=NULL, ...)
```

Arguments

<code>object</code>	A <code>mimModel</code> object
<code>arg</code>	Stepwise options to MIM
<code>critlevel</code>	Set the critical level for the model selection. Default is 0.05
<code>infconstant</code>	Penalizing parameter used when model selection is based on information criteria, see 'details' below.
<code>...</code>	Additional arguments, currently not used

Details

Setting `arg` to contain "A" leads to model selection by AIC, i.e. the model with the smallest value of $-2\log Q - 2 * p$ is chosen. With BIC, the model with the smallest value of $-2\log Q - \log(n) * p$ is chosen. Setting `infconstant` to some value `k` AND setting `arg` to contain "A" will lead to selecting the model with the smallest value of $-2\log Q - k * p$.

Value

A `mim` model object

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, *An Introduction to Graphical Modelling*, Springer Verlag, 2002

Examples

```
data(carcass)
gmd.carc <- as.gmData(carcass)

m.main <- fit(mim(".", data=gmd.carc))
m.sat <- fit(mim("..", data=gmd.carc))

m.main <- mim(".", data=gmd.carc)
m.sat <- mim("..", data=gmd.carc)

m.m <- stepwise(m.main, "f") # forward
m.s <- stepwise(m.sat, "s") # backward, exact tests
```

toMIM

Wrapper for submitting data and models to MIM

Description

Function for submitting data and models to MIM. The function is primarily intended for internal use

Usage

```
toMIM(data)
```

Arguments

```
data      ~~Describe data here~~
```

Value

NULL

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, *An Introduction to Graphical Modelling*, Second Edition, Springer Verlag, 2000

variableType

Information about variables in MIM model

Description

Retrieve information about variables in MIM model

Usage

```
variableType(object)
```

Arguments

object A mim model object

Value

variableType() returns either "discrete", "continuous" or "mixed". is.discrete() and is.continuous() return TRUE or FALSE

Note

Before using mimR, make sure that the MIM program is running.

Author(s)

Søren Højsgaard, sorenh@agrsci.dk

References

David Edwards, *An Introduction to Graphical Modelling*, Second Edition, Springer Verlag, 2000

Index

- *Topic **classes**
 - mim-class, 10
- *Topic **graphs**
 - mimR, 15
- *Topic **hplot**
 - display, 4
 - plot, 19
- *Topic **methods**
 - edges-methods, 5
- *Topic **models**
 - cliques, 3
 - editMIM, 5
 - fitted.mim, 7
 - fixEdges, 8
 - imputeMissing, 9
 - MIMbase, 11
 - MIMfit, 12
 - MIMmodels, 13
 - mimR, 15
 - modelTest, 17
 - nthOrderModel, 18
 - stepwise, 20
 - toMIM, 21
 - variableType, 22
- *Topic **multivariate**
 - mimR, 15
- *Topic **print**
 - mimRprint-MIM, 16
- *Topic **utilities**
 - cellCounts, 2
 - retrieveData, 19
- as.gmData, 14
- cellCounts, 2
- chainmim (MIMmodels), 13
- cliques, 3
- cliques, mim-method (cliques), 3
- deviance.mim (MIMmodels), 13
- DF (MIMmodels), 13
- display, 4
- display, mim-method (display), 4
- display.mim (display), 4
- edges (edges-methods), 5
- edges, mim, ANY-method (edges-methods), 5
- edges-methods, 5
- editMIM, 5
- editmim (editMIM), 5
- empCov (cellCounts), 2
- fit.mim, 9, 20
- fit.mim (MIMfit), 12
- fitted.mim, 7
- fixEdges, 8
- fixSet (fixEdges), 8
- help, 3
- helpmim (MIMbase), 11
- imputeMissing, 9
- is.continuous (variableType), 22
- is.discrete (variableType), 22
- likelihood (MIMmodels), 13
- mcm (MIMbase), 11
- mim, 5, 12, 19
- mim (MIMmodels), 13
- mim-class, 10
- mim.cmd (MIMbase), 11
- MIMbase, 11
- MIMfit, 12
- MIMmodels, 13
- mimR, 15
- mimRprint-MIM, 16
- modelInfo (fitted.mim), 7
- modelTest, 6, 17

nodes (*edges-methods*), 5
nodes, *mim-method (edges-methods)*,
5
nodes-methods (*edges-methods*), 5
nthOrderModel, 18

oldClass, 10

plot, 19
print.mim (*mimRprint-MIM*), 16
print.modelInfo (*fitted.mim*), 7
properties (*mimRprint-MIM*), 16

retrieveData, 19
retrieveMIMvalues (*editMIM*), 5

simulate, 8
stepwise, 20
stepwise.mim, 9
summary.mim (*mimRprint-MIM*), 16

testdelete, 17
testdelete (*editMIM*), 5
toMIM, 21

update.mim (*editMIM*), 5

variableType, 22