Package 'ggm'

July 25, 2025

Title Graphical Markov Models with Mixed Graphs

Version 2.5.2

Depends R (>= 3.6.0), methods

Imports BiocManager, graph, igraph

Description Provides functions for defining
mixed graphs containing three types of edges, directed,
undirected and bi-directed, with possibly multiple edges.
These graphs are useful because they capture fundamental
independence structures in multivariate distributions
and in the induced distributions after marginalization
and conditioning.
The package is especially concerned with Gaussian graphical
models for
(i) ML estimation for directed acyclic graphs, undirected and
bi-directed graphs and ancestral graph models
(ii) testing several conditional independencies
(iii) checking global identification of DAG Gaussian models
with one latent variable
(iv) testing Markov equivalences and generating Markov
equivalent graphs of specific types.

License GPL-2

Encoding UTF-8

LazyData true

NeedsCompilation no

Repository CRAN

Author Giovanni M. Marchetti [aut, cre] (ORCID:
<https://orcid.org/0000-0003-4413-9349>),
Mathias Drton [aut] (ORCID: <https://orcid.org/0000-0001-5614-3025>),
Kayvan Sadeghi [aut] (ORCID: <https://orcid.org/0000-0001-7314-744X>)

Maintainer Giovanni M. Marchetti <giovanni.marchetti@unifi.it>

Date/Publication 2025-07-25 14:10:02 UTC

Contents

`adjMatrix` *Adjacency matrix of a graph*

Description

Transforms the "edge matrix" of a graph into the adjacency matrix.

Usage

```
adjMatrix(A)
```

Arguments

`A`	a square matrix representing the edge matrix of a graph.

Details

Given the edge matrix A of a graph, this can be transformed into an adjacency matrix E with the formula $E = (A - I)^T$.

Value

E	the adjacency matrix of the graph.

Author(s)

Giovanni M. Marchetti

See Also

edgematrix

Examples

```
amat <- DAG(y ~ x+z, z~u+v)
E <- edgematrix(amat)
adjMatrix(E)
```

AG *Ancestral graph*

Description

AG generates and plots ancestral graphs after marginalization and conditioning.

Usage

```
AG(amat,M=c(),C=c(),showmat=TRUE,plot=FALSE, plotfun = plotGraph, ...)
```

Arguments

amat	An adjacency matrix, or a graph that can be of class graphNEL-class or an igraph object, or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
M	A subset of the node set of a that is going to be marginalized over
C	Another disjoint subset of the node set of a that is going to be conditioned on.
showmat	A logical value. TRUE (by default) to print the generated matrix.
plot	A logical value, FALSE (by default). TRUE to plot the generated graph.
plotfun	Function to plot the graph when plot == TRUE. Can be plotGraph (the default) or drawGraph.
...	Further arguments passed to plotfun.

Value

A matrix that is the adjacency matrix of the generated graph. It consists of 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Richardson, T.S. and Spirtes, P. (2002). Ancestral graph Markov models. *Annals of Statistics*, 30(4), 962-1030.

Sadeghi, K. (2013). Stable mixed graphs. *Bernoulli* 19(5B), 2330–2358.

See Also

MAG, RG, SG

Examples

```
##The adjacency matrix of a DAG
ex<-matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,
             0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,
             0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,0,
             1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,
             0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0),16,16,byrow=TRUE)
M <- c(3,5,6,15,16)
C <- c(4,7)
AG(ex, M, C, plot = TRUE)
```

`allEdges` *All edges of a graph*

Description

Finds the set of edges of a graph. That is the set of undirected edges if the graph is undirected and the set of arrows if the graph is directed.

Usage

```
allEdges(amat)
```

Arguments

`amat`	a square Boolean matrix, with dimnames, the adjacency matrix of a graph.

Value

a matrix with two columns. Each row of the matrix is a pair of indices indicating an edge of the graph. If the graph is undirected, then only one of the pairs $(i, j), (j, i)$ is reported.

Author(s)

Giovanni M. Marchetti

See Also

`cycleMatrix`

Examples

```
## A UG graph
allEdges(UG(~ y*v*k +v*k*d+y*d))

## A DAG
allEdges(DAG(u~h+o+p, h~o, o~p))
```

anger *Anger data*

Description

Anger data

Usage

```
data(anger)
```

Format

A covariance matrix for 4 variables measured on 684 female students.

X anxiety state

Y anger state

Z anxiety trait

U anger trait

Details

Trait variables are viewed as stable personality characteristics, and state variables denote behaviour in specific situations. See Cox and Wermuth (1996).

References

Cox, D. R. and Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman and Hall.

Cox, D.R. and Wermuth, N. (1990). *An approximation to maximum likelihood estimates in reduced models*. 77(4), 747-761.

Examples

```
# Fit a chordless 4-cycle model
data(anger)
G = UG(~ Y*X + X*Z + Z*U + U*Y)
fitConGraph(G,anger, 684)
```

basiSet *Basis set of a DAG*

Description

Finds a basis set for the conditional independencies implied by a directed acyclic graph, that is a minimal set of independencies that imply all the other ones.

Usage

```
basiSet(amat)
```

Arguments

amat	a square matrix with dimnames representing the adjacency matrix of a DAG.

Details

Given a DAG and a pair of non adjacent nodes (i, j) such that j has higher causal order than i, the set of independency statements i independent of j given the union of the parents of both i and j is a basis set (see Shipley, 2000). This basis set has the property to lead to independent test statistics.

Value

a list of vectors representing several conditional independence statements. Each vector contains the names of two non adjacent nodes followed by the names of nodes in the conditioning set (which may be empty).

Author(s)

Giovanni M. Marchetti

References

Shipley, B. (2000). A new inferential test for path models based on directed acyclic graphs. *Structural Equation Modeling*, 7(2), 206–218.

See Also

shipley.test, dSep, DAG

Examples

```
## See Shipley (2000), Figure 2, p. 213
A <- DAG(x5~ x3+x4, x3~ x2, x4~x2, x2~ x1)
basiSet(A)
```

`bfsearch` *Breadth first search*

Description

Breadth-first search of a connected undirected graph.

Usage

```
bfsearch(amat, v = 1)
```

Arguments

`amat`	a symmetric matrix with dimnames specifying the adjacency matrix of the undirected graph
`v`	an integer, indicating the starting node of the search. Defaults to the first node.

Details

Breadth-first search is a systematic method for exploring a graph. The algorithm is taken from Aho, Hopcroft & Ullman (1983).

Value

`tree`	the edge matrix of the resulting spanning tree
`branches`	a matrix with two columns, giving the indices of the branches of the spanning tree
`chords`	a matrix with two columns, giving the indices of the chords of the spanning tree

Author(s)

Giovanni M. Marchetti

References

Aho, A.V., Hopcrtoft, J.E. & Ullman, J.D. (1983). *Data structures and algorithms.* Reading: Addison-Wesley.

Thulasiraman, K. & Swamy, M.N.S. (1992). *Graphs: theory and algorithms*. New York: Wiley.

See Also

`UG`, `findPath`, `cycleMatrix`

Examples

```
## Finding a spanning tree of the butterfly graph
bfsearch(UG(~ a*b*o + o*u*j))
## Starting from another node
bfsearch(UG(~ a*b*o + o*u*j), v=3)
```

---

binve *Inverts a marginal log-linear parametrization*

Description

Inverts a marginal log-linear parametrization.

Usage

```
binve(eta, C, M, G, maxit = 500, print = FALSE, tol = 1e-10)
```

Arguments

eta	a vector of dimension t-1 where t is the number of cells of a contingency table.
C	A contrast matrix.
M	A marginalization matrix.
G	G is the model matrix of the loglinear parameterization with no constant term.
maxit	an integer, specifying the maximum number of iterations. Default 500.
print	a logical value: if TRUE, prints the criterion after each cycle.
tol	A small value specifying the tolerance for the convergence criterion. Default: 1e-10.

Details

A marginal log-linear link is defined by $\eta = C(M \log p)$. See Bartolucci et al. (2007).

Value

A vector of probabilities p.

Note

From a Matlab function by A. Forcina, University of Perugia, Italy.

Author(s)

Antonio Forcina, Giovanni M. Marchetti

References

Bartolucci, F., Colombi, R. and Forcina, A. (2007). An extended class of marginal link functions for modelling contingency tables by equality and inequality constraints. Statist. Sinica 17, 691-711.

See Also

mat.mlogit

`blkdiag` *Block diagonal matrix*

Description

Block diagonal concatenation of input arguments.

Usage

```
blkdiag(...)
```

Arguments

`...`	Variable number of matrices `M1, M2, ...`.

Value

A block diagonal matrix `diag(M1, M2, ...)`.

Author(s)

Giovanni M. Marchetti

See Also

diag

Examples

```
X <- c(1,1,2,2); Z <- c(10, 20, 30, 40); A <- factor(c(1,2,2,2))
blkdiag(model.matrix(~X+Z), model.matrix(~A))
```

blodiag *Block diagonal matrix*

Description

Split a vector x into a block diagonal matrix.

Usage

```
blodiag(x, blo)
```

Arguments

x	A vector of length n.
blo	A vector of positive integers such that sum(blo) == n.

Value

A block-diagonal matrix with as many row as elements of blo and n columns. The vector x is split into length(blo) sub-vectors and these are the blocks of the resulting matrix.

Author(s)

Giovanni M. Marchetti

See Also

blkdiag, diag

Examples

```
blodiag(1:10, blo = c(2, 3, 5))
blodiag(1:10, blo = c(3,4,0,1))
```

checkIdent *Identifiability of a model with one latent variable*

Description

Checks four sufficient conditions for identifiability of a Gaussian DAG model with one latent variable.

Usage

```
checkIdent(amat, latent)
```

Arguments

amat	a square matrix with dimnames, representing the adjacency matrix of a DAG.
latent	an integer representing the latent variables among the nodes, or the name of the node.

Details

Stanghellini and Wermuth (2005) give some sufficient conditions for checking if a Gaussian model that factorizes according to a DAG is identified when there is one hidden node over which we marginalize. Specifically, the function checks the conditions of Theorem 1, (i) and (ii) and of Theorem 2 (i) and (ii).

Value

a vector of length four, indicating if the model is identified according to the conditions of theorems 1 and 2 in Stanghellini & Wermuth (2005). The answer is TRUE if the condition holds and thus the model is globally identified or FALSE if the condition fails, and thus we do not know if the model is identifiable.

Author(s)

Giovanni M. Marchetti

References

Stanghellini, E. & Wermuth, N. (2005). On the identification of path-analysis models with one hidden variable. *Biometrika*, 92(2), 337-350.

See Also

isGident, InducedGraphs

Examples

```
## See DAG in Figure 4 (a) in Stanghellini & Wermuth (2005)
d <- DAG(y1 ~ y3, y2 ~ y3 + y5, y3 ~ y4 + y5, y4 ~ y6)
checkIdent(d, "y3")  # Identifiable
checkIdent(d, "y4")  # Not identifiable?

## See DAG in Figure 5 (a) in Stanghellini & Wermuth (2005)
d <- DAG(y1 ~ y5+y4, y2 ~ y5+y4, y3 ~ y5+y4)
checkIdent(d, "y4")  # Identifiable
checkIdent(d, "y5")  # Identifiable

## A simple function to check identifiability for each node

is.ident <- function(amat){
### Check suff. conditions on each node of a DAG.
   p <- nrow(amat)
   ## Degrees of freedom
```

```
  df <- p*(p+1)/2 - p  - sum(amat==1) - p + 1
if(df <= 0)
    warning(paste("The degrees of freedom are ", df))
 a <- rownames(amat)
 for(i in a) {
   b <- checkIdent(amat, latent=i)
   if(TRUE %in% b)
     cat("Node", i, names(b)[!is.na(b)], "\n")
   else
     cat("Unknown.\n")
 }
}
```

---

cmpGraph *The complementary graph*

Description

Finds the complementary graph of an undirected graph.

Usage

```
cmpGraph(amat)
```

Arguments

amat	the adjacency matrix of an undirected graph

Details

The complementary graph of an UG is the graph that has the same set of nodes and an undirected edge connecting i and j whenever there is not an (i, j) edge in the original UG.

Value

the edge matrix of the complementary graph.

Author(s)

Giovanni M. Marchetti

References

Lauritzen, S. (1996). *Graphical models*. Oxford: Clarendon Press.

See Also

UG, DAG

Examples

```
## A chordless four-cycle
four <- UG(~ a*b + b*d + d*e + e*a)
four
cmpGraph(four)
```

---

conComp *Connectivity components*

Description

Finds the connectivity components of a graph.

Usage

```
conComp(amat, method)
```

Arguments

amat	a square matrix with dimnames, the adjacency matrix of an UG.
method	an integer 1 or 2 to choose the method used to find the components. Method 2 is more efficient for large graphs.

Value

an integer vector representing a partition of the set of nodes.

Author(s)

Giovanni M. Marchetti

References

Lauritzen, S. (1996). *Graphical models*. Oxford: Clarendon Press.

See Also

UG

Examples

```
## three connected components
conComp(UG(~a*c+c*d+e+g*o*u))
## a connected graph
conComp(UG(~ a*b+b*c+c*d+d*a))
```

`correlations` *Marginal and partial correlations*

Description

Computes a correlation matrix with ones along the diagonal, marginal correlations in the lower triangle and partial correlations given all remaining variables in the upper triangle.

Usage

```
correlations(x)
```

Arguments

`x`	a square symmetric matrix, a covariance matrix, or a data.frame for n observations and p variables.

Value

a square correlation matrix with marginal correlations (lower triangle) and partial correlations (upper triangle).

Author(s)

Giovanni M. Marchetti

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

See Also

`parcor`, `cor`

Examples

```
## See Table 6.1 in Cox & Wermuth (1996)
data(glucose)
correlations(glucose)
```

`cycleMatrix` *Fundamental cycles*

Description

Finds the matrix of fundamental cycles of a connected undirected graph.

Usage

```
cycleMatrix(amat)
```

Arguments

`amat`	a symmetric matrix with dimnames denoting the adjacency matrix of the undirected graph. The graph must be connected, otherwise the function returns an error message.

Details

All the cycles in an UG can be obtained from combination (ring sum) of the set of fundamental cycles. The matrix of fundamental cycles is a Boolean matrix having as rows the fundamental cycles and as columns the edges of the graph. If an entry is one then the edge associated to that column belongs to the cycle associated to the row.

Value

a Boolean matrix of the fundamental cycles of the undirected graph. If there is no cycle the function returns `NULL`.

Note

This function is used by `isGident`. The row sum of the matrix gives the length of the cycles.

Author(s)

Giovanni M. Marchetti

References

Thulasiraman, K. & Swamy, M.N.S. (1992). *Graphs: theory and algorithms*. New York: Wiley.

See Also

`UG`, `findPath`, `fundCycles`, `isGident`, `bfsearch`

Examples

```
## Three cycles
cycleMatrix(UG(~a*b*d+d*e+e*a*f))
## No cycle
 cycleMatrix(UG(~a*b))
## two cycles: the first is even and the second is odd
cm <- cycleMatrix(UG(~a*b+b*c+c*d+d*a+a*u*v))
apply(cm, 1, sum)
```

DAG *Directed acyclic graphs (DAGs)*

Description

A simple way to define a DAG by means of regression model formulae.

Usage

```
DAG(..., order = FALSE)
```

Arguments

`...`	a sequence of model formulae
`order`	logical, defaulting to `FALSE`. If `TRUE` the nodes of the DAG are permuted according to the topological order. If `FALSE` the nodes are in the order they first appear in the model formulae (from left to right).

Details

The DAG is defined by a sequence of recursive regression models. Each regression is defined by a model formula. For each formula the response defines a node of the graph and the explanatory variables the parents of that node. If the regressions are not recursive the function returns an error message.

Some authors prefer the terminology acyclic directed graphs (ADG).

Value

the adjacency matrix of the DAG, i.e. a square Boolean matrix of order equal to the number of nodes of the graph and a one in position (i, j) if there is an arrow from i to j and zero otherwise. The rownames of the adjacency matrix are the nodes of the DAG.

If `order = TRUE` the adjacency matrix is permuted to have parents before children. This can always be done (in more than one way) for DAGs. The resulting adjacency matrix is upper triangular.

Note

The model formulae may contain interactions, but they are ignored in the graph.

Author(s)

G. M. Marchetti

References

Lauritzen, S. (1996). *Graphical models*. Oxford: Clarendon Press.

See Also

UG, topSort, edgematrix, fitDag

Examples

```
## A Markov chain
DAG(y ~ x, x ~ z, z ~ u)

## Another DAG
DAG(y ~ x + z + u, x ~ u, z ~ u)

## A DAG with an isolated node
DAG(v ~ v, y ~ x + z, z ~ w + u)

## There can be repetitions
DAG(y ~ x + u + v, y ~ z, u ~ v + z)

## Interactions are ignored
DAG(y ~ x*z + z*v, x ~ z)

## A cyclic graph returns an error!
## Not run: DAG(y ~ x, x ~ z, z ~ y)

## The order can be changed
DAG(y ~ z, y ~ x + u + v,  u ~ v + z)

## If you want to order the nodes (topological sort of the DAG)
DAG(y ~ z, y ~ x + u + v,  u ~ v + z, order=TRUE)
```

`derived` *Data on blood pressure body mass and age*

Description

Raw data on blood pressure, body mass and age on 44 female patients, and covariance matrix for derived variables.

Usage

```
data(derived)
```

Format

A list containing a dataframe `raw` with 44 lines and 5 columns and a symmetric 4x4 covariance matrix `S`.

The following is the description of the variables in the dataframe `raw`

`Sys` Systolic blood pressure, in mm Hg

`Dia` Diastolic blood pressure, in mm Hg

`Age` Age of the patient, in years

`Hei` Height, in cm

`Wei` Weight, in kg

The following is the description of the variables for the covariance matrix `S`.

`Y` Derived variable `Y=log(Sys/Dia)`

`X` Derived variables `X=log(Dia)`

`Z` Body mass index `Z=Wei/(Hei/100)^2`

`W` Age

References

Wermuth N. and Cox D.R. (1995). Derived variables calculated from similar joint responses: some characteristics and examples. *Computational Statistics and Data Analysis*, 19, 223-234.

Examples

```
# A DAG model with a latent variable U
G = DAG(Y ~ Z + U, X ~ U + W, Z ~ W)

data(derived)

# The model fitted using the derived variables
out = fitDagLatent(G, derived$S, n = 44, latent = "U")

# An ancestral graph model marginalizing over U
H = AG(G, M = "U")

# The ancestral graph model fitted obtaining the
# same result
out2 = fitAncestralGraph(H, derived$S, n = 44)
```

DG *Directed graphs*

Description

Defines the adjacency of a directed graph.

Usage

```
DG(...)
```

Arguments

...	a sequence of model formulae

Details

The directed graph is defined by a sequence of models formulae. For each formula the response defines a node of the graph and its parents. The graph contains no loops.

Value

the adjacency matrix of the directed graph, i.e., a square Boolean matrix of order equal to the number of nodes of the graph and a one in position (i, j) if there is an arrow from i to j and zero otherwise. The dimnames of the adjacency matrix are the labels for the nodes of the graph.

Author(s)

G. M. Marchetti

References

Lauritzen, S. (1996). *Graphical models*. Oxford: Clarendon Press.

See Also

DAG, UG

Examples

```
## A DAG
DG(y ~ x, x ~ z, z ~ u)

## A cyclic directed graph
DG(y ~ x, x ~ z, z ~ y)

## A graph with two arrows between two nodes
DG(y ~ x, x ~ y)
```

```
## There can be isolated nodes
DG(y ~ x, x ~ x)
```

---

diagv *Matrix product with a diagonal matrix*

Description

Computes faster the product of a diagonal matrix times a full matrix.

Usage

```
diagv(v, M)
```

Arguments

v	A numeric vector specifying the elements on the diagonal of a matrix.
M	A numeric matrix compatible with the product $D_v M$.

Details

Computes $N = D_v M$ where D_v is diagonal avoiding the diag operator.

Value

A matrix N.

See Also

diag

Examples

```
v <- 1:1000
M <- matrix(runif(3000), 1000, 3)
dim(diagv(v, M))
```

drawGraph *Drawing a graph with a simple point and click interface.*

Description

Draw a graph from its adjacency matrix representation.

Usage

```
drawGraph(amat, coor = NULL, adjust = FALSE, alpha = 1.5,
              beta = 3, lwd = 1, ecol = "blue", bda = 0.1, layout = layout.auto)
```

Arguments

amat	the adjacency matrix representation of the graph. This can be an undirected graph, a directed acyclic graph or a mixed graph with at most a summary graph structure. See also plotGraph
.	
coor	an optional matrix of dimensions p x 2 where p is the number of vertices of the graph. If coor=NULL then the function chooses a default position for the nodes.
adjust	a logical value, defaults to FALSE. If TRUE the graph is plotted and the system waits until the mouse button is pressed (same behaviour of locator function.
alpha	a positive value between controlling the distance from the end of the edges to the nodes of the graph.
beta	a positive value controlling the distance of the labels of the variables from the nodes.
lwd	line width of the edges (default: 1).
ecol	color of the edges (default: "blue").
bda	bidirected edge arrow length (default: 0.1).
layout	The name of a function used to compute the (initial) layout of the graph. The default is layout.auto. This can be further adjusted if adjust is TRUE.

Details

The function is a very simple tool useful for displaying small graphs, with a rudimentary interface for moving nodes and edges of a given graph and adjusting the final plot. For better displays use **dynamicGraph** or **Rgraphviz** package in Bioconductor project.

Value

The function plots the graph with a initial positioning of the nodes, as specified by coor and remains in a waiting state. The position of each node can be shifted by pointing and clicking (with the first mouse button) close to the node. When the mouse button is pressed the node which is closer to the selected point is moved to that position. Thus, one must be careful to click closer to the selected

node than to any other node. The nodes can be moved to any position by repeating the previous operation. The adjustment process is terminated by pressing any mouse button other than the first.

At the end of the process, the function returns invisibly the coordinates of the nodes. The coordinates may be used later to redisplay the graph.

Author(s)

Giovanni M. Marchetti

References

dynamicGraph, **Rgraphwiz**, https://www.bioconductor.org.

GraphViz, Graph Visualization Project. AT&T Research. https://www.graphviz.org.

See Also

UG, DAG, makeMG, plotGraph

Examples

```
## A directed acyclic graph
d <- DAG(y1 ~ y2+y6, y2 ~ y3, y3 ~ y5+y6, y4 ~ y5+y6)
## Not run: drawGraph(d)

## An undirected graph
g <- UG(~giova*anto*armo + anto*arj*sara)
## Not run: drawGraph(d)

## An ancestral graph
ag <- makeMG(ug=UG(~y0*y1), dg=DAG(y4~y2, y2~y1), bg=UG(~y2*y3+y3*y4))
drawGraph(ag, adjust = FALSE)
drawGraph(ag, adjust = FALSE)

## A more complex example with coordinates: the UNIX evolution
xy <-
structure(c(5, 15, 23, 25, 26, 17, 8, 6, 6, 7, 39, 33, 23, 49,
19, 34, 13, 29, 50, 68, 70, 86, 89, 64, 81, 45, 64, 49, 64, 87,
65, 65, 44, 37, 64, 68, 73, 85, 83, 95, 84, 0, 7, 15, 27, 44,
37, 36, 20, 51, 65, 44, 64, 59, 73, 69, 78, 81, 90, 97, 89, 72,
85, 74, 62, 68, 59, 52, 48, 43, 50, 34, 21, 18, 5, 1, 10, 2,
11, 2, 1, 44), .Dim = c(41, 2), .Dimnames = list(NULL, c("x",
"y")))
Unix <- DAG(
                SystemV.3 ~ SystemV.2,
                SystemV.2 ~ SystemV.0,
                SystemV.0 ~ TS4.0,
                TS4.0 ~ Unix.TS3.0 + Unix.TS.PP + CB.Unix.3,
                PDP11.SysV ~ CB.Unix.3,
                CB.Unix.3 ~ CB.Unix.2,
                CB.Unix.2 ~ CB.Unix.1,
                Unix.TS.PP ~ CB.Unix.3,
```

```
                  Unix.TS3.0 ~ Unix.TS1.0 + PWB2.0 + USG3.0 + Interdata,
                  USG3.0 ~ USG2.0,
                  PWB2.0 ~ Interdata + PWB1.2,
                  USG2.0 ~ USG1.0,
                  CB.Unix.1 ~ USG1.0,
                  PWB1.2 ~ PWB1.0,
                  USG1.0 ~ PWB1.0,
                  PWB1.0 ~ FifthEd,
                  SixthEd ~ FifthEd,
                  LSX ~ SixthEd,
                  MiniUnix ~ SixthEd,
                  Interdata ~ SixthEd,
                  Wollongong ~ SixthEd,
                  SeventhEd ~ Interdata,
                  BSD1 ~ SixthEd,
                  Xenix ~ SeventhEd,
                  V32 ~ SeventhEd,
                  Uniplus ~ SeventhEd,
                  BSD3 ~ V32,
                  BSD2 ~ BSD1,
                  BSD4 ~ BSD3,
                  BSD4.1 ~ BSD4,
                  EigthEd ~ SeventhEd + BSD4.1,
                  NinethEd ~ EigthEd,
                  Ultrix32 ~ BSD4.2,
                  BSD4.2 ~ BSD4.1,
                  BSD4.3 ~ BSD4.2,
                  BSD2.8 ~ BSD4.1 + BSD2,
                  BSD2.9 ~ BSD2.8,
                  Ultrix11 ~ BSD2.8 + V7M + SeventhEd,
                  V7M ~ SeventhEd
                  )
drawGraph(Unix, coor=xy, adjust=FALSE)
# dev.print(file="unix.fig", device=xfig) # Edit the graph with Xfig
```

---

dSep *d-separation*

Description

Determines if in a directed acyclic graph two set of nodes a d-separated by a third set of nodes.

Usage

```
dSep(amat, first, second, cond)
```

Arguments

amat	a Boolean matrix with dimnames, representing the adjacency matrix of a directed acyclic graph. The function does not check if this is the case. See the function `isAcyclic`.

first	a vector representing a subset of nodes of the DAG. The vector should be a character vector of the names of the variables matching the names of the nodes in rownames(A). It can be also a numeric vector of indices.
second	a vector representing another subset of nodes of the DAG. The set second must be disjoint from first. The mode of second must match the mode of first.
cond	a vector representing a conditioning subset of nodes. The set cond must be disjoint from the other two sets and must share the same mode.

Details

d-separation is a fundamental concept introduced by Pearl (1988).

Value

a logical value. TRUE if first and second are d-separated by cond.

Author(s)

Giovanni M. Marchetti

References

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems.* San Mateo: Morgan Kaufmann.

Lauritzen, S. (1996). *Graphical models*. Oxford: Clarendon Press.

See Also

DAG, shipley.test, inducedCovGraph

Examples

```
## Conditioning on a transition node
dSep(DAG(y ~ x, x ~ z), first="y", second="z", cond = "x")
## Conditioning on a collision node (collider)
dSep(DAG(y ~ x, y ~ z), first="x", second="z", cond = "y")
## Conditioning on a source node
dSep(DAG(y ~ x, z ~ x), first="y", second="z", cond = "x")
## Marginal independence
dSep(DAG(y ~ x, y ~ z), first="x", second="z", cond = NULL)
## The DAG defined on p.~47 of Lauritzen (1996)
dag <- DAG(g ~ x, h ~ x+f, f ~ b, x ~ l+d, d ~ c, c ~ a, l ~ y, y ~ b)
dSep(dag, first="a", second="b", cond=c("x", "y"))
dSep(dag, first="a", second=c("b", "d"), cond=c("x", "y"))
```

edgematrix *Edge matrix of a graph*

Description

Transforms the adjacency matrix of a graph into an "edge matrix".

Usage

```
edgematrix(E, inv=FALSE)
```

Arguments

E	a square matrix, representing the adjacency matrix of a graph.
inv	a logical value.

Details

In some matrix computations for graph objects the adjacency matrix of the graph is transformed into an "edge matrix". Briefly, if E is the adjacency matrix of the graph, the edge matrix is $A = sign(E + I)^T = [a_{ij}]$. Thus, A has ones along the diagonal and if the graph has no edge between nodes i and j the entries $a_{i,j}$ and $a_{j,i}$ are both zero. If there is an arrow from j to i $a_{i,j} = 1$ and $a_{j,i} = 0$. If there is an undirected edge, both $a_{i,j} = a_{j,i} = 1$.

Value

A	the edge matrix of the graph. If TRUE the nodes are sorted in inverted topological order and the edge matrix is upper triangular.

Author(s)

Giovanni M. Marchetti

References

Wermuth, N. (2003). Analysing social science data with graphical Markov models. In: *Highly Structured Stochastic Systems.* P. Green, N. Hjort & T. Richardson (eds.), 47–52. Oxford: Oxford University Press.

See Also

adjMatrix

Examples

```
amat <- DAG(y ~ x+z, z~u+v)
amat
edgematrix(amat)
edgematrix(amat, inv=TRUE)
```

essentialGraph *Essential graph*

Description

Find the essential graph from a given directed acyclic graph.

Usage

```
essentialGraph(dagx)
```

Arguments

dagx	a square binary matrix, the adjacency matrix of a directed acyclic graph. The names of rows and of the columns are the nodes of the DAG.

Details

Converts a DAG into the Essential Graph. Is implemented by the algorithm by D.M.Chickering (1995).

Value

returns the adjacency matrix of the essential graph.

Author(s)

Giovanni M. Marchetti, from a MATLAB function by Tomas Kocka, AAU

References

Chickering, D.M. (1995). A transformational characterization of equivalent Bayesian network structures. *Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence*, Montreal, QU, 87-98. Morgan Kaufmann.

See Also

DAG, InducedGraphs

Examples

```
dag = DAG(U ~ Y+Z, Y~X, Z~X)
essentialGraph(dag)
```

findPath *Finding paths*

Description

Finds one path between two nodes of a graph.

Usage

```
findPath(amat, st, en, path = c())
```

Arguments

amat	a square Boolean matrix with dimnames, the adjacency matrix of a graph.
st	an integer, the starting node.
en	an integer, the ending node.
path	a vector of integers, used in recursive calls. At the beginning is NULL. It should not be modified by the user.

Value

a vector of integers, the sequence of nodes of a path, starting from st to en. In some graphs (spanning trees) there is only one path between two nodes.

Note

This function is not intended to be directly called by the user.

Author(s)

Giovanni M. Marchetti, translating the original **Python** code (see references).

References

Python Softftware Foundation (2003). Python Patterns — Implementing Graphs. https://www.python.org/doc/essays/graphs/.

See Also

fundCycles

Examples

```
## A (single) path on a spanning tree
findPath(bfsearch(UG(~ a*b*c + b*d + d*e+ e*c))$tree, st=1, en=5)
```

`fitAncestralGraph` *Fitting of Gaussian Ancestral Graph Models*

Description

Iterative conditional fitting of Gaussian Ancestral Graph Models.

Usage

```
fitAncestralGraph(amat, S, n, tol = 1e-06)
```

Arguments

`amat`	a square matrix, representing the adjacency matrix of an ancestral graph.
`S`	a symmetric positive definite matrix with row and col names, the sample covariance matrix.
`n`	the sample size, a positive integer.
`tol`	a small positive number indicating the tolerance used in convergence checks.

Details

In the Gaussian case, the models can be parameterized using precision parameters, regression coefficients, and error covariances (compare Richardson and Spirtes, 2002, Section 8). This function finds the MLE $\hat{\Lambda}$ of the precision parameters by fitting a concentration graph model. The MLE $\hat{B}$ of the regression coefficients and the MLE $\hat{\Omega}$ of the error covariances are obtained by iterative conditional fitting (Drton and Richardson, 2003, 2004). The three sets of parameters are combined to the MLE $\hat{\Sigma}$ of the covariance matrix by matrix multiplication:

$$\hat{\Sigma} = \hat{B}^{-1}(\hat{\Lambda} + \hat{\Omega})\hat{B}^{-T}.$$

Note that in Richardson and Spirtes (2002), the matrices Λ and Ω are defined as submatrices.

Value

`Shat`	the fitted covariance matrix.
`Lhat`	matrix of the fitted precisions associated with undirected edges and vertices that do not have an arrowhead pointing at them.
`Bhat`	matrix of the fitted regression coefficients associated to the directed edges. Precisely said `Bhat` contains ones on the diagonal and the off-diagonal entry (i, j) equals the *negated* MLE of the regression coefficient for variable j in the regression of variable i on its parents. Note that this (i, j) entry in `Bhat` corresponds to a directed edge $j \to i$, and thus to a one as (j, i) entry in the adjacency matrix.
`Ohat`	matrix of the error covariances and variances of the residuals between regression equations associated with bi-directed edges and vertices with an arrowhead pointing at them.
`dev`	the 'deviance' of the model.
`df`	the degrees of freedom.
`it`	the iterations.

Author(s)

Mathias Drton

References

Drton, M. and Richardson, T. S. (2003). A new algorithm for maximum likelihood estimation in Gaussian graphical models for marginal independence. *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, 184-191.

Drton, M. and Richardson, T. S. (2004). Iterative Conditional Fitting for Gaussian Ancestral Graph Models. Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, Department of Statistics, 130-137.

Richardson, T. S. and Spirtes, P. (2002). Ancestral Graph Markov Models. *Annals of Statistics*. 30(4), 962-1030.

See Also

fitCovGraph, icf, makeMG, fitDag

Examples

```
## A covariance matrix
"S" <- structure(c(2.93, -1.7, 0.76, -0.06,
                   -1.7, 1.64, -0.78, 0.1,
                    0.76, -0.78, 1.66, -0.78,
                   -0.06, 0.1, -0.78, 0.81), .Dim = c(4,4),
                  .Dimnames = list(c("y", "x", "z", "u"), c("y", "x", "z", "u")))
## The following should give the same fit.
## Fit an ancestral graph y -> x <-> z <- u
fitAncestralGraph(ag1 <- makeMG(dg=DAG(x~y,z~u), bg = UG(~x*z)), S, n=100)

## Fit an ancestral graph y <-> x <-> z <-> u
fitAncestralGraph(ag2 <- makeMG(bg= UG(~y*x+x*z+z*u)), S, n=100)

## Fit the same graph with fitCovGraph
fitCovGraph(ag2, S, n=100)

## Another example for the mathematics marks data

data(marks)
S <- var(marks)
mag1 <- makeMG(bg=UG(~mechanics*vectors*algebra+algebra*analysis*statistics))
fitAncestralGraph(mag1, S, n=88)

mag2 <- makeMG(ug=UG(~mechanics*vectors+analysis*statistics),
               dg=DAG(algebra~mechanics+vectors+analysis+statistics))
fitAncestralGraph(mag2, S, n=88) # Same fit as above
```

`fitConGraph` *Fitting a Gaussian concentration graph model*

Description

Fits a concentration graph (a covariance selection model).

Usage

```
fitConGraph(amat, S, n, cli = NULL, alg = 3, pri = FALSE, tol = 1e-06)
```

Arguments

`amat`	a square Boolean matrix representing the adjacency matrix of an UG
`S`	the sample covariance matrix
`n`	an integer denoting the sample size
`cli`	a list containing the cliques of the graph. The components of the list are character vectors containing the names of the nodes in the cliques. The names must match the names of the vertices. The knowledge of the cliques is not needed. If the cliques are not specified the function uses the algorithm by Hastie et al. (2009, p. 446).
`alg`	The algorithm used.
`pri`	If TRUE is verbose
`tol`	a small positive number indicating the tolerance used in convergence tests.

Details

The algorithms for fitting concentration graph models by maximum likelihood are discussed in Speed and Kiiveri (1986). If the cliques are known the function uses the iterative proportional fitting algorithm described by Whittaker (1990, p. 184). If the cliques are not specified the function uses the algorithm by Hastie et al. (2009, p. 631ff).

Value

`Shat`	the fitted covariance matrix.
`dev`	the 'deviance' of the model.
`df`	the degrees of freedom.
`it`	the iterations.

Author(s)

Giovanni M. Marchetti

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

Hastie, T., Tibshirani, R. and Friedman, J. (2009). *The elements of statistical learning.* Springer Verlag: New York.

Speed, T.P. and Kiiveri, H (1986). Gaussian Markov distributions over finite graphs. *Annals of Statistics*, 14, 138–150.

Whittaker, J. (1990). *Graphical models in applied multivariate statistics*. Chichester: Wiley.

See Also

UG, fitDag, marks

Examples

```
## A model for the mathematics marks (Whittaker, 1990)
data(marks)
## A butterfly concentration graph
G <- UG(~ mechanics*vectors*algebra + algebra*analysis*statistics)
fitConGraph(G, cov(marks), nrow(marks))
## Using the cliques

cl = list(c("mechanics", "vectors",  "algebra"), c("algebra", "analysis" , "statistics"))
fitConGraph(G, S = cov(marks), n = nrow(marks), cli = cl)
```

---

fitCovGraph *Fitting of Gaussian covariance graph models*

Description

Fits a Gaussian covariance graph model by maximum likelihood.

Usage

```
fitCovGraph(amat, S,n ,alg = "icf", dual.alg = 2, start.icf = NULL, tol = 1e-06)
```

Arguments

amat	A symmetric Booloean matrix with dimnames representing the adjacency matrix of an UG.
S	A symmetric positive definite matrix with dimnames, the sample covariance matrix.
n	A positive integer, the sample size.
alg	A character string, the algorithm used. If alg="icf" (the default) the algorithm is based on iterative conditional fitting (see Drton and Richardson, 2003). In this case the ML estimates are returned. If alg="dual" the algorithm is based on the dual likelihood (see Kauermann, 1996). The fitted values are an approximation of the ML estimates.

`dual.alg`	And integer equal to 1 or 2. It is used if `alg="dual"`. In this case a concentration graph model is fitted to the inverse of the sample covariance matrix, and `dual.alg` is passed to `fitConGraph` to specify the algorithm used in `fitConGraph`.
`start.icf`	A symmetric matrix used as starting value of the algorithm. If `start=NULL` the starting value is a diagonal matrix with diagonal entries equal to sample variances.
`tol`	A small positive number indicating the tolerance used in convergence tests.

Details

A covariance graph is an undirected graph in which the variables associated to two non-adjacent nodes are marginally independent. The edges of these models are represented by bi-directed edges (Drton and Richardson, 2003) or by dashed lines (Cox and Wermuth, 1996).

By default, this function gives the ML estimates in the covariance graph model, by iterative conditional fitting (Drton and Richardson, 2003). Otherwise, the estimates from a "dual likelihood" estimator can be obtained (Kauermann, 1996; Edwards, 2000, section 7.4).

Value

`Shat`	the fitted covariance matrix.
`dev`	the 'deviance' of the model.
`df`	the degrees of freedom.
`it`	the iterations.

Author(s)

Mathias Drton

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

Drton, M. and Richardson, T. S. (2003). A new algorithm for maximum likelihood estimation in Gaussian graphical models for marginal independence. *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, 184–191.

Kauermann, G. (1996). On a dualization of graphical Gaussian models. *Scandinavian Journal of Statistics*. 23, 105–116.

See Also

`fitConGraph`, `icf`

Examples

```
## Correlations among four strategies to cope with stress for
## 72 students. Cox & Wermuth (1996), p. 73.

data(stress)

## A chordless 4-cycle covariance graph
G <- UG(~ Y*X + X*U + U*V + V*Y)

fitCovGraph(G, S = stress, n=72)
fitCovGraph(G, S = stress, n=72, alg="dual")
```

`fitDag` *Fitting of Gaussian DAG models*

Description

Fits linear recursive regressions with independent residuals specified by a DAG.

Usage

```
fitDag(amat, S, n)
```

Arguments

`amat`	a square matrix with dimnames representing the adjacency matrix of the DAG
`S`	a symmetric positive definite matrix, the sample covariance matrix
`n`	an integer > 0, the sample size

Details

`fitDag` checks if the order of the nodes in adjacency matrix is the same of `S` and if not it reorders the adjacency matrix to match the order of the variables in `S`. The nodes of the adjacency matrix may form a subset of the variables in `S`.

Value

`Shat`	the fitted covariance matrix.
`Ahat`	a square matrix of the fitted regression coefficients. The entry `Ahat[i,j]` is minus the regression coefficient of variable `i` in the regression equation `j`. Thus there is a non zero partial regression coefficient `Ahat[i,j]` corresponding to each non zero value `amat[j,i]` in the adjacency matrix.
`Dhat`	a vector containing the partial variances of each variable given the parents.
`dev`	the 'deviance' of the model.
`df`	the degrees of freedom.

Author(s)

Giovanni M. Marchetti

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

See Also

DAG, swp.

Examples

```
dag <- DAG(y ~ x+u, x ~ z, z ~ u)
"S" <- structure(c(2.93, -1.7, 0.76, -0.06,
                   -1.7, 1.64, -0.78, 0.1,
                    0.76, -0.78, 1.66, -0.78,
                    -0.06, 0.1, -0.78, 0.81), .Dim = c(4,4),
         .Dimnames = list(c("y", "x", "z", "u"), c("y", "x", "z", "u")))
fitDag(dag, S, 200)
```

---

fitDagLatent *Fitting Gaussian DAG models with one latent variable*

Description

Fits by maximum likelihood a Gaussian DAG model where one of the nodes of the graph is latent and it is marginalised over.

Usage

```
fitDagLatent(amat, Syy, n, latent, norm = 1, seed,
             maxit = 9000, tol = 1e-06, pri = FALSE)
```

Arguments

amat	a square matrix with dimnames representing the adjacency matrix of the DAG.
Syy	a symmetric positive definite matrix, with dimnames, the sample covariance matrix of the observed variables. The set of the observed nodes of the graph must be a subset of the set of the names of the variables in Syy.
n	a positive integer, the sample size.
latent	the name of the latent variable.
norm	an integer, the kind of normalization of the latent variable. If norm=1, the latent is scaled to have unit variance. If norm=2, the latent is scaled to have unit partial variance given its parents.
seed	an integer, used by set.seed to specify a random starting point of the EM algorithm.

maxit	an integer denoting the maximum number of iterations allowed for the EM algorithm. If the convergence criterion is not satisfied within maxit iterations the algorithms stops and a warning message is returned.
tol	a small real value, denoting the tolerance used in testing convergence.
pri	logical, if pri=TRUE then the value of the deviance at each iteration is printed.

Details

For the EM algorithm used see Kiiveri (1987).

Value

Shat	the fitted covariance matrix of all the variables including the latent one. The latent variable is the last. If norm=1 then the variance of the latent variable is constrained to 1.
Ahat	a square matrix of the fitted regression coefficients. The entry Ahat[i,j] is minus the regression coefficient of variable i in the regression equation j. Thus there is a non zero partial regression coefficient Ahat[i,j] corresponding to each non zero value amat[j,i] in the adjacency matrix.
Dhat	a vector containing the partial variances of each variable given the parents. If norm=2 then the partial variance of the latent variable is constrained to 1.
dev	the 'deviance' of the model.
df	the degrees of freedom of the model.
it	the number of EM algorithm iterations at convergence.

Author(s)

Giovanni M. Marchetti

References

Kiiveri,H. T. (1987). An incomplete data approach to the analysis of covariance structures. *Psychometrika*, 52, 4, 539–554.

Joreskog, K.G. and Goldberger, A.S. (1975). Estimation of a model with multiple indicators and multiple causes of a single latent variable. *Journal of the American Statistical Association*, 10, 631–639.

See Also

fitDag, checkIdent

Examples

```
## data from Joreskog and Goldberger (1975)
V <- matrix(c(1,     0.36,   0.21,  0.10,  0.156, 0.158,
              0.36,  1,      0.265, 0.284, 0.192, 0.324,
              0.210, 0.265,  1,     0.176, 0.136, 0.226,
              0.1,   0.284,  0.176, 1,     0.304, 0.305,
```

```
              0.156, 0.192,  0.136, 0.304, 1,     0.344,
              0.158, 0.324,  0.226, 0.305, 0.344, 1),     6,6)
nod <- c("y1", "y2", "y3", "x1", "x2", "x3")
dimnames(V) <- list(nod,nod)
dag <- DAG(y1 ~ z, y2 ~ z, y3 ~ z, z ~ x1 + x2 + x3, x1~x2+x3, x2~x3)
fitDagLatent(dag, V, n=530, latent="z", seed=4564)
fitDagLatent(dag, V, n=530, latent="z", norm=2, seed=145)
```

`fitmlogit` *Multivariate logistic models*

Description

Fits a logistic regression model to multivariate binary responses.

Usage

```
fitmlogit(..., C = c(), D = c(), data, mit = 100, ep = 1e-80, acc = 1e-04)
```

Arguments

`...`	Model formulae of marginal logistic models for each response and for each association parameters (log-odds ratios).
`C`	Matrix of equality constraints.
`D`	Matrix of inequality cosntraints.
`data`	A data frame containing the responses and the explanatory variables.
`mit`	A positive integer: maximum number of iterations. Default: `100`.
`ep`	A tolerance used in the algorithm: default `1e-80`.
`acc`	A tolerance used in the algorithm: default `1e-4`.

Details

See Evans and Forcina (2011).

Value

`LL`	The maximized log-likelihood.
`be`	The vector of the Maximum likelihood estimates of the parameters.
`S`	The estimated asymptotic covariance matrix.
`P`	The estimated cell probabilities for each individual.

Author(s)

Antonio Forcina, Giovanni M. Marchetti

References

Evans, R.J. and Forcina, A. (2013). Two algorithms for fitting constrained marginal models. *Computational Statistics and Data Analysis*, 66, 1-7.

See Also

glm

Examples

```
data(surdata)
out1 <- fitmlogit(A ~X, B ~ Z, cbind(A, B) ~ X*Z, data = surdata)
out1$beta
out2 <- fitmlogit(A ~X, B ~ Z, cbind(A, B) ~ 1, data = surdata)
out2$beta
```

---

fundCycles *Fundamental cycles*

Description

Finds the list of fundamental cycles of a connected undirected graph.

Usage

```
fundCycles(amat)
```

Arguments

amat	a symmetric matrix with dimnames denoting the adjacency matrix of the undirected graph. The graph must be connected, otherwise the function returns an error message.

Details

All the cycles in an UG can be obtained from combination (ring sum) of the set of fundamental cycles.

Value

a list of matrices with two columns. Every component of the list is associated to a cycle. The cycle is described by a $k \times 2$ matrix whose rows are the edges of the cycle. If there is no cycle the function returns NULL.

Note

This function is used by cycleMatrix and isGident.

Author(s)

Giovanni M. Marchetti

References

Thulasiraman, K. & Swamy, M.N.S. (1992). *Graphs: theory and algorithms*. New York: Wiley.

See Also

UG,findPath, cycleMatrix, isGident,bfsearch

Examples

```
## Three fundamental cycles
fundCycles(UG(~a*b*d + d*e + e*a*f))
```

---

ggm *The package* ggm*: summary information*

Description

This package provides functions for defining, manipulating and fitting graphical Markov models with mixed graphs. It is intended as a contribution to the gR-project described by Lauritzen (2002).

For a tutorial illustrating the new functions in the package 'ggm' that deal with ancestral, summary and ribbonless graphs see Sadeghi and Marchetti (2012) in the references.

Functions

The main functions can be classified as follows.

- Functions for defining graphs (undirected, directed acyclic, ancestral and summary graphs): UG, DAG, makeMG, grMAT;
- Functions for doing graph operations (parents, boundary, cliques, connected components, fundamental cycles, d-separation, m-separation): pa, bd, conComp, fundCycles;
- Functions for testing independence statements and generating maximal graphs from non-maximal graphs: dSep, msep, Max;
- Function for finding covariance and concentration graphs induced by marginalization and conditioning: inducedCovGraph, inducedConGraph;
- Functions for finding multivariate regression graphs and chain graphs induced by marginalization and conditioning: inducedRegGraph, inducedChainGraph, inducedDAG;
- Functions for finding stable mixed graphs (ancestral, summary and ribbonless) after marginalization and conditioning: AG, SG, RG;
- Functions for fitting by ML Gaussian DAGs, concentration graphs, covariance graphs and ancestral graphs: fitDag, fitConGraph, fitCovGraph, fitAncestralGraph;
- Functions for testing several conditional independences:shipley.test;

- Functions for checking global identification of DAG Gaussian models with one latent variable (Stanghellini-Vicard's condition for concentration graphs, new sufficient conditions for DAGs): `isGident`, `checkIdent`;
- Functions for fitting Gaussian DAG models with one latent variable: `fitDagLatent`;
- Functions for testing Markov equivalences and generating Markov equivalent graphs of specific types: `MarkEqRcg`, `MarkEqMag`, `RepMarDAG`, `RepMarUG`, `RepMarBG`.

Authors

Giovanni M. Marchetti, Dipartimento di Statistica, Informatica, Applicazioni 'G. Parenti'. University of Florence, Italy

Mathias Drton, Department of Statistics, University of Washington, USA

Kayvan Sadeghi, Department of Statistics, Carnegie Mellon University, USA

Acknowledgements

Many thanks to Fulvia Pennoni for testing some of the functions, to Elena Stanghellini for discussion and examples and to Claus Dethlefsen and Jens Henrik Badsberg for suggestions and corrections. The function `fitConGraph` was corrected by Ilaria Carobbi. Helpful discussions with Steffen Lauritzen and Nanny Wermuth, are gratefully acknowledged. Thanks also to Michael Perlman, Thomas Richardson and David Edwards.

Giovanni Marchetti has been supported by MIUR, Italy, under grant scheme PRIN 2002, and Mathias Drton has been supported by NSF grant DMS-9972008 and University of Washington RRF grant 65-3010.

References

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

Sadeghi, K. and Marchetti, G.M. (2012). Graphical Markov models with mixed graphs in R. *The R Journal*, 4(2):65-73. https://journal.r-project.org/archive/2012/RJ-2012-015/RJ-2012-015.pdf

---

`glucose` *Glucose control*

Description

Data on glucose control of diabetes patients.

Usage

```
data(glucose)
```

Format

A data frame with 68 observations on the following 8 variables.

Y a numeric vector, Glucose control (glycosylated haemoglobin), values up to about 7 or 8 indicate good glucose control.

X a numeric vector, a score for knowledge about the illness.

Z a numeric vector, a score for fatalistic externality (mere chance determines what occurs).

U a numeric vector, a score for social externality (powerful others are responsible).

V a numeric vector, a score for internality (the patient is him or herself responsible).

W a numeric vector, duration of the illness in years.

A a numeric vector, level of education, with levels `-1`: at least 13 years of formal schooling, `1`: less then 13 years.

B a numeric vector, gender with levels `-1`: females, `1`: males.

Details

Data on 68 patients with fewer than 25 years of diabetes. They were collected at the University of Mainz to identify psychological and socio-economic variables possibly important for glucose control, when patients choose the appropriate dose of treatment depending on the level of blood glucose measured several times per day.

The variable of primary interest is `Y`, glucose control, measured by glycosylated haemoglobin. `X`, knowledge about the illness, is a response of secondary interest. Variables `Z`, `U` and `V` measure patients' type of attribution, called fatalistic externality, social externality and internality. These are intermediate variables. Background variables are `W`, the duration of the illness, `A` the duration of formal schooling and `B`, gender. The background variables `A` and `B` are binary variables with coding `-1`, `1`.

Source

Cox & Wermuth (1996), p. 229.

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

Examples

```
data(glucose)
## See Cox & Wermuth (1996), Figure 6.3 p. 140
coplot(Y ~ W | A, data=glucose)
```

grMAT *Graph to adjacency matrix*

Description

grMAT generates the associated adjacency matrix to a given graph.

Usage

```
grMAT(agr)
```

Arguments

agr	A graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).

Value

A matrix that consists 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

Examples

```
## Generating the adjacency matrix from a vector
exvec <-c ('b',1,2,'b',1,14,'a',9,8,'l',9,11,'a',10,8,
           'a',11,2,'a',11,10,'a',12,1,'b',12,14,'a',13,10,'a',13,12)
grMAT(exvec)
```

In *Indicator matrix*

Description

Finds the indicator matrix of the zeros of a matrix.

Usage

```
In(A)
```

Arguments

A	a matrix.

Details

The indicator matrix is a matrix of zeros and ones which has a zero element iff the corresponding element of A is (exactly) zero.

Value

a matrix of the same dimensions as A.

Author(s)

Giovanni M. Marchetti

References

Wermuth, N. & Cox, D.R. (2004). Joint response graphs and separation induced by triangular systems. *J.R. Statist. Soc. B*, 66, Part 3, 687-717.

See Also

DAG, inducedCovGraph, inducedConGraph

Examples

```
## A simple way to find the overall induced concentration graph
## The DAG on p. 198 of Cox & Wermuth (1996)
amat <- DAG(y1 ~ y2 + y3, y3 ~ y5, y4 ~ y5)
A <- edgematrix(amat)
In(crossprod(A))
```

InducedGraphs *Graphs induced by marginalization or conditioning*

Description

Functions to find induced graphs after conditioning on a set of variables and marginalizing over another set.

Usage

```
inducedCovGraph(amat, sel = rownames(amat), cond = NULL)
inducedConGraph(amat, sel = rownames(amat), cond = NULL)
inducedRegGraph(amat, sel = rownames(amat), cond = NULL)
inducedChainGraph(amat, cc=rownames(amat), cond = NULL, type="LWF")
inducedDAG(amat, order, cond = NULL)
```

Arguments

amat	a square Boolean matrix, the adjacency matrix of a directed acyclic graph. The names of rows and of the columns are the nodes of the DAG.
sel	a character vector representing a subset of selected variables. The elements of the vector must be a subset of the names of the nodes i.e. of rownames(A). By default sel is the set of the nodes of the DAG.
cond	a character vector representing the variables on which you want to condition. cond must be disjoint from sel and their union must be a subset of the set of nodes. The set difference between the set of nodes and the union of sel and cond are the variables over which we marginalize. cond may be the null vector (the default), meaning that you want to condition on the empty set.
cc	a list of character vectors specifying the chain components for the chain graph.
type	a string indicating the interpretation of the chain graph. It can be either "LWF" (Lauritzen, Wermuth, Frydenberg interpretation), "AMP" (Andersson, Madigan, Perlman interpretation) or "MRG" (Multivariate regression graph interpretation).
order	a character vector indicating the ordering of the vertices of a DAG (left to right, past to future).

Details

Given a directed acyclic graph representing a set of conditional independencies it is possible to obtain other graphs of conditional independence implied after marginalizingover and conditionig on sets of nodes. Such graphs are the covariance graph, the concentration graph, the multivariate regression graph and the chain graph with different interpretations (see Cox & Wermuth, 1996, 2004).

Value

inducedCovGraph returns the adjacency matrix of the covariance graph of the variables in set sel given the variables in set cond, implied by the original directed acyclic graph with adjacency matrix amat.

inducedConGraph returns the adjacency matrix of the concentration graph of the variables in set sel given the variables in set cond, implied by the original directed acyclic graph with adjacency matrix amat.

inducedRegGraph returns the adjacency matrix of the multivariate regression graph of the variables in set sel given the variables in set cond, implied by the original directed acyclic graph with adjacency matrix amat.

inducedChainGraph returns the adjacency matrix of the chain graph for the variables in chain components cc, given the variables in set cond, with interpretation specified by string type, implied by the original directed acyclic graph with adjacency matrix amat.

inducedDAG returns the adjacency matrix of the DAG with the ordering order, implied by the original directed acyclic graph with adjacency matrix amat.

Note

If sel is NULL the functions return the null matrix. If cond is NULL, the conditioning set is empty and the functions inducedConGraph and inducedCovGraph return the overall induced covariance or concentration matrices of the selected variables. If you do not specify sel you cannot specify a non NULL value of cond.

Author(s)

Giovanni M. Marchetti

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

Wermuth, N. & Cox, D.R. (2004). Joint response graphs and separation induced by triangular systems. *J.R. Statist. Soc. B*, 66, Part 3, 687-717.

See Also

DAG, UG,isAcyclic

Examples

```
## Define a DAG
dag <- DAG(a ~ x, c ~ b+d, d~ x)
dag
## Induced covariance graph of a, b, d given the empty set.
inducedCovGraph(dag, sel=c("a", "b", "d"), cond=NULL)

## Induced concentration graph of a, b, c given x
inducedConGraph(dag, sel=c("a", "b", "c"), cond="x")

## Overall covariance graph
inducedCovGraph(dag)

## Overall concentration graph
inducedConGraph(dag)

## Induced covariance graph of x, b, d given c, x.
inducedCovGraph(dag, sel=c("a", "b", "d"), cond=c("c", "x"))

## Induced concentration graph of a, x, c given d, b.
inducedConGraph(dag, sel=c("a", "x", "c"), cond=c("d", "b"))

## The DAG on p. 198 of Cox & Wermuth (1996)
dag <- DAG(y1~ y2 + y3, y3 ~ y5, y4 ~ y5)

## Cf. figure 8.7 p. 203 in Cox & Wermuth (1996)
inducedCovGraph(dag, sel=c("y2", "y3", "y4", "y5"), cond="y1")
inducedCovGraph(dag, sel=c("y1", "y2", "y4", "y5"), cond="y3")
inducedCovGraph(dag, sel=c("y1", "y2", "y3", "y4"), cond="y5")
```

```
## Cf. figure 8.8 p. 203 in Cox & Wermuth (1996)
inducedConGraph(dag, sel=c("y2", "y3", "y4", "y5"), cond="y1")
inducedConGraph(dag, sel=c("y1", "y2", "y4", "y5"), cond="y3")
inducedConGraph(dag, sel=c("y1", "y2", "y3", "y4"), cond="y5")

## Cf. figure 8.9 p. 204 in Cox & Wermuth (1996)
inducedCovGraph(dag, sel=c("y2", "y3", "y4", "y5"), cond=NULL)
inducedCovGraph(dag, sel=c("y1", "y2", "y4", "y5"), cond=NULL)
inducedCovGraph(dag, sel=c("y1", "y2", "y3", "y4"), cond=NULL)

## Cf. figure 8.10 p. 204 in Cox & Wermuth (1996)
inducedConGraph(dag, sel=c("y2", "y3", "y4", "y5"), cond=NULL)
inducedConGraph(dag, sel=c("y1", "y2", "y4", "y5"), cond=NULL)
inducedConGraph(dag, sel=c("y1", "y2", "y3", "y4"), cond=NULL)

## An induced regression graph
dag2 = DAG(Y ~ X+U, W ~ Z+U)
inducedRegGraph(dag2, sel="W",  cond=c("Y", "X", "Z"))

## An induced DAG
inducedDAG(dag2, order=c("X","Y","Z","W"))

## An induced multivariate regression graph
inducedRegGraph(dag2, sel=c("Y", "W"), cond=c("X", "Z"))

## An induced chain graph with LWF interpretation
dag3 = DAG(X~W, W~Y, U~Y+Z)
cc = list(c("W", "U"), c("X", "Y", "Z"))
inducedChainGraph(dag3, cc=cc, type="LWF")

## ... with AMP interpretation
inducedChainGraph(dag3, cc=cc, type="AMP")

## ... with multivariate regression interpretation
cc= list(c("U"), c("Z", "Y"), c("X", "W"))
inducedChainGraph(dag3, cc=cc, type="MRG")
```

`isAcyclic` *Graph queries*

Description

Checks if a given graph is acyclic.

Usage

```
isAcyclic(amat, method = 2)
```

Arguments

amat	a square Boolean matrix with dimnames, the adjacency matrix of a graph.
method	an integer 1 or 2 specifying the method used. If method=1 the function calls the function clusters in package igraph to find the strong components: two nodes v and w are in the same strong component iff there are directed paths from v to w and from w to v. If method=2 the function uses the ggm function transClos. Method 1 is faster.

Value

a logical value, TRUE if the graph is acyclic and FALSE otherwise.

Author(s)

David Edwards, Giovanni M. Marchetti

References

Aho, A.V., Hopcroft, J.E. & Ullman, J.D. (1983). *Data structures and algorithms.* Reading: Addison-Wesley.

Examples

```
## A cyclic graph
d <- matrix(0,3,3)
rownames(d) <- colnames(d) <- c("x", "y", "z")
d["x","y"] <- d["y", "z"] <- d["z", "x"] <- 1
## Test if the graph is acyclic
isAcyclic(d)
isAcyclic(d, method = 1)
```

---

isADMG *Acyclic directed mixed graphs*

Description

Check if it is an adjacency matrix of an ADMG

Usage

```
isADMG(amat)
```

Arguments

amat	An adjacency matrix.

Details

Checks if the following conditions must hold: (i) no undirected edge meets an arrowhead; (ii) no directed cycles;

Value

A logical value, `TRUE` if it is an ancestral graph and `FALSE` otherwise.

Author(s)

Giovanni M. Marchetti, Mathias Drton

References

Richardson, T. S. and Spirtes, P. (2002). Ancestral Graph Markov Models. *Annals of Statistics*, 30(4), 962–1030.

See Also

`makeMG`, `isADMG`

Examples

```
## Examples from Richardson and Spirtes (2002)
a1 <- makeMG(dg=DAG(a~b, b~d, d~c), bg=UG(~a*c))
isADMG(a1)    # Not an AG. (a2) p.969
a2 <- makeMG(dg=DAG(b ~ a, d~c), bg=UG(~a*c+c*b+b*d))          # Fig. 3 (b1) p.969
isADMG(a2)
```

`isAG` *Ancestral graph*

Description

Check if it is an adjacency matrix of an ancestral graph

Usage

```
isAG(amat)
```

Arguments

`amat`	An adjacency matrix.

Details

Checks if the following conditions must hold: (i) no undirected edge meets an arrowhead; (ii) no directed cycles; (iii) spouses cannot be ancestors. For details see Richardson and Spirtes (2002).

Value

A logical value, TRUE if it is an ancestral graph and FALSE otherwise.

Author(s)

Giovanni M. Marchetti, Mathias Drton

References

Richardson, T. S. and Spirtes, P. (2002). Ancestral Graph Markov Models. *Annals of Statistics*, 30(4), 962–1030.

See Also

makeMG, isADMG

Examples

```
## Examples from Richardson and Spirtes (2002)
a1 <- makeMG(dg=DAG(a~b, b~d, d~c), bg=UG(~a*c))
isAG(a1)    # Not an AG. (a2) p.969
a2 <- makeMG(dg=DAG(b ~ a, d~c), bg=UG(~a*c+c*b+b*d))          # Fig. 3 (b1) p.969
isAG(a2)
```

---

isGident *G-identifiability of an UG*

Description

Tests if an undirected graph is G-identifiable.

Usage

```
isGident(amat)
```

Arguments

amat	a symmetric matrix with dimnames representing the adjacency matrix of an undirected graph

Details

An undirected graph is said G-identifiable if every connected component of the complementary graph contains an odd cycle (Stanghellini and Wermuth, 2005). See also Tarantola and Vicard (2002).

Value

a logical value, TRUE if the graph is G-identifiable and FALSE if it is not.

Author(s)

Giovanni M. Marchetti

References

Stanghellini, E. & Wermuth, N. (2005). On the identification of path-analysis models with one hidden variable. *Biometrika*, 92(2), 337-350.

Stanghellini, E. (1997). Identification of a single-factor model using graphical Gaussian rules. *Biometrika*, 84, 241–244.

Tarantola, C. & Vicard, P. (2002). Spanning trees and identifiability of a single-factor model. *Statistical Methods & Applications*, 11, 139–152.

Vicard, P. (2000). On the identification of a single-factor model with correlated residuals. *Biometrika*, 87, 199–205.

See Also

UG, cmpGraph, cycleMatrix

Examples

```
## A not G-identifiable UG
G1 <- UG(~ a*b + u*v)
isGident(G1)
## G-identifiable UG
G2 <- UG(~ a + b + u*v)
isGident(G2)
## G-identifiable UG
G3 <- cmpGraph(UG(~a*b*c+x*y*z))
isGident(G3)
```

MAG *Maximal ancestral graph*

Description

MAG generates and plots maximal ancestral graphs after marginalisation and conditioning.

Usage

```
MAG(amat,M=c(),C=c(),showmat=TRUE,plot=FALSE, plotfun = plotGraph, ...)
```

Arguments

amat	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
M	A subset of the node set of a that is going to be marginalized over
C	Another disjoint subset of the node set of a that is going to be conditioned on.
showmat	A logical value. TRUE (by default) to print the generated matrix.
plot	A logical value, FALSE (by default). TRUE to plot the generated graph.
plotfun	Function to plot the graph when plot == TRUE. Can be plotGraph (the default) or drawGraph.
...	Further arguments passed to plotfun.

Details

This function uses the functions AG and Max.

Value

A matrix that consists 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Richardson, T. S. and Spirtes, P. (2002). Ancestral graph Markov models. *Annals of Statistics*, 30(4), 962-1030.

Sadeghi, K. (2013). Stable mixed graphs. *Bernoulli* 19(5B), 2330–2358.

Sadeghi, K. and Lauritzen, S.L. (2014). Markov properties for loopless mixed graphs. *Bernoulli* 20(2), 676-696.

See Also

AG, Max, MRG, MSG

Examples

```
ex<-matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, ##The adjacency matrix of a DAG
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,
```

```
              0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,
              0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
              0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
              0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,
              0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,
              0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,
              1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
              0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,
              0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
              1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,
              0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0), 16, 16, byrow = TRUE)
M <- c(3,5,6,15,16)
C <- c(4,7)
MAG(ex, M, C, plot=TRUE)
##################################################
H <- matrix(c(0,100,1,0,100,0,100,0,0,100,0,100,0,1,100,0),4,4)
Max(H)
```

---

makeMG *Mixed Graphs*

Description

Defines a loopless mixed graph from the directed, undirected and undirected components.

Usage

```
makeMG(dg = NULL, ug = NULL, bg = NULL)
```

Arguments

dg	the adjacency matrix of a directed graph specifying the arrows of the mixed graph.
ug	the adjacency matrix of an undirected graph specifying the lines of the mixed graph.
bg	the adjacency matrix of an undirected graph specifying the bidirected edges of the mixed graph.

Details

A loopless mixed graph is a mixed graph with three types of edges: undirected, directed and bi-directed edges. Note that the three adjacency matrices must have labels and may be defined using the functions DG, DAG or UG. The adjacency matrices of the undirected graphs may be just symmetric Boolean matrices.

Value

a square matrix obtained by combining the three graph components into an adjacency matrix of a mixed graph. The matrix consists of 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Giovanni M. Marchetti, Mathias Drton

References

Richardson, T. S. and Spirtes, P. (2002). Ancestral Graph Markov Models. *Annals of Statistics*, 30(4), 962–1030.

See Also

UG, DAG

Examples

```
## Examples from Richardson and Spirtes (2002)
a1 <- makeMG(dg=DAG(a~b, b~d, d~c), bg=UG(~a*c))
isAG(a1)    # Not an AG. (a2) p.969
a2 <- makeMG(dg=DAG(b ~ a, d~c), bg=UG(~a*c+c*b+b*d))           # Fig. 3 (b1) p.969
isAG(a1)
a3 <- makeMG(ug = UG(~ a*c), dg=DAG(b ~ a, d~c), bg=UG(~ b*d)) # Fig. 3 (b2) p.969
a5 <- makeMG(bg=UG(~alpha*beta+gamma*delta), dg=DAG(alpha~gamma,
delta~beta))  # Fig. 6 p. 973
## Another Example
a4 <- makeMG(ug=UG(~y0*y1), dg=DAG(y4~y2, y2~y1), bg=UG(~y2*y3+y3*y4))
## A mixed graphs with double edges.
mg <- makeMG(dg = DG(Y ~ X, Z~W, W~Z, Q~X), ug = UG(~X*Q),
bg = UG(~ Y*X+X*Q+Q*W + Y*Z) )
## Chronic pain data: a regression graph
chronic.pain <- makeMG(dg = DAG(Y ~ Za, Za ~ Zb + A, Xa ~ Xb,
Xb ~ U+V, U ~ A + V, Zb ~ B, A ~ B), bg = UG(~Za*Xa + Zb*Xb))
```

`marg.param` *Link function of marginal log-linear parameterization*

Description

Provides the contrast and marginalization matrices for the marginal parametrization of a probability vector.

Usage

```
marg.param(lev, type)
```

Arguments

lev	Integer vector containing the number of levels of each variable.
type	A character vector with elements "l", "g", "c", or "r" indicating the type of logit. The meaning is as follows: "g" for global, "c" for continuation, "r" for reverse continuation and "l" for local.

Details

See Bartolucci, Colombi and Forcina (2007).

Value

C	Matrix of constrasts (the first sum(lev)-length(r) elements are referred to univariate logits)
M	Marginalization matrix with elements 0 and 1.
G	Corresponding design matrix for the corresponding log-linear model.

Note

Assumes that the vector of probabilities is in inv lex order. The interactions are returned in order of dimension, like e.g., 1, 2, 3, 12, 13, 23, 123.

Author(s)

Francesco Bartolucci, Antonio Forcina, Giovanni M. Marchetti

References

Bartolucci, F., Colombi, R. and Forcina, A. (2007). An extended class of marginal link functions for modelling contingency tables by equality and inequality constraints. Statist. Sinica 17, 691-711.

See Also

mat.mlogit

Examples

```
marg.param(c(3,3), c("l", "g"))
```

MarkEqMag *Markov equivalence of maximal ancestral graphs*

Description

MarkEqMag determines whether two MAGs are Markov equivalent.

Usage

```
MarkEqMag(amat, bmat)
```

Arguments

amat	An adjacency matrix of a MAG, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
bmat	The same as amat.

Details

The function checks whether the two graphs have the same skeleton and colliders with order.

Value

"Markov Equivalent" or "NOT Markov Equivalent".

Author(s)

Kayvan Sadeghi

References

Ali, R.A., Richardson, T.S. and Spirtes, P. (2009) Markov equivalence for ancestral graphs. *Annals of Statistics*, 37(5B),2808-2837.

See Also

MarkEqRcg, msep

Examples

```
H1<-matrix(  c(0,100,  0,  0,
         100,  0,100,  0,
               0,100,  0,100,
               0,  1,100,  0), 4, 4)
H2<-matrix(c(0,0,0,0,1,0,100,0,0,100,0,100,0,1,100,0),4,4)
H3<-matrix(c(0,0,0,0,1,0,0,0,0,1,0,100,0,1,100,0),4,4)
```

```
MarkEqMag(H1,H2)
MarkEqMag(H1,H3)
MarkEqMag(H2,H3)
```

MarkEqRcg *Markov equivalence for regression chain graphs.*

Description

MarkEqMag determines whether two RCGs (or subclasses of RCGs) are Markov equivalent.

Usage

```
MarkEqRcg(amat, bmat)
```

Arguments

amat	An adjacency matrix of an RCG or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
bmat	The same as amat.

Details

The function checks whether the two graphs have the same skeleton and unshielded colliders.

Value

"Markov Equivalent" or "NOT Markov Equivalent".

Author(s)

Kayvan Sadeghi

References

Wermuth, N. and Sadeghi, K. (2012). Sequences of regressions and their independences. Test 21:215–252.

See Also

MarkEqMag, msep

Examples

```
H1<-matrix(c(0,100,0,0,0,100,0,100,0,0,0,100,0,0,0,1,0,0,0,100,0,0,1,100,0),5,5)
H2<-matrix(c(0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,100,0,0,1,100,0),5,5)
H3<-matrix(c(0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1,0),5,5)
#MarkEqRcg(H1,H2)
#MarkEqRcg(H1,H3)
#MarkEqRcg(H2,H3)
```

`marks` *Mathematics marks*

Description

Examination marks of 88 students in five subjects.

Usage

```
data(marks)
```

Format

A data frame with 88 observations on the following 5 variables.

mechanics a numeric vector, mark in Mechanics

vectors a numeric vector, mark in Vectors

algebra a numeric vector, mark in Algebra

analysis a numeric vector, mark in Analysis

statistics a numeric vector, mark in Statistics

Details

Mechanics and Vectors were closed book examinations. Algebra, Analysis and Statistics were open book examinations.

Source

Mardia, K.V., Kent, J.T. and Bibby, (1979). *Multivariate analysis*. London: Academic Press.

References

Whittaker, J. (1990). *Graphical models in applied multivariate statistics*. Chichester: Wiley.

Examples

```
data(marks)
pairs(marks)
```

`mat.mlogit` *Multivariate logistic parametrization*

Description

Find matrices `C` and `M` of `e` binary multivariate logistic parameterization.

Usage

```
mat.mlogit(d, P = powerset(1:d))
```

Arguments

`d`	A positive integer, the number of binary responses.
`P`	A list of vectors of integers specifying margins. For instance `list(1, 2, c(1,2))`. Default: the power set of `1:d`.

Details

The power set is in the order of dimensions of the sets.

Value

`C`	A contrast matrix.
`L`	A marginalization matrix.

Author(s)

Giovanni M. Marchetti

References

Glonek, G. J. N. and McCullagh, P. (1995). Multivariate logistic models. Journal of the Royal Statistical Society, Ser. B 57, 533-546.

See Also

`binomial`, `marg.param`

Examples

```
mat.mlogit(2)
```

Max *Maximisation for graphs*

Description

Max generates a maximal graph that induces the same independence model from a non-maximal graph.

Usage

```
Max(amat)
```

Arguments

amat	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).

Details

Max looks for non-adjacent pais of nodes that are connected by primitive inducing paths, and connect such pairs by an appropriate edge.

Value

A matrix that consists 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Richardson, T.S. and Spirtes, P. (2002). Ancestral graph Markov models. *Annals of Statistics*, 30(4), 962-1030.

Sadeghi, K. and Lauritzen, S.L. (2014). Markov properties for loopless mixed graphs. *Bernoulli* 20(2), 676-696.

See Also

MAG, MRG, msep, MSG

Examples

```
H <- matrix(c(  0,100,  1,  0,
          100,  0,100,  0,
            0,100,  0,100,
            0,  1,100,  0), 4, 4)
Max(H)
```

MRG *Maximal ribbonless graph*

Description

MRG generates and plots maximal ribbonless graphs (a modification of MC graph to use m-separation) after marginalisation and conditioning.

Usage

```
MRG(amat,M=c(),C=c(),showmat=TRUE,plot=FALSE, plotfun = plotGraph, ...)
```

Arguments

amat	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
M	A subset of the node set of a that is going to be marginalized over
C	Another disjoint subset of the node set of a that is going to be conditioned on.
showmat	A logical value. TRUE (by default) to print the generated matrix.
plot	A logical value, FALSE (by default). TRUE to plot the generated graph.
plotfun	Function to plot the graph when plot == TRUE. Can be plotGraph (the default) or drawGraph.
...	Further arguments passed to plotfun.

Details

This function uses the functions RG and Max.

Value

A matrix that consists 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Koster, J.T.A. (2002). Marginalizing and conditioning in graphical models. *Bernoulli*, 8(6), 817-840.

Richardson, T.S. and Spirtes, P. (2002). Ancestral graph Markov models. *Annals of Statistics*, 30(4), 962-1030.

Sadeghi, K. (2013). Stable mixed graphs. *Bernoulli* 19(5B), 2330–2358.

Sadeghi, K. and Lauritzen, S.L. (2014). Markov properties for loopless mixed graphs. *Bernoulli* 20(2), 676-696.

See Also

MAG, Max, MSG, RG

Examples

```
ex <- matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, ##The adjacency matrix of a DAG
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,
               0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,
               0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,0,
               1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,
               0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0),16,16, byrow = TRUE)
M <- c(3,5,6,15,16)
C <- c(4,7)
MRG(ex, M, C, plot = TRUE)
##################################################
H <- matrix(c( 0, 100,   1,   0,
             100,   0, 100,   0,
               0, 100,   0, 100,
               0,   1, 100,   0), 4,4)
Max(H)
```

---

msep *The m-separation criterion*

Description

msep determines whether two set of nodes are m-separated by a third set of nodes.

Usage

```
msep(a, alpha, beta, C = c())
```

Arguments

a	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
alpha	A subset of the node set of a
beta	Another disjoint subset of the node set of a
C	A third disjoint subset of the node set of a

Value

A logical value. TRUE if alpha and beta are m-separated given C. FALSE otherwise.

Author(s)

Kayvan Sadeghi

References

Richardson, T.S. and Spirtes, P. (2002) Ancestral graph Markov models. *Annals of Statistics*, 30(4), 962-1030.

Sadeghi, K. and Lauritzen, S.L. (2014). Markov properties for loopless mixed graphs. *Bernoulli* 20(2), 676-696.

See Also

dSep, MarkEqMag

Examples

```
H <-matrix(c(0,0,0,0,
         1,0,0,1,
         0,1,0,0,
         0,0,0,0),4,4)
msep(H,1,4, 2)
msep(H,1,4, c())
```

MSG *Maximal summary graph*

Description

MAG generates and plots maximal summary graphs after marginalization and conditioning.

Usage

```
MSG(amat,M=c(),C=c(),showmat=TRUE,plot=FALSE, plotfun = plotGraph, ...)
```

Arguments

amat	An adjacency matrix of a MAG, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
M	A subset of the node set of a that is going to be marginalized over
C	Another disjoint subset of the node set of a that is going to be conditioned on.
showmat	A logical value. TRUE (by default) to print the generated matrix.
plot	A logical value, FALSE (by default). TRUE to plot the generated graph.
plotfun	Function to plot the graph when plot == TRUE. Can be plotGraph (the default) or drawGraph.
...	Further arguments passed to plotfun.

Details

This function uses the functions SG and Max.

Value

A matrix that consists 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Richardson, T.S. and Spirtes, P. (2002). Ancestral graph Markov models. *Annals of Statistics*, 30(4), 962-1030.

Sadeghi, K. (2013). Stable mixed graphs. *Bernoulli* 19(5B), 2330–2358.

Sadeghi, K. and Lauritzen, S.L. (2014). Markov properties for loopless mixed graphs. *Bernoulli* 20(2), 676-696.

Wermuth, N. (2011). Probability distributions with summary graph structure. *Bernoulli*, 17(3), 845-879.

See Also

MAG, Max, MRG, SG

Examples

```
ex<-matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, ##The adjacency matrix of a DAG
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,
             0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,
             0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,0,
             1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,
             0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
             1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,
             0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0), 16, 16, byrow=TRUE)
M <- c(3,5,6,15,16)
C <- c(4,7)
MSG(ex,M,C,plot=TRUE)
##################################################
H<-matrix(c(0,100,1,0,100,0,100,0,0,100,0,100,0,1,100,0),4,4)
Max(H)
```

`null` *Null space of a matrix*

Description

Given a matrix `M` find a matrix `N` such that $N^T M$ is zero.

Usage

```
null(M)
```

Arguments

M	A matrix.

Value

The matrix N with the basis for the null space, or an empty vector if the matrix M is square and of maximal rank.

See Also

Null, ~~~

Examples

```
null(c(1,1,1))
```

---

parcor *Partial correlations*

Description

Finds the matrix of the partial correlations between pairs of variables given the rest.

Usage

```
parcor(S)
```

Arguments

S	a symmetric positive definite matrix, representing a covariance matrix.

Details

The algorithm computes $-\sigma^{rs}/(\sigma^{rr}\sigma^{ss})^{1/2}$ where the σ^{rs} are concentrations, i.e. elements of the inverse covariance matrix.

Value

A symmetric matrix with ones along the diagonal and in position (r, s) the partial correlation between variables r and s given all the remaining variables.

Author(s)

Giovanni M. Marchetti

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

See Also

var, cor, correlations

Examples

```
### Partial correlations for the mathematics marks data
data(marks)
S <- var(marks)
parcor(S)
```

---

pcor *Partial correlation*

Description

Computes the partial correlation between two variables given a set of other variables.

Usage

```
pcor(u, S)
```

Arguments

u	a vector of integers of length > 1. The first two integers are the indices of variables the correlation of which must be computed. The rest of the vector is the conditioning set.
S	a symmetric positive definite matrix, a sample covariance matrix.

Value

a scalar, the partial correlation matrix between variables `u[1]` and `u[2]` given `u[-c(1,2)]`.

Author(s)

Giovanni M. Marchetti

See Also

cor, parcor, correlations

Examples

```
data(marks)
## The correlation between vectors and algebra given analysis and statistics
 pcor(c("vectors", "algebra", "analysis", "statistics"), var(marks))
## The same
pcor(c(2,3,4,5), var(marks))
## The correlation between vectors and algebra given statistics
 pcor(c("vectors", "algebra", "statistics"), var(marks))
## The marginal correlation between analysis and statistics
pcor(c("analysis","statistics"), var(marks))
```

`pcor.test` *Test for zero partial association*

Description

Test for conditional independence between two variables, given the other ones, assuming a multivariate normal distribution.

Usage

```
pcor.test(r, q, n)
```

Arguments

`r`	a partial correlation coefficient, computed by `pcor`.
`q`	the number of variables in the conditioning set.
`n`	integer > 0, the sample size.

Value

`tval`	The Student's t-test statistic.
`df`	The degrees of freedom
`pvalue`	The P-value, assuming a two-sided alternative.

Author(s)

Giovanni M. Marchetti

See Also

`pcor`, `shipley.test`

Examples

```
## Are 2,3 independent given 1?
data(marks)
pcor.test(pcor(c(2,3,1), var(marks)), 1, n=88)
```

plotGraph *Plot of a mixed graph*

Description

Plots a mixed graph from an adjacency matrix, a graphNEL object, an igraph object, or a descriptive vector.

Usage

```
plotGraph(a, dashed = FALSE, tcltk = TRUE, layout = layout.auto,
directed = FALSE, noframe = FALSE, nodesize = 15, vld = 0, vc = "gray",
vfc = "black", colbid = "FireBrick3", coloth = "black", cex = 1.5, ...)
```

Arguments

a	An adjacency matrix: a matrix that consists of 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers can be added to generate multiple edges of different types. The matrix must be symmetric w.r.t full lines and bi-directed arrows. Or a graph that can be a graphNEL or an igraph object.Or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type,node1label,node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
dashed	A logical value. If TRUE the bi-directed edges are plotted as undirected dashed edges.
tcltk	A logical value. If TRUE the function opens a tcltk device to plot the graphs, allowing the interactive manimulation of the graph. If FALSEthe function opens a standard device without interaction.
layout	The name of a function used to compute the (initial) layout of the graph. The default is layout.auto. This can be further adjusted if tcltk is TRUE.
directed	A logical value. If FALSE a symmetric adjacency matrix with entries 1 is interpreted as an undirected graph. If TRUE it is interpreted as a directed graph with double arrows. If a is not an adjacency matrix, it is ignored.
noframe	A logical value. If TRUE, then the nodes are not circled.
nodesize	An integer denoting the size of the nodes (default 15). It can be increased to accommodate larger labels.
vld	An integer defining the distance between a vertex and its label. Defaults to 0.
vc	Vertex color. Default is "gray".
vfc	Vertex frame color. Default is "black".
colbid	Color of the bi-directed edges. Default is "FireBrick3".
coloth	Color of all the other edges. Default is "black".
cex	An integer (defaults to 1) to adjust the scaling of the font of the labels.
...	Further arguments to be passed to plot or tkplot.

Details

plotGraph uses plot.igraph and tkplot in **igraph** package.

Value

Plot of the associated graph and returns invisibly a list with two slots: tkp.id, graph, the input graph as an igraph object. The id can be used to get the layout of the adjusted graph. The bi-directed edges are plotted in red.

Author(s)

Kayvan Sadeghi, Giovanni M. Marchetti

See Also

grMAT, tkplot, drawGraph, plot.igraph

Examples

```
exvec<-c("b",1,2,"b",1,14,"a",9,8,"1",9,11,
         "a",10,8,"a",11,2,"a",11,9,"a",11,10,
         "a",12,1,"b",12,14,"a",13,10,"a",13,12)
plotGraph(exvec)
##########################################
amat<-matrix(c(0,11,0,0,10,0,100,0,0,100,0,1,0,0,1,0),4,4)
plotGraph(amat)
plotGraph(makeMG(bg = UG(~a*b*c+ c*d), dg = DAG(a ~ x + z, b ~ z )))
plotGraph(makeMG(bg = UG(~a*b*c+ c*d), dg = DAG(a ~ x + z, b ~ z )), dashed = TRUE)
# A graph with double and triple edges
G <-
structure(c(0, 101, 0, 0, 100, 0, 100, 100, 0, 100, 0, 100, 0,
111, 100, 0), .Dim = c(4L, 4L), .Dimnames = list(c("X", "Z",
"Y", "W"), c("X", "Z", "Y", "W")))
plotGraph(G)
# A regression chain graph with longer labels
 plotGraph(makeMG(bg = UG(~Love*Constraints+ Constraints*Reversal+ Abuse*Distress),
   dg = DAG(Love ~ Abuse + Distress, Constraints ~ Distress, Reversal ~ Distress,
   Abuse ~ Fstatus, Distress ~ Fstatus),
   ug = UG(~Fstatus*Schooling+ Schooling*Age)),
   dashed = TRUE, noframe = TRUE)
# A graph with 4 edges between two nodes.
G4 = matrix(0, 2, 2); G4[1,2] = 111; G4[2,1] = 111
plotGraph(G4)
```

powerset *Power set*

Description

Finds the list of all subsets of a set.

Usage

```
powerset(set, sort = TRUE, nonempty = TRUE)
```

Arguments

set	A numeric or character vector.
sort	Logical value. If TRUE the subsets are sorted according to dimension. Default is TRUE.
nonempty	Logical value. If TRUE the empty set is omitted. Default is TRUE.

Details

If sort == FALSE the sets are in inverse lexicographical order.

Value

A list of all subsets of set.

Author(s)

Giovanni M. Marchetti

Examples

```
powerset(c("A", "B", "C"), nonempty = FALSE)
powerset(1:3, sort = FALSE, nonempty = TRUE)
```

rcorr *Random correlation matrix*

Description

Generates a random correlation matrix with the method of Marsaglia and Olkin (1984).

Usage

```
rcorr(d)
```

Arguments

d	an integer > 0, the order of the correlation matrix.

Details

The algorithm uses rsphere to generate d vectors on a sphere in d-space. If Z is a matrix with such vectors as rows, then the random correlation matrix is ZZ'.

Value

a correlation matrix of order d.

Author(s)

Giovanni M. Marchetti

References

Marshall, G.& Olkin, I. (1984).Generating correlation matrices. *SIAM J. Sci. Stat. Comput.*, 5, 2, 470–475.

See Also

rsphere

Examples

```
## A random correlation matrix of order 3
rcorr(3)
## A random correlation matrix of order 5
rcorr(5)
```

RepMarBG *Representational Markov equivalence to bidirected graphs.*

Description

RepMarBG determines whether a given maximal ancestral graph can be Markov equivalent to a bidirected graph, and if that is the case, it finds a bidirected graph that is Markov equivalent to the given graph.

Usage

```
RepMarBG(amat)
```

Arguments

amat	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).

Details

RepMarBG looks for presence of an unshielded non-collider V-configuration in graph.

Value

A list with two components: verify and amat. verify is a logical value, TRUE if there is a representational Markov equivalence and FALSE otherwise. amat is either NA if verify == FALSE or the adjacency matrix of the generated graph, if verify == TRUE. In this case it consists of 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Sadeghi, K. (2011). Markov equivalences for subclasses of loopless mixed graphs. *Submitted*, 2011.

See Also

MarkEqMag, MarkEqRcg, RepMarDAG, RepMarUG

Examples

```
H<-matrix(c(0,10,0,0,10,0,0,0,0,1,0,100,0,0,100,0),4,4)
RepMarBG(H)
```

RepMarDAG *Representational Markov equivalence to directed acyclic graphs.*

Description

RepMarDAG determines whether a given maximal ancestral graph can be Markov equivalent to a directed acyclic graph, and if that is the case, it finds a directed acyclic graph that is Markov equivalent to the given graph.

Usage

```
RepMarDAG(amat)
```

Arguments

amat	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).

Details

RepMarDAG first looks whether the subgraph induced by full lines is chordal and whether there is a minimal collider path or cycle of length 4 in graph.

Value

A list with two components: verify and amat. verify is a logical value, TRUE if there is a representational Markov equivalence and FALSE otherwise. amat is either NA if verify == FALSE or the adjacency matrix of the generated graph, if verify == TRUE. In this case it consists of 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Sadeghi, K. (2011). Markov equivalences for subclasses of loopless mixed graphs. *Submitted*, 2011.

See Also

MarkEqMag, MarkEqRcg, RepMarBG, RepMarUG

Examples

```
H<-matrix(c(0,10,0,0,10,0,0,0,0,1,0,100,0,0,100,0),4,4)
RepMarBG(H)
```

RepMarUG *Representational Markov equivalence to undirected graphs.*

Description

RepMarUG determines whether a given maximal ancestral graph can be Markov equivalent to an undirected graph, and if that is the case, it finds an undirected graph that is Markov equivalent to the given graph.

Usage

```
RepMarUG(amat)
```

Arguments

amat	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).

Details

RepMarBG looks for presence of an unshielded collider V-configuration in graph.

Value

A list with two components: verify and amat. verify is a logical value, TRUE if there is a representational Markov equivalence and FALSE otherwise. amat is either NA if verify == FALSE or the adjacency matrix of the generated graph, if verify == TRUE. In this case it consists of 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Sadeghi, K. (2011). Markov equivalences for subclasses of loopless mixed graphs. *Submitted*, 2011.

See Also

MarkEqMag, MarkEqRcg, RepMarBG, RepMarDAG

Examples

```
H<-matrix(c(0,10,0,0,10,0,0,0,0,1,0,100,0,0,100,0),4,4)
RepMarUG(H)
```

RG *Ribbonless graph*

Description

RG generates and plots ribbonless graphs (a modification of MC graph to use m-separation) after marginalization and conditioning.

Usage

```
RG(amat,M=c(),C=c(),showmat=TRUE,plot=FALSE, plotfun = plotGraph, ...)
```

Arguments

amat	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
M	A subset of the node set of a that is going to be marginalized over
C	Another disjoint subset of the node set of a that is going to be conditioned on.
showmat	A logical value. TRUE (by default) to print the generated matrix.
plot	A logical value, FALSE (by default). TRUE to plot the generated graph.
plotfun	Function to plot the graph when plot == TRUE. Can be plotGraph (the default) or drawGraph.
...	Further arguments passed to plotfun.

Value

A matrix that consists 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Koster, J.T.A. (2002). Marginalizing and conditioning in graphical models. *Bernoulli*, 8(6), 817-840.

Sadeghi, K. (2013). Stable mixed graphs. *Bernoulli* 19(5B), 2330–2358.

See Also

AG,, MRG, SG

Examples

```
ex <- matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, ##The adjacency matrix of a DAG
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,
               0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,
               0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,0,
               1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,
               0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0),16,16, byrow = TRUE)

M<-c(3,5,6,15,16)
C<-c(4,7)
RG(ex,M,C,plot=TRUE)
```

---

rnormDag *Random sample from a decomposable Gaussian model*

Description

Generates a sample from a mean centered multivariate normal distribution whose covariance matrix has a given triangular decomposition.

Usage

```
rnormDag(n, A, Delta)
```

Arguments

n	an integer > 0, the sample size.
A	a square, upper triangular matrix with ones along the diagonal. It defines, together with Delta, the concentration matrix (and also the covariance matrix) of the multivariate normal. The order of A is the number of components of the normal.
Delta	a numeric vector of length equal to the number of columns of A.

Details

The value in position (i, j) of `A` (with $i < j$) is a regression coefficient (with sign changed) in the regression of variable i on variables $i + 1, \ldots, d$.

The value in position i of `Delta` is the residual variance in the above regression.

Value

a matrix with `n` rows and `nrow(A)` columns, a sample from a multivariate normal distribution with mean zero and covariance matrix `S = solve(A) %*% diag(Delta) %*% t(solve(A))`.

Author(s)

Giovanni M. Marchetti

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

See Also

`triDec`, `fitDag`

Examples

```
## Generate a sample of 100 observation from a multivariate normal
## The matrix of the path coefficients
A <- matrix(
c(1, -2, -3,  0, 0,  0,  0,
  0,  1,  0, -4, 0,  0,  0,
  0,  0,  1,  2, 0,  0,  0,
  0,  0,  0,  1, 1, -5,  0,
  0,  0,  0,  0, 1,  0,  3,
  0,  0,  0,  0, 0,  1, -4,
  0,  0,  0,  0, 0,  0,  1), 7, 7, byrow=TRUE)
D <- rep(1, 7)
X <- rnormDag(100, A, D)

## The true covariance matrix
solve(A) %*% diag(D) %*% t(solve(A))

## Triangular decomposition of the sample covariance matrix
triDec(cov(X))$A
```

rsphere *Random vectors on a sphere*

Description

Generates a sample of points uniformly distributed on the surface of a sphere in d-space.

Usage

```
rsphere(n, d)
```

Arguments

n	an integer, the sample size.
d	an integer, the dimension of the space. For example, a circle is defined in 2D-space, a sphere in 3D-space.

Details

The algorithm is based on normalizing to length 1 each d-vector of a sample from a multivariate normal $N(0, I)$.

Value

a matrix of n rows and d columns.

Author(s)

Giovanni M. Marchetti

See Also

rnorm, rcorr

Examples

```
## 100 points on circle
z <- rsphere(100,2)
plot(z)

## 100 points on a sphere
z <- rsphere(100, 3)
pairs(z)
```

SG *summary graph*

Description

SG generates and plots summary graphs after marginalization and conditioning.

Usage

```
SG(amat,M=c(),C=c(),showmat=TRUE,plot=FALSE, plotfun = plotGraph, ...)
```

Arguments

amat	An adjacency matrix, or a graph that can be a graphNEL or an igraph object or a vector of length $3e$, where e is the number of edges of the graph, that is a sequence of triples (type, node1label, node2label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines).
M	A subset of the node set of a that is going to be marginalised over
C	Another disjoint subset of the node set of a that is going to be conditioned on.
showmat	A logical value. TRUE (by default) to print the generated matrix.
plot	A logical value, FALSE (by default). TRUE to plot the generated graph.
plotfun	Function to plot the graph when plot == TRUE. Can be plotGraph (the default) or drawGraph.
...	Further arguments passed to plotfun.

Value

A matrix that consists 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Author(s)

Kayvan Sadeghi

References

Sadeghi, K. (2013). Stable mixed graphs. *Bernoulli* 19(5B), 2330–2358.

Wermuth, N. (2011). Probability distributions with summary graph structure. *Bernoulli*, 17(3),845-879.

See Also

AG, MSG, RG

Examples

```
ex <- matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, ##The adjacency matrix of a DAG
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,
               0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,
               0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,0,
               1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,
               0,0,0,0,0,0,0,0,0,0,1,0,1,0,0),16,16, byrow = TRUE)
M <- c(3,5,6,15,16)
C <- c(4,7)
SG(ex, M, C, plot = TRUE)
SG(ex, M, C, plot = TRUE, plotfun = drawGraph, adjust = FALSE)
```

`shipley.test` *Test of all independencies implied by a given DAG*

Description

Computes a simultaneous test of all independence relationships implied by a given Gaussian model defined according to a directed acyclic graph, based on the sample covariance matrix.

Usage

```
shipley.test(amat, S, n)
```

Arguments

`amat`	a square Boolean matrix, of the same dimension as `S`, representing the adjacency matrix of a DAG.
`S`	a symmetric positive definite matrix, the sample covariance matrix.
`n`	a positive integer, the sample size.

Details

The test statistic is $C = -2 \sum \ln p_j$ where p_j are the p-values of tests of conditional independence in the basis set computed by `basiSet(A)`. The p-values are independent uniform variables on $(0, 1)$ and the statistic has exactly a chi square distribution on $2k$ degrees of freedom where k is the number of elements of the basis set. Shipley (2002) calls this test Fisher's C test.

Value

ctest	Test statistic C.
df	Degrees of freedom.
pvalue	The P-value of the test, assuming a two-sided alternative.

Author(s)

Giovanni M. Marchetti

References

Shipley, B. (2000). A new inferential test for path models based on directed acyclic graphs. *Structural Equation Modeling*, 7(2), 206–218.

See Also

basiSet, pcor.test

Examples

```
## A decomposable model for the mathematics marks data
data(marks)
dag <- DAG(mechanics ~ vectors+algebra, vectors ~ algebra,
statistics ~ algebra+analysis, analysis ~ algebra)
shipley.test(dag, cov(marks), n=88)
```

Simple Graph Operations

Simple graph operations

Description

Finds the boundary, children, parents of a subset of nodes of a graph.

Usage

```
bd(nn, amat)
ch(nn, amat)
pa(nn, amat)
```

Arguments

nn	a vector of nodes. It may either a numeric vector, or a character vector. If it is character vector must be a subset of the rownames of the edge matrix.
amat	a square matrix with dimnames specifying the adjacency matrix of the graph

Details

For definitions of the operators see Lauritzen (1996).

Value

The operators return a character vector specifying the boundary or the children or the parents of nodes nn in the graph. This is a numeric or a character vector depending on the mode of nn.

Author(s)

Giovanni M. Marchetti

References

Lauritzen, S. (1996). *Graphical models*. Oxford: Clarendon Press.

See Also

UG, DAG

Examples

```
## find boundary of a subset of nodes of a DAG
G <- DAG(y ~ x+b+a, b~a, x~a)
bd("b", G)
bd(c("b", "x"), G)
bd("x", G)
bd(c("x","b"), G)
## find boundary of a subset of nodes of an UG
G <- UG(~ y*x*z + z*h*v)
bd("z", G)
bd(c("y", "x"), G)
bd("v", G)
bd(c("x","v"), G)
## children of a subset of nodes of a DAG
G <- DAG(y ~ x+b+a, b~a, x~a)
ch("b", G)
ch(c("b", "x"), G)
ch("x", G)
ch(c("a","x"), G)
## parents of a subset of nodes of a DAG
pa("b", G)
pa(c("b", "x"), G)
pa("x", G)
pa(c("x","b"), G)
```

stress *Stress*

Description

Stress data

Usage

```
data(stress)
```

Format

A 4×4 covariance matrix for the following variables.

Y

V

X

U

Details

See Cox and Wermuth (1996).

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

Slangen K., Kleemann P.P and Krohne H.W. (1993). Coping with surgical stress. In: Krohne H. W. (ed.). *Attention and avoidance: Strategies in coping with aversiveness*. New York, Heidelberg: Springer, 321-346.

Examples

```
data(stress)
G = UG(~ Y*X + X*V + V*U + U*Y)
fitConGraph(G, stress, 100)
```

surdata *A simulated data set*

Description

Simulated data following a seemingly unrelated regression model.

Usage

```
data(surdata)
```

Format

A data frame with 600 observations on the following 4 variables.

`A` a numeric response vector

`B` a numeric response vector

`X` a numeric vector

`Z` a numeric vector with codes `1` and `-1` for a binary variables.

Examples

```
data(surdata)
pairs(surdata)
```

swp *Sweep operator*

Description

Sweeps a covariance matrix with respect to a subset of indices.

Usage

```
swp(V, b)
```

Arguments

`V`	a symmetric positive definite matrix, the covariance matrix.
`b`	a subset of indices of the columns of `V`.

Details

The sweep operator has been introduced by Beaton (1964) as a tool for inverting symmetric matrices (see Dempster, 1969).

Value

a square matrix U of the same order as V. If a is the complement of b, then U[a,b] is the matrix of regression coefficients of a given b and U[a,a] is the corresponding covariance matrix of the residuals.

If b is empty the function returns V.

If b is the vector 1:nrow(V) (or its permutation) then the function returns the opposite of the inverse of V.

Author(s)

Giovanni M. Marchetti

References

Beaton, A.E. (1964). *The use of special matrix operators in statistical calculus*. Ed.D. thesis, Harvard University. Reprinted as Educational Testing Service Research Bulletin 64-51. Princeton.

Dempster, A.P. (1969). *Elements of continuous multivariate analysis*. Reading: Addison-Wesley.

See Also

fitDag

Examples

```
## A very simple example
V <- matrix(c(10, 1, 1, 2), 2, 2)
swp(V, 2)
```

---

topSort *Topological sort*

Description

topOrder returns the topological order of a directed acyclic graph (parents, before children). topSort permutates the adjacency matrix according to the topological order.

Usage

```
topSort(amat)
topOrder(amat)
```

Arguments

amat	a square Boolean matrix with dimnames, representing the adjacency matrix of a directed acyclic graph.

Details

The topological order needs not to be unique. After the permutation the adjacency matrix of the graph is upper triangular. The function is a translation of the Matlab function `topological_sort` in Toolbox **BNT** written by Kevin P. Murphy.

Value

`topOrder(amat)` returns a vector of integers representing the permutation of the nodes. `topSort(amat)` returns the adjacency matrix with rows and columns permutated.

Note

The order of the nodes defined by `DAG` is that of their first appearance in the model formulae (from left to right).

Author(s)

Kevin P. Murphy, Giovanni M. Marchetti

References

Aho, A.V., Hopcrtoft, J.E. & Ullman, J.D. (1983). *Data structures and algorithms.* Reading: Addison-Wesley.

Lauritzen, S. (1996). *Graphical models*. Oxford: Clarendon Press.

See Also

`DAG`, `isAcyclic`

Examples

```
## A simple example
dag <- DAG(a ~ b, c ~ a + b, d ~ c + b)
dag
topOrder(dag)
topSort(dag)
```

`transClos` *Transitive closure of a graph*

Description

Computes the transitive closure of a graph (undirected or directed acyclic).

Usage

```
transClos(amat)
```

Arguments

amat	a Boolean matrix with dimnames representing the adjacency matrix of a graph.

Details

The transitive closure of a directed graph with adjacency matrix A is a graph with adjacency matrix A^* such that $A^*_{i,j} = 1$ if there is a directed path from i to j. The transitive closure of an undirected graph is defined similarly (by substituting path to directed path).

Value

A	The adjacency matrix of the transitive closure.

Author(s)

Giovanni M. Marchetti

See Also

DAG, UG

Examples

```
## Closure of a DAG
d <- DAG(y ~ x, x ~ z)
transClos(d)

## Closure of an UG
g <- UG(~ x*y*z+z*u+u*v)
transClos(g)
```

---

triDec *Triangular decomposition of a covariance matrix*

Description

Decomposes a symmetric positive definite matrix with a variant of the Cholesky decomposition.

Usage

```
triDec(Sigma)
```

Arguments

Sigma	a symmetric positive definite matrix.

Details

Any symmetric positive definite matrix Σ can be decomposed as $\Sigma = B\Delta B^T$ where B is upper triangular with ones along the main diagonal and Δ is diagonal. If Σ is a covariance matrix, the concentration matrix is $\Sigma^{-1} = A^T\Delta^{-1}A$ where $A = B^{-1}$ is the matrix of the regression coefficients (with the sign changed) of a system of linear recursive regression equations with independent residuals. In the equations each variable i is regressed on the variables $i+1, \ldots, d$. The elements on the diagonal of Δ are the partial variances.

Value

`A`	a square upper triangular matrix of the same order as `Sigma` with ones on the diagonal.
`B`	the inverse of `A`, another triangular matrix with unit diagonal.
`Delta`	a vector containing the diagonal values of Δ.

Author(s)

Giovanni M. Marchetti

References

Cox, D. R. & Wermuth, N. (1996). *Multivariate dependencies*. London: Chapman & Hall.

See Also

`chol`

Examples

```
## Triangular decomposition of a covariance matrix
B <- matrix(c(1,  -2, 0, 1,
              0,   1, 0, 1,
              0,   0, 1, 0,
              0,   0, 0, 1), 4, 4, byrow=TRUE)
B
D <- diag(c(3, 1, 2, 1))
S <- B %*% D %*% t(B)
triDec(S)
solve(B)
```

UG *Defining an undirected graph (UG)*

Description

A simple way to define an undirected graph by means of a single model formula.

Usage

```
UG(f)
```

Arguments

f	a single model formula without response

Details

The undirected graph $G = (V, E)$ is defined by a set of nodes V and a set of pairs E. The set of pairs is defined by the set of interactions in the formula. Interactions define complete subgraphs (not necessarily maximal) of the UG. The best way is to specify interactions that match the cliques of the undirected graph. This is the standard way to define graphical models for contingency tables. Remember that some hierarchical models are not graphical, but they imply the same graph.

The function returns the edge matrix of the graph, i.e. a square Boolean matrix of order equal to the number of nodes of the graph and a one in position (i, j) if there is an arrow from j to i and zero otherwise. By default this matrix has ones along the main diagonal. For UGs this matrix is symmetric. The dimnames of the edge matrix are the nodes of the UG.

Value

a Boolean matrix with dimnames, the adjacency matrix of the undirected graph.

Author(s)

Giovanni M. Marchetti

References

Lauritzen, S. (1996). *Graphical models*. Oxford: Clarendon Press.

See Also

fitConGraph, fitCovGraph, DAG

Examples

```
## X independent of Y given Z
UG(~ X*Z + Y*Z)

# The saturated model
UG(~ X*Y*Z)

## The model without three-way interactions has the same graph
UG(~ X*Y + Y*Z + Z*X)
UG(~ (X + Y + Z)^2)

## Butterfly model defined from the cliques
UG(~ mec*vec*alg + alg*ana*sta)
```

```
## Some isolated nodes
UG(~x*y*z + a + b)
```

---

unmakeMG *Loopless mixed graphs components*

Description

Splits the adjacency matrix of a loopless mixed graph into three components: directed, undirected and bi-directed.

Usage

```
unmakeMG(amat)
```

Arguments

amat	a square matrix, with dimnames, representing a loopless mixed graph. The matrix consists of 4 different integers as an ij-element: 0 for a missing edge between i and j, 1 for an arrow from i to j, 10 for a full line between i and j, and 100 for a bi-directed arrow between i and j. These numbers are added to be associated with multiple edges of different types. The matrix is symmetric w.r.t full lines and bi-directed arrows.

Details

The matrices ug, and bg are just symmetric Boolean matrices.

Value

It is the inverse of makeAG. It returns the following components.

dg	the adjacency matrix of the directed edges.
ug	the adjacency matrix of the undirected edges.
bg	the adjacency matrix of the bi-directed edges.

Author(s)

Mathias Drton, Giovanni M. Marchetti

See Also

makeMG

Examples

```
ag <- makeMG(ug=UG(~y0*y1), dg=DAG(y4~y2, y2~y1), bg=UG(~y2*y3+y3*y4))
isAG(ag)
unmakeMG(ag)
```

Utility Functions *Utility functions*

Description

Functions used internally.

Author(s)

Kayvan Sadeghi, Giovanni M. Marchetti

See Also

unique,setdiff, is.element

Index