# Package 'rCausalMGM'

May 9, 2026

**Type** Package

**Title** Scalable Causal Discovery and Model Selection on Mixed Datasets with 'rCausalMGM'

**Version** 1.0.1

**Date** 2026-03-13

**Author** Tyler C Lovelace [aut],
Max Dudek [aut],
Jack Fiore [aut],
Panayiotis V Benos [aut, cre]

**Maintainer** Panayiotis V Benos <pbenos@ufl.edu>

**Description** Scalable methods for learning causal graphical models from mixed data, including continuous, discrete, and censored variables. The package implements CausalMGM, which combines a convex, score-based approach for learning an initial moralized graph with a producer-consumer scheme that enables efficient parallel conditional independence testing in constraint-based causal discovery algorithms. The implementation supports high-dimensional datasets and provides individual access to core components of the workflow, including MGM and the PC-Stable and FCI-Stable causal discovery algorithms. To support practical applications, the package includes multiple model selection strategies, including information criteria based on likelihood and model complexity, cross-validation for out-of-sample likelihood estimation, and stability-based approaches that assess graph robustness across subsamples.

**License** GPL-3

**Imports** Rcpp (>= 1.0.3), survival

**LinkingTo** BH, Rcpp, RcppArmadillo, RcppThread

**Suggests** Rgraphviz, graph

**RoxygenNote** 7.3.2

**Encoding** UTF-8

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2026-03-13 19:00:02 UTC

# Contents

---

`adjMat2Graph` *Convert an adjacency matrix into a graph*

---

## Description

Convert an adjacency matrix into a graph

## Usage

```
adjMat2Graph(adj, nodes, directed = FALSE)
```

## Arguments

| | |
|---|---|
| `adj` | The adjacency matrix, p x p, with non-zero values indicating the presence of an adjacency. |
| `nodes` | The names of the nodes, length p. |
| `directed` | TRUE if the graph should be directed. This default is FALSE. |

## Value

A graph object representing the adjacency matrix.

## Examples

```
mat <- matrix(sample(c(0,1), 16, replace=TRUE), nrow=4)
mat <- mat + t(mat)
nodes <- c("X1", "X2", "X3", "X4")
g <- adjMat2Graph(mat, nodes)
```

---

`allMetrics` *Combined graph recovery metrics*

---

## Description

Calculate the SHD, precision, recall, F1, and Matthew's Correlation Coefficient (MCC) for the adjacencies and orientations of an estimated graph compared to the ground truth. This is the concatenated output of the SHD, adjacency PR metrics, and the orientation PR metrics.

## Usage

```
allMetrics(estimate, groundTruth, groundTruthDAG = NULL)
```

### Arguments

| | |
|---|---|
| estimate | An estimated graph object |
| groundTruth | A ground truth graph object of the same type as the estimated graph object |
| groundTruthDAG | A ground truth graph object containing the true causal DAG. Only necessary for calculating the or precision, recall, F1, and MCC for partial ancestral graphs (PAGs) |

### Value

The orientation precision, recall, F1, and MCC, between the two graph objects

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- pcStable(sim$data)
allMetrics(g, cpdag(sim$graph))
```

---

## bootstrap *Runs bootstrapping for a causal graph on the dataset.*

---

### Description

Runs bootstrapping for a causal graph on the dataset. This function can be used to estimate the stability of edge adjacencies and orientations in the causal graph. It returns an ensemble graph which consists of the most common edges accross bootstrap samples. The ensemble graph is constructed based on edge-wise probabilities, so it is not guaranteed to be a valid CPDAG or PAG. The ensemble graph's stabilites entry contains information about the frequency of each possible orientation for each edge that appears at least once across bootstrap samples.

### Usage

```
bootstrap(
  data,
  graph,
  knowledge = NULL,
  numBoots = 20L,
  threads = -1L,
  replace = FALSE,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| graph | A graph object containing the graph to estimate the stability of through bootstrapping. |
| knowledge | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |
| numBoots | The number of bootstrap samples to run. The default is 20. |
| threads | An integer value denoting the number of threads to use for parallelization. The default value is -1, which will all available CPUs. |
| replace | A logical value indicating whether to use sampling with replacement or to draw subsamples of size floor(0.632 * N). The default value is FALSE. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graph object representing an ensemble graph learned from bootstrapped samples. For each adjacency observed across the bootstrap graphs, if absence is not the most frequent outcome, the edge orientation with the highest frequency is included in the ensemble graph. The object also contains a 'stabilities' data frame that records the frequencies of all possible edge orientations for each observed adjacency. The ensemble graph may not becorrespond to a valid CPDAG or PAG and is not guaranteed to represent a causal graph.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- pcStable(sim$data)
g.boot <- bootstrap(sim$data, g)
print(g.boot)
print(g.boot$stabilities[1:6,])
```

boss *Runs the BOSS causal discovery algorithm on the dataset*

## Description

Runs the BOSS causal discovery algorithm on the dataset

## Usage

```
boss(
  data,
  numStarts = 3L,
  penalty = 2,
  threads = -1L,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| numStarts | The number of restarts (with different randomly sampled initial topological orders). Reduces the variance that can result from being stuck with an unfavorable initial starting order. |
| penalty | A numeric value that represents the strength of the penalty for model complexity. The default value is 2, which corresponds to twice the BIC penalty. |
| threads | An integer value denoting the number of threads to use for parallelization. The default value is -1, which will all available CPUs. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

The CPDAG learned by BOSS

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- boss(sim$data)
print(g)
```

coxmgm *Calculate the CoxMGM graph on a dataset.*

## Description

Calculate the CoxMGM graph on a dataset. The dataset must contain at least one censored variable formatted as Surv object from the survival package.

## Usage

```
coxmgm(
  data,
  lambda = as.numeric(c(0.2, 0.2, 0.2, 0.2, 0.2)),
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. All censored variables must be a survival::Surv object. Any rows with missing values will be dropped. |
| lambda | A numeric vector of five values for the regularization parameter lambda: the first for continuous-continuous edges, the second for continuous-discrete, the third for discrete-discrete, the fourth for continuous-survival, and the fifth for discrete-survival. Defaults to c(0.2, 0.2, 0.2, 0.2, 0.2). If a single value is provided, all three values in the vector will be set to that value. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print updates on the progress of optimizing MGM. The default is FALSE. |

## Value

The calculated CoxMGM graph

## Examples

```
sim <- simRandomDAG(200, 25, 1)
ig <- coxmgm(sim$data)
print(ig)
```

coxmgmCV *Implements k-fold cross-validation for CoxMGM*

## Description

Calculate the solution path for a CoxMGM graph on a dataset with k-fold cross-validation. The dataset must contain at least one censored variable formatted as Surv object from the survival package. This function returns the graph that minimizes negative log(pseudolikelihood) and the graph selected by the one standard error rule.

## Usage

```
coxmgmCV(
  data,
  lambdas = NULL,
  nLambda = 30L,
  nfolds = 5L,
  foldid = NULL,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the CoxMGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. All censored variables must be a survival::Surv object. Any rows with missing values will be dropped. |
| lambdas | A numeric vector containing the values of lambda to learn an MGM with. The default value is NULL, in which case a log-spaced vector of nLambda values for lambda will be supplied instead. |
| nLambda | A numeric value indicating the number of lambda values to test when the lambdas vector is NULL. The default is 30. |
| nfolds | An integer value defining the number of folds to be used for cross-validation if foldid is NULL. The default value is 5. |
| foldid | An integer vector containing values in the range of 1 to K for each sample that identifies which test set that sample belongs to. This enables users to define their own cross-validation splits, for example in the case stratified cross-validation is needed. The default value is NULL. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphCV object that contains the minimum and one standard error rule selected graphs.

## Examples

```
sim <- simRandomDAG(200, 25, 1)
ig.cv <- coxmgmCV(sim$data)
print(ig.cv)
```

---

coxmgmPath *Estimates a solution path for CoxMGM*

---

## Description

Calculate the solution path for a CoxMGM graph on a dataset. The dataset must contain at least one censored variable formatted as Surv object from the survival package. It also returns the models selected by the BIC and AIC scores.

## Usage

```
coxmgmPath(data, lambdas = NULL, nLambda = 30L, rank = FALSE, verbose = FALSE)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the CoxMGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. All censored variables must be a survival::Surv object. Any rows with missing values will be dropped. |
| lambdas | A numeric vector containing the values of lambda to learn an MGM with. The default value is NULL, in which case a log-spaced vector of nLambda values for lambda will be supplied instead. |
| nLambda | A numeric value indicating the number of lambda values to test when the lambdas vector is NULL. The default is 30. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphPath object that contains CoxMGM graphs learned by the solution path, as well as the BIC and AIC selected models

## Examples

```
sim <- simRandomDAG(200, 25, 1)
ig.path <- coxmgmPath(sim$data)
print(ig.path)
```

---

cpdag *Calculate the CPDAG for a given DAG*

---

## Description

Create the completed partially directed acyclic graph (CPDAG) for the input directed acyclic graph (DAG). The CPDAG represents the Markov equivalence class of the true cauasl DAG. The PC algorithms are only identifiable up to the Markov equivalence class, so assessments of causal structure recovery should be compared to the CPDAG rather than the causal DAG.

## Usage

```
cpdag(graph)
```

## Arguments

| | |
|---|---|
| graph | The graph object used to generate the CPDAG. Should be the ground-truth causal DAG |

## Value

The CPDAG corresponding to the input DAG

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
sim$cpdag <- cpdag(sim$graph)
print(sim$cpdag)
```

---

createKnowledge *A function to create a prior knowledge object for use with causal discovery algorithms*

---

## Description

A function to create a prior knowledge object for use with causal discovery algorithms

### Usage

```
createKnowledge(
  tiers = list(),
  forbiddenWithinTier = NULL,
  forbidden = list(),
  required = list()
)
```

### Arguments

| | |
|---|---|
| tiers | A list containing ordered vectors of variables where variables in tier t can only be ancestors of variables in tiers t+1 ... T and descendants of variables in tiers (1 .. t-1). If tiers are used, all variables must be in a tier, and no variable can be in multiple tiers. |
| forbiddenWithinTier | A vector of logical values indicating whether edges are allowed between variables in a given tier. The value is NULL by default, which results in forbiddenWithinTier being set to FALSE for each tier. |
| forbidden | A list containing vectors of node pairs that forbid a specific directed edge. For example, to forbid A –> B, add c("A", "B") to forbidden. |
| required | A list containing vectors of node pairs that require the presence of a specific directed edge. For example, to require B –> A, add c("B", "A") to required. |

### Value

A knowledge object that can be passed to causal discovery algorithms.

---

## fciCV *Implements k-fold cross-validation for FCI-Stable*

---

### Description

Runs k-fold cross-validation to select the value of alpha and orientation rule for FCI-Stable. Returns a graphCV object containing the causal graphical models that minimize the negative log(pseudo-likelihood) and the sparsest model within one standard error of the minimum.

### Usage

```
fciCV(
  data,
  initialGraph = NULL,
  knowledge = NULL,
  orientRule = as.character(c("majority", "maxp", "conservative")),
  alphas = NULL,
  nfolds = 5L,
  foldid = NULL,
```

```
  threads = -1L,
  fdr = FALSE,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| initialGraph | An undirected rCausalMGM graph object containing the initial skeleton of adjacencies used in the causal discovery algorithm. This graph can be learned by 'mgm' or learned by another method and imported into an undirected rCausalMGM graph object from its adjacency matrix. The default is NULL, in which case a fully connected graph is used as the initial skeleton. |
| knowledge | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |
| orientRule | A vector of strings to determine which of the orientation rules to test in the cross-validation procedure to select the optimal model. The default is a vector that contains the "majority", "maxp", and "conservative" orientation rules. |
| alphas | A numeric vector containing values of alpha to test in the cross-validation procedure. The default value is NULL, in which case we set alpha = c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2). |
| nfolds | An integer value defining the number of folds to be used for cross-validation if foldid is NULL. The default value is 5. |
| foldid | An integer vector containing values in the range of 1 to K for each sample that identifies which test set that sample belongs to. This enables users to define their own cross-validation splits, for example in the case stratified cross-validation is needed. The default value is NULL. |
| threads | An integer value denoting the number of threads to use for parallelization of independence tests. The default value is -1, which will all available CPUs. |
| fdr | A logical value indicating whether to use false discovery rate control for the discovery of adjacencies in the causal graph. The default value is FALSE. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphCV object containing the PAGs selected by the minimum and one standard error rule.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g.cv <- fciCV(sim$data)
print(g.cv)
```

---

`fciStable` *Runs the causal discovery algorithm FCI-Stable on a dataset.*

---

## Description

Runs the causal discovery algorithm FCI-Stable on a dataset. The FCI-Stable algorithm is designed to recover the Markov equivalence class of causal MAGs that could give rise to the observed conditional independence relationships in the causally insufficient case. This means that FCI-Stable can still learn the Markov equivalence class of the true MAG even in the presence of latent confounders and/or selection bias. The resulting graph is a partial ancestral graph (PAG).

## Usage

```
fciStable(
  data,
  initialGraph = NULL,
  knowledge = NULL,
  orientRule = as.character(c("majority")),
  alpha = 0.05,
  threads = -1L,
  possDsep = TRUE,
  fdr = FALSE,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| `data` | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| `initialGraph` | An undirected rCausalMGM graph object containing the initial skeleton of adjacencies used in the causal discovery algorithm. This graph can be learned by 'mgm' or learned by another method and imported into an undirected rCausalMGM graph object from its adjacency matrix. The default is NULL, in which case a fully connected graph is used as the initial skeleton. |
| `knowledge` | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |

| | |
|---|---|
| orientRule | Determines which of the four possible orientation rules will be utilized to orient colliders in the FCI-Stable algorithm. Possible options are "majority", "maxp", "conservative", and "sepsets". The default value is "majority". Additionally, a vector of valid orientation rules can be provided, and fciStable will return a list containing the graphs learned with each. |
| alpha | A numeric value containing the significance threshold alpha for the conditional independence tests used during constraint-based causal discovery. This parameter directly controls graph sparsity, with low values of alpha yielding sparse graphs and high values yielding dense graphs. The default value is 0.05. |
| threads | An integer value denoting the number of threads to use for parallelization of independence tests. The default value is -1, which will all available CPUs. |
| possDsep | A logical value indicating whether to perform the possible-D-Sep search stage of the FCI algorithm. The possible-D-Sep search is necessaey fro correctness but can be computationally expensive in dense or high-dimensional or graphs. If set to FALSE, the RFCI rule R0 will be applied to remove some of the extraneous adjacencies that would have been removed by possible-D-Sep search. The default value is TRUE. |
| fdr | A logical value indicating whether to use false discovery rate control for the discovery of adjacencies in the causal graph. The default value is FALSE. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

The PAG learned by FCI-Stable.

## Examples

```
sim <- simRandomDAG(200, 50, deg=3)
g <- fciStable(sim$data)
print(g)
```

---

# fciStars *Implements StARS for FCI-Stable*

---

## Description

Runs StARS to select the value of alpha for FCI-Stable based on adjacency stability. Returns a graphSTARS object containing the PAG selected by StARS and the adjacency instabilities for each alpha.

## Usage

```
fciStars(
  data,
  initialGraph = NULL,
  knowledge = NULL,
  orientRule = as.character(c("majority")),
  alphas = NULL,
  gamma = 0.01,
  numSub = 20L,
  subSize = -1L,
  leaveOneOut = FALSE,
  threads = -1L,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| initialGraph | An undirected rCausalMGM graph object containing the initial skeleton of adjacencies used in the causal discovery algorithm. This graph can be learned by 'mgm' or learned by another method and imported into an undirected rCausalMGM graph object from its adjacency matrix. The default is NULL, in which case a fully connected graph is used as the initial skeleton. |
| knowledge | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |
| orientRule | Determines which of the four possible orientation rules will be utilized to orient colliders in the FCI-Stable algorithm. Possible options are "majority", "maxp", "conservative", and "sepsets". The default value is "majority". |
| alphas | A numeric vector containing values of alpha to test in the cross-validation procedure. The default value is NULL, in which case we set alpha = c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2). |
| gamma | The threshold for edge instability. The default value is 0.01, and it is not recommended to change this value. |
| numSub | The number of subsamples of the dataset used to estimate edge instability. The default value is 20. |
| subSize | The number of samples to be drawn without replacement for each subsample. The default value is -1. When subSize is -1, it is set to min(floor(0.75 * N), floor(10*sqrt(N))), where N is the number of samples. |
| leaveOneOut | If TRUE, performs leave-one-out subsampling. Defaults to FALSE. |

| | |
|---|---|
| threads | An integer value denoting the number of threads to use for parallelization of independence tests. The default value is -1, which will all available CPUs. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphSTARS object containing the PAG selected by StARS and the instabilities at each value of alpha.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g.stars <- fciStars(sim$data)
print(g.stars)
```

---

# graphTable

*A function to generate a data.frame for objects from graph class. It incorporates adjacency and orientation frequency if estimates of edge stability are available.*

---

## Description

A function to generate a data.frame for objects from graph class. It incorporates adjacency and orientation frequency if estimates of edge stability are available.

## Usage

```
graphTable(graph, stabilities = NULL)
```

## Arguments

| | |
|---|---|
| graph | The graph object |
| stabilities | The stability data.frame from bootstrapping or StEPS. If NULL, the stabilities entry of the graph object is used. If that is also NULL, only edge interactions are returned. The default is NULL |

## Value

A data.frame containing source, target, and interaction columns for each edge in the graph. If stabilities are available, then the adjFrequency and orientation frequencies (if applicable) are returned for each edge.

grasp *Runs the GRaSP causal discovery algorithm on the dataset*

## Description

Runs the GRaSP causal discovery algorithm on the dataset

## Usage

```
grasp(
  data,
  depth = 2L,
  numStarts = 3L,
  penalty = 2,
  bossInit = FALSE,
  threads = -1L,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| depth | The maximum search depth used in the depth-first search in GRaSP. |
| numStarts | The number of restarts (with different randomly sampled initial topological orders). Reduces the variance that can result from being stuck with an unfavorable initial starting order. |
| penalty | A numeric value that represents the strength of the penalty for model complexity. The default value is 2, which corresponds to twice the BIC penalty. |
| bossInit | A logical value indicating whether to initialize the causal order for GRaSP with the forward search procedure of BOSS. |
| threads | An integer value denoting the number of threads to use for parallelization. The default value is -1, which will all available CPUs. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

The CPDAG learned by GRaSP

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- grasp(sim$data)
print(g)
```

---

growShrinkMB *Implements Grow-Shrink algorithm for Markov blanket identification*

---

## Description

Runs the Grow-Shrink algorithm to find the Markov blanket of a feature in a dataset

## Usage

```
growShrinkMB(data, target, penalty = 1, rank = FALSE, verbose = FALSE)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| target | A string denoting the name of the target variable to identify the Markov blanket of. |
| penalty | A numeric value that represents the strength of the penalty for model complexity. The default value is 1, which corresponds to the BIC score. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

The list of features in the Markov Blanket and the BIC score

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
mb <- growShrinkMB(sim$data, "X1")
print(mb)
```

`loadGraph` *Load a graph from a ".txt" file*

## Description

Load a graph from a ".txt" file

## Usage

```
loadGraph(filename)
```

## Arguments

| | |
|---|---|
| `filename` | The graph file |

## Value

The graph as a graph object, which can be passed into search functions

`mgm` *Calculate the Mixed Graphical Model (MGM) graph on a dataset.*

## Description

Calculate the MGM graph on a dataset. The dataset may contain continuous and discrete variables. In the case that it contains only continuous variables, MGM reduces to a pseudo-likelihood estimate of the graphical LASSO, and in the case that it contains only discrete variables, MGM reduces to a pseudo-likelihood estimate of a pairwise Markov random field.

## Usage

```
mgm(data, lambda = as.numeric(c(0.2, 0.2, 0.2)), rank = FALSE, verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `data` | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| `lambda` | A numeric vector of three values for the regularization parameter lambda: the first for continuous-continuous edges, the second for continuous-discrete, and the third for discrete-discrete. Defaults to c(0.2, 0.2, 0.2). If a single value is provided, all three values in the vector will be set to that value. |
| `rank` | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| `verbose` | A logical value indicating whether to print updates on the progress of optimizing MGM. The default is FALSE. |

## Value

The calculated MGM graph

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- mgm(sim$data)
print(g)
```

---

mgmCV *Implements k-fold cross-validation for MGM*

---

## Description

Calculate the solution path for an MGM graph on a dataset with k-fold cross-validation. This function returns the graph that minimizes negative log(pseudolikelihood) and the graph selected by the one standard error rule.

## Usage

```
mgmCV(
  data,
  lambdas = NULL,
  nLambda = 30L,
  nfolds = 5L,
  foldid = NULL,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| lambdas | A numeric vector containing the values of lambda to learn an MGM with. The default value is NULL, in which case a log-spaced vector of nLambda values for lambda will be supplied instead. |
| nLambda | A numeric value indicating the number of lambda values to test when the lambdas vector is NULL. The default is 30. |
| nfolds | An integer value defining the number of folds to be used for cross-validation if foldid is NULL. The default value is 5. |
| foldid | An integer vector containing values in the range of 1 to K for each sample that identifies which test set that sample belongs to. This enables users to define their own cross-validation splits, for example in the case stratified cross-validation is needed. The default value is NULL. |

| | |
|---|---|
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphCV object that contains the minimum and one standard error rule selected graphs.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
ig.cv <- mgmCV(sim$data)
print(ig.cv)
```

---

# mgmfciCV *Implements k-fold cross-validation for MGM-FCI-Stable*

---

## Description

Runs k-fold cross-validation to select the value of lambda, alpha, and the orientation rule for MGM-FCI-Stable. Returns a graphCV object containing the causal graphical models that minimize the negative log(pseudo-likelihood) and the sparsest model within one standard error of the minimum.

## Usage

```
mgmfciCV(
  data,
  knowledge = NULL,
  cvType = "random",
  orientRule = as.character(c("majority", "maxp", "conservative")),
  lambdas = NULL,
  nLambda = 20L,
  alphas = NULL,
  numPoints = 60L,
  nfolds = 5L,
  foldid = NULL,
  threads = -1L,
  fdr = FALSE,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| knowledge | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |
| cvType | A string determining whether to perform random search or grid search cross-validation, indicated by "random" or "grid" respectively. The default value is "random". |
| orientRule | A vector of strings to determine which of the orientation rules to test in the cross-validation procedure to select the optimal model. The default is a vector that contains the "majority", "maxp", and "conservative" orientation rules. |
| lambdas | A numeric vector containing the values of lambda to learn an MGM with. The default value is NULL, in which case a log-spaced vector of nLambda values for lambda will be supplied instead. |
| nLambda | A numeric value indicating the number of lambda values to test when the lambdas vector is NULL. The default is 20. |
| alphas | A numeric vector containing values of alpha to test in the cross-validation procedure. The default value is NULL, in which case we set alpha = c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2). |
| numPoints | An integer value containing indicating the number of samples to draw uniformly from the search space if performing random search cross-validation. The default is 60, the number of points required to have a 5% chance of sampling a model in the top 5% of the search space. |
| nfolds | An integer value defining the number of folds to be used for cross-validation if foldid is NULL. The default value is 5. |
| foldid | An integer vector containing values in the range of 1 to K for each sample that identifies which test set that sample belongs to. This enables users to define their own cross-validation splits, for example in the case stratified cross-validation is needed. The default value is NULL. |
| threads | An integer value denoting the number of threads to use for parallelization of independence tests. The default value is -1, which will all available CPUs. |
| fdr | A logical value indicating whether to use false discovery rate control for the discovery of adjacencies in the causal graph. The default value is FALSE. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphCV object containing the PAGs selected by the minimum and one standard error rule.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g.cv <- mgmfciCV(sim$data)
print(g.cv)
```

---

`mgmPath` *Estimates a solution path for MGM*

---

## Description

Calculate the solution path for an MGM graph on a dataset. It also returns the models selected by the BIC and AIC scores.

## Usage

```
mgmPath(data, lambdas = NULL, nLambda = 30L, rank = FALSE, verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `data` | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| `lambdas` | A numeric vector containing the values of lambda to learn an MGM with. The default value is NULL, in which case a log-spaced vector of nLambda values for lambda will be supplied instead. |
| `nLambda` | A numeric value indicating the number of lambda values to test when the lambdas vector is NULL. The default is 30. |
| `rank` | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| `verbose` | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphPath object that contains MGM graphs learned by the solution path, as well as the BIC and AIC selected models

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
ig.path <- mgmPath(sim$data)
print(ig.path)
```

mgmpcCV *Implements k-fold cross-validation for MGM-PC-Stable*

## Description

Runs k-fold cross-validation to select the value of lambda, alpha, and the orientation rule for MGM-PC-Stable. Returns a graphCV object containing the causal graphical models that minimize the negative log(pseudo-likelihood) and the sparsest model within one standard error of the minimum.

## Usage

```
mgmpcCV(
  data,
  knowledge = NULL,
  cvType = "random",
  orientRule = as.character(c("majority", "maxp", "conservative")),
  lambdas = NULL,
  nLambda = 20L,
  alphas = NULL,
  numPoints = 60L,
  nfolds = 5L,
  foldid = NULL,
  threads = -1L,
  fdr = FALSE,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| knowledge | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |
| cvType | A string determining whether to perform random search or grid search cross-validation, indicated by "random" or "grid" respectively. The default value is "random". |
| orientRule | A vector of strings to determine which of the orientation rules to test in the cross-validation procedure to select the optimal model. The default is a vector that contains the "majority", "maxp", and "conservative" orientation rules. |

| | |
|---|---|
| lambdas | A numeric vector containing the values of lambda to learn an MGM with. The default value is NULL, in which case a log-spaced vector of nLambda values for lambda will be supplied instead. |
| nLambda | A numeric value indicating the number of lambda values to test when the lambdas vector is NULL. The default is 20. |
| alphas | A numeric vector containing values of alpha to test in the cross-validation procedure. The default value is NULL, in which case we set alpha = c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2). |
| numPoints | An integer value containing indicating the number of samples to draw uniformly from the search space if performing random search cross-validation. The default is 60, the number of points required to have a 5% chance of sampling a model in the top 5% of the search space. |
| nfolds | An integer value defining the number of folds to be used for cross-validation if foldid is NULL. The default value is 5. |
| foldid | An integer vector containing values in the range of 1 to K for each sample that identifies which test set that sample belongs to. This enables users to define their own cross-validation splits, for example in the case stratified cross-validation is needed. The default value is NULL. |
| threads | An integer value denoting the number of threads to use for parallelization of independence tests. The default value is -1, which will all available CPUs. |
| fdr | A logical value indicating whether to use false discovery rate control for the discovery of adjacencies in the causal graph. The default value is FALSE. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphCV object containing the CPDAGs selected by the minimum and one standard error rule.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g.cv <- mgmpcCV(sim$data)
print(g.cv)
```

---

# moral *Calculate the moral graph for a given DAG*

---

## Description

Create the moral graph for the input directed acyclic graph (DAG). The moral graph is the undirected graphical model that is equivalent to the input DAG.

### Usage

```
moral(graph)
```

### Arguments

| | |
|---|---|
| graph | The graph object used to generate the moral graph. Should be the ground-truth causal DAG |

### Value

The moral graph corresponding to the input DAG

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
sim$moral <- moral(sim$graph)
print(sim$moral)
```

---

## pag *Calculate the PAG for a given DAG and set of latent variables*

---

### Description

Create the partial ancestral graph (PAG) for the input directed acyclic graph (DAG). The PAG represents the Markov equivalence class of the true cauasl MAG. The FCI algorithms are only identifiable up to the Markov equivalence class, so assessments of causal structure recovery should be compared to the PAG rather than the causal MAG.

### Usage

```
pag(graph, latent = NULL)
```

### Arguments

| | |
|---|---|
| graph | The graph object used to generate the PAG. Should be the ground-truth causal DAG |
| latent | The names of latent (unobserved) variables in the causal DAG. The default is NULL. |

### Value

The PAG corresponding to the input DAG

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
sim$pag <- pag(sim$graph)
print(sim$pag)
```

pcCV *Implements k-fold cross-validation for PC-Stable*

## Description

Runs k-fold cross-validation to select the value of alpha and orientation rule for PC-Stable. Returns a graphCV object containing the causal graphical models that minimize the negative log(pseudo-likelihood) and the sparsest model within one standard error of the minimum.

## Usage

```
pcCV(
  data,
  initialGraph = NULL,
  knowledge = NULL,
  orientRule = as.character(c("majority", "maxp", "conservative")),
  alphas = NULL,
  nfolds = 5L,
  foldid = NULL,
  threads = -1L,
  fdr = FALSE,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| initialGraph | An undirected rCausalMGM graph object containing the initial skeleton of adjacencies used in the causal discovery algorithm. This graph can be learned by 'mgm' or learned by another method and imported into an undirected rCausalMGM graph object from its adjacency matrix. The default is NULL, in which case a fully connected graph is used as the initial skeleton. |
| knowledge | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |
| orientRule | A vector of strings to determine which of the orientation rules to test in the cross-validation procedure to select the optimal model. The default is a vector that contains the "majority", "maxp", and "conservative" orientation rules. |

| | |
|---|---|
| alphas | A numeric vector containing values of alpha to test in the cross-validation procedure. The default value is NULL, in which case we set alpha = c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2). |
| nfolds | An integer value defining the number of folds to be used for cross-validation if foldid is NULL. The default value is 5. |
| foldid | An integer vector containing values in the range of 1 to K for each sample that identifies which test set that sample belongs to. This enables users to define their own cross-validation splits, for example in the case stratified cross-validation is needed. The default value is NULL. |
| threads | An integer value denoting the number of threads to use for parallelization of independence tests. The default value is -1, which will all available CPUs. |
| fdr | A logical value indicating whether to use false discovery rate control for the discovery of adjacencies in the causal graph. The default value is FALSE. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphCV object containing the CPDAGs selected by the minimum and one standard error rule.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g.cv <- pcCV(sim$data)
print(g.cv)
```

---

# pcStable *Runs the causal discovery algorithm PC-Stable on a dataset.*

---

## Description

Runs the causal discovery algorithm PC-Stable on a dataset. The PC-Stable algorithm is designed to recover the Markov equivalence class of causal DAGs that could give rise to the observed conditional independence relationships under the assumption of causal sufficiency. A dataset is said to be causally sufficient if all variables relevant to the causal process are observed (i.e. there are no latent confounders). The resulting graph is a completed partially directed acyclic graph (CPDAG) containing directed edges where the causal orientation can be uniquely determined and an undirected edge where multiple orientations are possible.

## Usage

```
pcStable(
  data,
  initialGraph = NULL,
  knowledge = NULL,
  orientRule = as.character(c("majority")),
  alpha = 0.05,
  threads = -1L,
  fdr = FALSE,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| initialGraph | An undirected rCausalMGM graph object containing the initial skeleton of adjacencies used in the causal discovery algorithm. This graph can be learned by 'mgm' or learned by another method and imported into an undirected rCausalMGM graph object from its adjacency matrix. The default is NULL, in which case a fully connected graph is used as the initial skeleton. |
| knowledge | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |
| orientRule | Determines which of the four possible orientation rules will be utilized to orient colliders in the PC-Stable algorithm. Possible options are "majority", "maxp", "conservative", and "sepsets". The default value is "majority". Additionally, a vector of valid orientation rules can be provided, and pcStable will return a list containing the graphs learned with each. |
| alpha | A numeric value containing the significance threshold alpha for the conditional independence tests used during constraint-based causal discovery. This parameter directly controls graph sparsity, with low values of alpha yielding sparse graphs and high values yielding dense graphs. The default value is 0.05. |
| threads | An integer value denoting the number of threads to use for parallelization of independence tests. The default value is -1, which will all available CPUs. |
| fdr | A logical value indicating whether to use false discovery rate control for the discovery of adjacencies in the causal graph. The default value is FALSE. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

The CPDAG learned by PC-Stable.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- pcStable(sim$data)
print(g)
```

---

pcStars *Implements StARS for PC-Stable*

---

## Description

Runs StARS to select the value of alpha for PC-Stable based on adjacency stability. Returns a graphSTARS object containing the CPDAG selected by StARS and the adjacency instabilities for each alpha.

## Usage

```
pcStars(
  data,
  initialGraph = NULL,
  knowledge = NULL,
  orientRule = as.character(c("majority")),
  alphas = NULL,
  gamma = 0.01,
  numSub = 20L,
  subSize = -1L,
  leaveOneOut = FALSE,
  threads = -1L,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| initialGraph | An undirected rCausalMGM graph object containing the initial skeleton of adjacencies used in the causal discovery algorithm. This graph can be learned by 'mgm' or learned by another method and imported into an undirected rCausalMGM graph object from its adjacency matrix. The default is NULL, in which case a fully connected graph is used as the initial skeleton. |

| | |
|---|---|
| knowledge | A knowledge object containing prior knowledge about the causal interactions in a dataset. This knowledge can be used to forbid or require certain edges in the causal graph, helping to inform causal discovery an prevent orientations known to be nonsensical. The default is NULL, in which case no prior knowledge is provided to the causal discovery algorithm. |
| orientRule | Determines which of the four possible orientation rules will be utilized to orient colliders in the PC-Stable algorithm. Possible options are "majority", "maxp", "conservative", and "sepsets". The default value is "majority". |
| alphas | A numeric vector containing values of alpha to test in the cross-validation procedure. The default value is NULL, in which case we set alpha = c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2). |
| gamma | The threshold for edge instability. The default value is 0.01, and it is not recommended to change this value. |
| numSub | The number of subsamples of the dataset used to estimate edge instability. The default value is 20. |
| subSize | The number of samples to be drawn without replacement for each subsample. The default value is -1. When subSize is -1, it is set to min(floor(0.75 * N), floor(10*sqrt(N))), where N is the number of samples. |
| leaveOneOut | If TRUE, performs leave-one-out subsampling. Defaults to FALSE. |
| threads | An integer value denoting the number of threads to use for parallelization of independence tests. The default value is -1, which will all available CPUs. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphSTARS object containing the CPDAG selected by StARS and the instabilities at each value of alpha.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g.stars <- pcStars(sim$data)
print(g.stars)
```

---

# plot.graph *A plot override function for the graph class*

---

## Description

A plot override function for the graph class

### Usage

```
## S3 method for class 'graph'
plot(x, nodes = c(), nodeAttr = list(), edgeAttr = list(), ...)
```

### Arguments

| | |
|---|---|
| x | The graph object |
| nodes | A subset of nodes in the graph to plot. If only a single node is supplied, then that node and its Markov blanket will be plotted. |
| nodeAttr | A list of options to modify graph nodes (e.g. fontsize). |
| edgeAttr | A list of options to modify graph edges. |
| ... | Additional plot arguments |

### Value

No return value, the function plots a graph object.

---

## plot.graphCV *A plot override function for the graphCV class*

---

### Description

A plot override function for the graphCV class

### Usage

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

### Arguments

| | |
|---|---|
| x | The graph object |
| ... | Additional plot arguments |

### Value

No return value. This function plots graph sparsity, quantified by the average Markov blanket size for causal graphs or the regularization parameter for undirected graphs, against -log(pseudo-likelihood), with lines indicating the selected models.

## plot.graphPath *A plot override function for the graphPath class*

### Description

A plot override function for the graphPath class

### Usage

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

### Arguments

| | |
|---|---|
| x | The graph object |
| ... | Additional plot arguments |

### Value

No return value. This function plots graph sparsity, quantified by the regularization parameter, against the AIC and BIC scores along a solution path, with lines indicating the selected models.

## plot.graphSTARS *A plot override function for the graphSTARS class*

### Description

A plot override function for the graphSTARS class

### Usage

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

### Arguments

| | |
|---|---|
| x | The graph object |
| ... | Additional plot arguments |

### Value

No return value. This function plots graph sparsity, quantified by the significance threshold alpha, against the average edge instability used for stability-based model selection, with a horizontal line indicating the instability threshold and a vertical line indicating the selected threshold.

## plot.graphSTEPS *A plot override function for the graphSTEPS class*

### Description

A plot override function for the graphSTEPS class

### Usage

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

### Arguments

| | |
|---|---|
| x | The graph object |
| ... | Additional plot arguments |

### Value

No return value. This function plots graph sparsity, quantified by the regularization parameters, against the average edge instability used for stability-based model selection, with a horizontal line indicating the instability threshold and vertical lines indicating the selected regularization parameters.

## print.graph *A print override function for the graph class*

### Description

A print override function for the graph class

### Usage

```
## S3 method for class 'graph'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | The graph object |
| ... | Additional print arguments |

### Value

No return value, the function prints a summary of the graph object.

## print.graphCV *A print override function for the graphCV class*

### Description

A print override function for the graphCV class

### Usage

```
## S3 method for class 'graphCV'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | The graphCV object |
| ... | Additional print arguments |

### Value

No return value, the function prints a summary of the graphCV object.

## print.graphPath *A print override function for the graphPath class*

### Description

A print override function for the graphPath class

### Usage

```
## S3 method for class 'graphPath'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | The graphPath object |
| ... | Additional print arguments |

### Value

No return value, the function prints a summary of the graphPath object.

## print.graphSTARS *A print override function for the graphSTARS class*

### Description

A print override function for the graphSTARS class

### Usage

```
## S3 method for class 'graphSTARS'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | The graphSTARS object |
| ... | Additional print arguments |

### Value

No return value, the function prints a summary of the graphSTARS object.

## print.graphSTEPS *A print override function for the graphSTEPS class*

### Description

A print override function for the graphSTEPS class

### Usage

```
## S3 method for class 'graphSTEPS'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | The graphSTEPS object |
| ... | Additional print arguments |

### Value

No return value, the function prints a summary of the graphSTEPS object.

## print.knowledge *A print override function for the knowledge class*

### Description

A print override function for the knowledge class

### Usage

```
## S3 method for class 'knowledge'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | The knowledge object |
| ... | Additional print arguments |

### Value

No return value, the function prints a summary of the knowledge object.

## printGraph *Display a graph object as text.*

### Description

Display a graph object as text. This is the same format as written in ".txt" save files.

### Usage

```
printGraph(graph)
```

### Arguments

| | |
|---|---|
| graph | The graph object |

### Value

No return value, this function prints the full details of a graph object, including nodes, edges, the algorithm used to learn the model, and relevant hyperparameters.

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- mgm(sim$data)
printGraph(g)
```

## prMetrics *Combined adjaceny and orientation precision-recall metrics*

### Description

Calculate the precision, recall, F1, and Matthew's Correlation Coefficient (MCC) for the adjacencies and orientations of an estimated graph compared to the ground truth. This is the concatenated output of the adjacency PR metrics and the orientation PR metrics.

### Usage

```
prMetrics(estimate, groundTruth, groundTruthDAG = NULL)
```

### Arguments

| | |
|---|---|
| estimate | An estimated graph object |
| groundTruth | A ground truth graph object of the same type as the estimated graph object |
| groundTruthDAG | A ground truth graph object containing the true causal DAG. Only necessary for calculating the or precision, recall, F1, and MCC for partial ancestral graphs (PAGs) |

### Value

The orientation precision, recall, F1, and MCC, between the two graph objects

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- pcStable(sim$data)
prMetrics(g, cpdag(sim$graph))
```

## prMetricsAdjacency *Adjacency Precision-Recall Metrics*

### Description

Calculate the skeleton precision, recall, F1, and Matthew's Correlation Coefficient (MCC) between an estimated and ground truth graph.

### Usage

```
prMetricsAdjacency(estimate, groundTruth)
```

### Arguments

| | |
|---|---|
| estimate | An estimated graph object |
| groundTruth | A ground truth graph object |

### Value

The skeleton precision, recall, F1, and MCC, between the two graph objects

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- pcStable(sim$data)
prMetricsAdjacency(g, cpdag(sim$graph))
```

---

## prMetricsCausal *Causal Orientaion Precision-Recall Metrics for CPDAGs*

---

### Description

Calculate the causal orientation precision, recall, and F1 between an estimated CPDAG and ground truth graph causal DAG.

### Usage

```
prMetricsCausal(estimate, groundTruthDAG)
```

### Arguments

| | |
|---|---|
| estimate | An estimated graph object. |
| groundTruthDAG | A ground truth graph object of the type "directed acyclic graph". |

### Value

The causal orientation precision, recall, and F1 between the two graph objects

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- pcStable(sim$data)
prMetricsCausal(g, sim$graph)
```

## prMetricsOrientation *Orientation Precision-Recall Metrics*

### Description

Calculate the orientation precision, recall, F1, and Matthew's Correlation Coefficient (MCC) between an estimated and ground truth graph.

### Usage

```
prMetricsOrientation(estimate, groundTruth, groundTruthDAG = NULL)
```

### Arguments

| | |
|---|---|
| estimate | An estimated graph object |
| groundTruth | A ground truth graph object of the same type as the estimated graph object |
| groundTruthDAG | A ground truth graph object containing the true causal DAG. Only necessary for calculating the or precision, recall, F1, and MCC for partial ancestral graphs (PAGs) |

### Value

The orientation precision, recall, F1, and MCC, between the two graph objects

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- pcStable(sim$data)
prMetricsOrientation(g, cpdag(sim$graph))
```

## saveGraph *Save a graph to a file. Supported file types are ".txt" and ".sif".*

### Description

Save a graph to a file. Supported file types are ".txt" and ".sif".

### Usage

```
saveGraph(graph, filename)
```

### Arguments

| | |
|---|---|
| graph | The graph object |
| filename | The graph filename |

### Value

No return value. This function saves the full details of a graph object to a .txt file, including nodes, edges, the algorithm used to learn the model, and relevant hyperparameters. This format can then be read back into R with the loadGraph function.

---

## SHD *Structural Hamming Distance (SHD)*

---

### Description

Calculate the Structural Hamming Distance (SHD) between two graphs.

### Usage

```
SHD(graph1, graph2)
```

### Arguments

| | |
|---|---|
| graph1 | A graph object |
| graph2 | A graph object |

### Value

The SHD btween the two graph objects

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
g <- pcStable(sim$data)
SHD(g, cpdag(sim$graph))
```

---

## simRandomDAG *A function to simulate a random forward DAG from a SEM model.*

---

### Description

A function to simulate a random forward DAG from a SEM model.

## Usage

```
simRandomDAG(
  n = 1000,
  p = 50,
  r = 0,
  discFrac = 0.5,
  deg = 3,
  coefMin = 0.5,
  coefMax = 1.5,
  noiseMin = 1,
  noiseMax = 2,
  censorRate = 0.3,
  seed = NULL
)
```

## Arguments

| | |
|---|---|
| n | The sample size of the generated dataset. The default is 1000. |
| p | The number of features in the generated dataset. The default is 50. |
| r | The number of censored features in the generated dataset. The default is 0. |
| discFrac | The fraction of variables in the dataset that are discrete. The default is 0.5. |
| deg | The average graph degree for the simulated graph. The default is 3. |
| coefMin | The lower bound on the magnitude of the effect size. The default is 0.5. |
| coefMax | The upper bound on the magnitude of the effect size. The default is 1.5. |
| noiseMin | The lower bound on the standard deviation of the Gaussian noise for continuous variables. The default is 1. |
| noiseMax | The upper bound on the standard deviation of the Gaussian noise for continuous variables. The default is 2. |
| censorRate | The rate censored variables are censored at. The default is 0.3. |
| seed | The random seed for generating the simulated DAG. The default is NULL. |

## Value

A list containing the simulated dataset and the corresponding ground truth causal DAG.

## Examples

```
sim <- simRandomDAG(200, 25)
print(sim$graph)
print(sim$data[1:6,])
```

## skeleton *Calculate the undirected skeleton for a given DAG*

### Description

Create the skeleton graph for the input directed acyclic graph (DAG). The skeleton graph is the undirected graph that contains the same adjacencies as the input DAG.

### Usage

```
skeleton(graph)
```

### Arguments

| | |
|---|---|
| graph | The graph object used to generate the skeleton graph. Should be the ground-truth causal DAG |

### Value

The skeleton graph corresponding to the input DAG

### Examples

```
sim <- simRandomDAG(200, 25, deg=2)
sim$skeleton <- skeleton(sim$graph)
print(sim$skeleton)
```

## steps *Implements StEPS and StARS for MGM*

### Description

Calculates the optimal lambda values for the MGM algorithm using StEPS and StARS. Returns a graphSTEPS object that contains the MGMs selected by StEPS and StARS as well as the instability at each value of lambda.

### Usage

```
steps(
  data,
  lambdas = NULL,
  nLambda = 30L,
  gamma = 0.05,
  numSub = 20L,
  subSize = -1L,
  leaveOneOut = FALSE,
```

```
  threads = -1L,
  rank = FALSE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| data | A data.frame containing the dataset to be used for estimating the MGM, with each row representing a sample and each column representing a variable. All continuous variables must be of the numeric type, while categorical variables must be factor or character. Any rows with missing values will be dropped. |
| lambdas | A numeric vector containing the values of lambda to learn an MGM with. The default value is NULL, in which case a log-spaced vector of nLambda values for lambda will be supplied instead. |
| nLambda | A numeric value indicating the number of lambda values to test when the lambdas vector is NULL. The default is 30. |
| gamma | The threshold for edge instability. The default value is 0.05, and it is not recommended to change this value. |
| numSub | The number of subsamples of the dataset used to estimate edge instability. The default value is 20. |
| subSize | The number of samples to be drawn without replacement for each subsample. The default value is -1. When subSize is -1, it is set to min(floor(0.75 * N), floor(10*sqrt(N))), where N is the number of samples. |
| leaveOneOut | If TRUE, performs leave-one-out subsampling. Defaults to FALSE. |
| threads | An integer value denoting the number of threads to use for parallelization of learning MGMs across subsamples. The default value is -1, which will all available CPUs. |
| rank | A logical value indicating whether to use the nonparanormal transform to learn rank-based associations. The default is FALSE. |
| verbose | A logical value indicating whether to print progress updates. The default is FALSE. |

## Value

A graphSTEPS object containing the MGMs selected by StEPS and StARS, as well as the instability of each edge type at each value of lambda.

## Examples

```
sim <- simRandomDAG(200, 25, deg=2)
ig.steps <- steps(sim$data)
print(ig.steps)
```

# Index