Package ‘rCausalMGM’

November 26, 2024

Type Package

Title Scalable Causal Discovery and Model Selection on Mixed Datasets

Version 1.0

Date 2024-04-14

Author Tyler C Lovelace, Panayiotis V Benos

Maintainer Panayiotis V Benos <pbenos@ufl.edu>

Description
The rCasualMGM package is a scalable implementation of CausalMGM, a method for learning causal graphical models over mixed (continuous and discrete) data. rCausalMGM scales to high-dimensional mixed datasets by utilizing a convex score-based method for learning the initial (moralized) graph and a producer-consumer scheme that efficiently performs the conditional independence tests in constraint-based causal discovery algorithms in parallel. Each individual component of the CausalMGM procedure, such as MGM and the causal discovery algorithms PC-Stable and FCI-Stable, are also available individually. Finally, in real-world applications, model selection is essential. As such, rCausalMGM implements three approaches to model selection: (1) information criteria based on model likelihood and complexity, (2) a cross-validation approach to estimating model likelihood on out-of-sample data, and (3) stability-based approaches that assesses how stable a graphical model is across subsamples of the dataset.

License GPL (>= 2)

Imports Rcpp (>= 1.0.3), Rgraphviz, graph

LinkingTo BH, Rcpp, RcppArmadillo, RcppThread

SystemRequirements C++14

RoxygenNote 7.3.2

Encoding UTF-8

NeedsCompilation yes

R topics documented:

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)
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.

Examples

```
## Not run:
sim <- simRandomDAG(200, 25)
g <- pcStable(sim$data)
g.boot <- bootstrap(sim$data, g)
print(g.boot)
print(head(g.boot$stabilities))

## End(Not run)
```

`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)
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

```
## Not run:
sim <- simRandomDAG(200, 25)
g.cv <- fciCV(sim$data)
print(g.cv)

## End(Not run)
```

`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, 25)
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

```
## Not run:
sim <- simRandomDAG(200, 25)
```

```
g.stars <- fciStars(sim$data)
print(g.stars)

## End(Not run)
```

---

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.

---

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)
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 MGM graph on a dataset*

Description

Calculate the MGM graph on a dataset

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)
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

```
## Not run:
sim <- simRandomDAG(200, 25)
ig.cv <- mgmCV(sim$data)
print(ig.cv)

## End(Not run)
```

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

```
## Not run:
sim <- simRandomDAG(200, 25)
g.cv <- mgmfciCV(sim$data)
print(g.cv)

## End(Not run)
```

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

```
## Not run:
sim <- simRandomDAG(200, 25)
ig.path <- mgmPath(sim$data)
print(ig.path)

## End(Not run)
```

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

```
## Not run:
sim <- simRandomDAG(200, 25)
g.cv <- mgmpcCV(sim$data)
print(g.cv)

## End(Not run)
```

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)
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)
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

```
## Not run:
sim <- simRandomDAG(200, 25)
g.cv <- pcCV(sim$data)
print(g.cv)

## End(Not run)
```

`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)
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

```
## Not run:
sim <- simRandomDAG(200, 25)
```

```
g.stars <- pcStars(sim$data)
print(g.stars)

## End(Not run)
```

`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

`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

`plot.graphPath` *A plot override function for the graphCV class*

Description

A plot override function for the graphCV class

Usage

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

Arguments

`x`	The graph object
`...`	Additional plot arguments

`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

`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

`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

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

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

`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

`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

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

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

Examples

```
sim <- simRandomDAG(200, 25)
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)
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)
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)
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

```
data("train_n10000_p10")
sim <- simRandomDAG(200, 25)
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

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)
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,
  discFrac = 0.5,
  deg = 3,
  coefMin = 0.5,
  coefMax = 1.5,
  noiseMin = 1,
  noiseMax = 2,
  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.
`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.
`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)
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.

Index