Package 'sparsebn'

August 29, 2016

Title Learning Sparse Bayesian Networks from High-Dimensional Data

Version 0.0.1

Date 2016-08-12

Maintainer Bryon Aragam <sparsebn@gmail.com>

Description Fast methods for learning sparse Bayesian networks from high-dimensional data using sparse regularization. Designed to incorporate mixed experimental and observational data with thousands of variables with either continuous or discrete observations.

Depends R (>= 3.2.3), sparsebnUtils, ccdrAlgorithm, discretecdAlgorithm

Suggests knitr, rmarkdown, mvtnorm, igraph, graph, testthat

URL https://github.com/itsrainingdata/sparsebn

BugReports https://github.com/itsrainingdata/sparsebn/issues

License GPL (>= 2)

RoxygenNote 5.0.1

VignetteBuilder knitr

LazyData true

NeedsCompilation no

Author Bryon Aragam [aut, cre]

Repository CRAN

Date/Publication 2016-08-19 21:42:53

R topics documented:

cytometryContinuous *The continuous cytometry network*

Description

Data and network for analyzing the flow cytometry experiment from Sachs et al. (2005). This dataset contains the raw measurements from these experiments.

Usage

```
data(cytometryContinuous)
```

Format

A `list` with three components:

- `dag` An `edgeList` containing the consensus network (11 nodes, 17 edges).
- `data` A `data.frame` with 11 variables and 7466 observations.
- `ivn` A `list` specifying which nodes are under intervention in each observation. Compatible with the input to `sparsebnData`.

Examples

```
# Create a valid sparsebnData object from the cytometry data
data(cytometryContinuous)
dat <- sparsebnData(cytometryContinuous$data, type = "c", ivn = cytometryContinuous$ivn)
```

cytometryDiscrete *The discrete cytometry network*

Description

Data and network for analyzing the flow cytometry experiment from Sachs et al. (2005). The data is a cleaned and discretized version of the raw data (see `cytometryContinuous`) from these experiments.

Usage

```
data(cytometryDiscrete)
```

Format

A list with three components:

- dag An edgeList containing the consensus network (11 nodes, 17 edges).
- data A data.frame with 11 variables and 5400 observations.
- ivn A list specifying which nodes are under intervention in each observation. Compatible with the input to sparsebnData.

Examples

```
# Create a valid sparsebnData object from the cytometry data
data(cytometryDiscrete)
dat <- sparsebnData(cytometryDiscrete$data, type = "d", ivn = cytometryDiscrete$ivn)
```

---

estimate.covariance *Covariance estimation*

Description

Methods for inferring (i) Covariance matrices and (ii) Precision matrices for continuous, Gaussian data.

Usage

```
estimate.covariance(data, ...)

estimate.precision(data, ...)
```

Arguments

data	data as sparsebnData object.
...	(optional) additional parameters to estimate.dag

Details

For Gaussian data, the precision matrix corresponds to an undirected graphical model for the distribution. This undirected graph can be tied to the corresponding directed graphical model; see Sections 2.1 and 2.2 (equation (6)) of Aragam and Zhou (2015) for more details.

Value

Solution path as a plain list. Each component is a Matrix corresponding to an estimate of the covariance or precision (inverse covariance) matrix for a given value of lambda.

Examples

```
## Not run:
data(cytometryContinuous)
dat <- sparsebnData(cytometryContinuous$data, type = "d", ivn = cytometryContinuous$ivn)
estimate.covariance(dat) # estimate covariance
estimate.precision(dat)  # estimate precision

## End(Not run)
```

`estimate.dag` *Estimate a DAG from data*

Description

Estimate the structure of a DAG (Bayesian network) from data. Works with any combination of discrete / continuous and observational / experimental data.

Usage

```
estimate.dag(data, lambdas = NULL, lambdas.length = 20, error.tol = 1e-04,
  max.iters = NULL, edge.threshold = NULL, concavity = 2,
  weight.scale = 1, convLb = 0.01, upperbound = 100, verbose = FALSE)
```

Arguments

`data`	Data as `sparsebnData`.
`lambdas`	(optional) Numeric vector containing a grid of lambda values (i.e. regularization parameters) to use in the solution path. If missing, a default grid of values will be used based on a decreasing log-scale (see also generate.lambdas).
`lambdas.length`	Integer number of values to include in the solution path. If `lambdas` has also been specified, this value will be ignored.
`error.tol`	Error tolerance for the algorithm, used to test for convergence.
`max.iters`	Maximum number of iterations for each internal sweep.
`edge.threshold`	Threshold parameter used to terminate the algorithm whenever the number of edges in the current estimate has `> edge.threshold` edges. NOTE: This is not the same as `alpha` in `ccdr.run`.
`concavity`	(CCDr only) Value of concavity parameter. If `gamma > 0`, then the MCP will be used with `gamma` as the concavity parameter. If `gamma < 0`, then the L1 penalty will be used and this value is otherwise ignored.
`weight.scale`	(CD only) A postitive number to scale weight matrix.
`convLb`	(CD only) Small positive number used in Hessian approximation.
`upperbound`	(CD only) A large positive value used to truncate the adaptive weights. A -1 value indicates that there is no truncation.
`verbose`	`TRUE / FALSE` whether or not to print out progress and summary reports.

Details

For details on the underlying methods, see ccdr.run and cd.run.

Value

A sparsebnPath object.

Examples

```
## Not run:
# Estimate a DAG from the cytometry data
data(cytometryContinuous)
dat <- sparsebnData(cytometryContinuous$data, type = "d", ivn = cytometryContinuous$ivn)
estimate.dag(dat)

## End(Not run)
```

---

pathfinder *The pathfinder network*

Description

Simulated data and network for the pathfinder network from the Bayesian network repository. Pathfinder is an expert system developed by Heckerman et. al (1992) to assist with the diagnosis of lymph-node diseases.

Usage

```
data(pathfinder)
```

Format

A list with four components:

- dag An edgeList containing the pathfinder network (109 nodes, 195 edges).
- data A data.frame with 109 variables and 1000 observations.
- ivn A list specifying which nodes are under intervention in each observation; since this dataset is purely observational, this is just NULL. Compatible with the input to sparsebnData.
- cov Covariance matrix used to generate the data.

Details

The data is simulated from a Gaussian SEM assuming unit edge weights and unit variances for all nodes.

Examples

```
## Not run:
# Create a valid sparsebnData object from the simulated pathfinder data
data(pathfinder)
dat <- sparsebnData(pathfinder$data, type = "c")

# If desired, change the edge weights to be randomly generated
coefs <- get.adjacency.matrix(pathfinder$dag)
coefs[coefs != 0] <- runif(n = num.edges(pathfinderDAG), min = 0.5, max = 2)
vars <- Matrix::Diagonal(n = num.nodes(pathfinderDAG), x = rep(1, num.nodes(pathfinderDAG)))
id <- vars
covMat <- t(solve(id - coefs)) %*% vars %*% solve(id - coefs)
pathfinder.data <- rmvnorm(n = 1000, sigma = as.matrix(covMat))

## End(Not run)
```

`sparsebn` *sparsebn: Learning Sparse Bayesian Networks from High-Dimensional Data.*

Description

Methods for learning sparse Bayesian networks and other graphical models from observational and experimental data via sparse regularization. Includes algorithms for both continuous and discrete data.

Details

The main methods for learning graphical models in `sparsebn` are:

- `estimate.dag` for directed acyclic graphs.
- `estimate.precision` for undirected graphs.
- `estimate.covariance` for covariance matrices.

The workhorse behind `sparsebn` is the `sparsebnUtils` package, which provides various S3 classes and methods for representing and manipulating graphs. For more details on this package and the functionality it provides, see `?sparsebnUtils`.

Index