Package: backShift (via r-universe)

August 26, 2024

Type Package

Title Learning Causal Cyclic Graphs from Unknown Shift Interventions

Version 0.1.4.2

Date 2017-09-07

Author Christina Heinze-Deml <heinzedeml@stat.math.ethz.ch>

Depends R (>= 3.1.0)

Imports methods, clue, igraph, matrixcalc, reshape2, ggplot2, mvnmle, MASS

Maintainer Christina Heinze-Deml <heinzedeml@stat.math.ethz.ch>

Description Code for 'backShift', an algorithm to estimate the connectivity matrix of a directed (possibly cyclic) graph with hidden variables. The underlying system is required to be linear and we assume that observations under different shift interventions are available. For more details, see <arXiv:1506.02494>.

License GPL

LazyData true

Suggests knitr, pander, fields, testthat, pcalg, rmarkdown

VignetteBuilder knitr

URL https://github.com/christinaheinze/backShift

BugReports https://github.com/christinaheinze/backShift/issues

RoxygenNote 6.0.1

Repository https://christinaheinze.r-universe.dev

RemoteUrl https://github.com/christinaheinze/backshift

RemoteRef HEAD

RemoteSha 8cc72909850ac8844f58ff07d97c79a853faa8a9

Contents

`backShift` *Estimate connectivity matrix of a directed graph with linear effects and hidden variables.*

Description

This function estimates the connectivity matrix of a directed (possibly cyclic) graph with hidden variables. The underlying system is required to be linear and we assume that observations under different shift interventions are available. More precisely, the function takes as an input an (nxp) data matrix, where n is the sample size and p the number of variables. In each environment j (j in $\{1, \ldots, J\}$) we have observed n_j samples generated from

$$X_j = X_j * A + c_j + e_j$$

(in case of cycles this should be understood as an equilibrium distribution). The c_j is a p-dimensional random vector that is assumed to have a diagonal covariance matrix. The noise vector e_j is assumed to have the same distribution in all environments j but is allowed to have an arbitrary covariance matrix. The different intervention settings are provided to the method with the help of the vector `ExpInd` of length $n = (n_1 + ... + n_j + ... + n_J)$. The goal is to estimate the connectivity matrix A.

Usage

```
backShift(X, ExpInd, covariance=TRUE, ev=0, threshold =0.75, nsim=100,
          sampleSettings=1/sqrt(2), sampleObservations=1/sqrt(2),
          nodewise=TRUE, tolerance = 10^(-4), baseSettingEnv = 1,
          verbose = FALSE)
```

Arguments

`X`	A (nxp)-dimensional matrix (or data frame) with n observations of p variables.
`ExpInd`	Indicator of the experiment or the intervention type an observation belongs to. A numeric vector of length n. Has to contain at least three different unique values.

`covariance`	A boolean variable. If `TRUE`, use only shift in covariance matrix; otherwise use shift in Gram matrix. Set only to `FALSE` if at most one variable has a non-zero shift in mean in the same setting (default is `TRUE`).
`ev`	The expected number of false selections for stability selection. No stability selection computed if `ev=0`. Defaults to `ev=0`.
`threshold`	The selection threshold for stability selection (has to be between 0.5 and 1). Edges which are selected with empirical proportion higher than `threshold` will be retained.
`nsim`	Number of resamples taken (if using stability selection).
`sampleSettings`	The proportion of unique settings to resample for each resample; has to be in [0,1].
`sampleObservations`	The fraction of all samples to retain when subsampling (no replacement); has to be in [0,1].
`nodewise`	If `FALSE`, stability selection retains for each subsample the largest overall entries in the connectivity matrix. If `TRUE`, values are ordered row- and node-wise first and then the largest entries in each row and column are retained. Error control is valid (under exchangeability assumption) in both cases. The latter setting `TRUE` is perhaps more robust and is the default.
`tolerance`	Precision parameter for `ffdiag`: the algorithm stops when the criterium difference between two iterations is less than `tolerance`. Default is 10^(-4).
`baseSettingEnv`	Index for baseline environment against which the intervention variances are measured. Defaults to 1.
`verbose`	If `FALSE`, most messages are supressed.

Value

A list with elements

`Ahat`	The connectivity matrix where entry (i,j) is the effect pointing from variable i to variable j.
`AhatAdjacency`	If `ev`>0, the connectivity matrix retained by stability selection. Entries give the rounded percentage of times the edge has been retained (and 0 if below the critical threshold).
`varianceEnv`	The estimated interventions variances up to an offset. `varianceEnv` is a (Gxp)-dimensional matrix where G is the number of unique environments. The ij-th entry contains the difference between the estimated intervention variance of variable j in environment i and the estimated intervention variance of variable j in the base setting (given by input parameter `baseSettingEnv`).

Author(s)

Christina Heinze-Deml <heinzedeml@stat.math.ethz.ch>

References

Dominik Rothenhaeusler, Christina Heinze, Jonas Peters, Nicolai Meinshausen: backShift: Learning causal cyclic graphs from unknown shift interventions. Advances in Neural Information Processing Systems (NIPS) 28, 2015. arXiv: http://arxiv.org/abs/1506.02494

See Also

ICP and hiddenICP for reconstructing the parents of a variable under interventions on all other variables. getParents and getParentsStable from the package CompareCausalNetworks to estimate the connectivity matrix of a directed causal graph, using various possible methods (including backShift).

Examples

```
## Simulate data with connectivity matrix A

seed <- 1
# sample size n
n <- 10000
# 3 predictor variables
p  <- 3
A <- diag(p)*0
A[1,2] <- 0.8
A[2,3] <- -0.8
A[3,1] <- 0.8

# divide data into 10 different environments
G <- 10

# simulate
simulation.res <- simulateInterventions(
                    n, p, A, G, intervMultiplier = 2,
                    noiseMult = 1, nonGauss = FALSE,
                    fracVarInt = 0.5, hidden = TRUE,
                    knownInterventions = FALSE,
                    simulateObs = TRUE, seed)

environment <- simulation.res$environment
X <- simulation.res$X

## Compute feedback estimator with stability selection

network <- backShift(X, environment, ev = 1)

## Print point estimates and stable edges

# true connectivity matrix
print(A)
# point estimate
print(network$Ahat)
# shows empirical selection probability for stable edges
print(network$AhatAdjacency)
```

`bootstrapBackShift` *Computes a simple model-based bootstrap confidence interval for success of joint diagonalization procedure. The model-based bootstrap approach assumes normally distributed error terms; the parameters of the noise distribution are estimated with maximum likelihood.*

Description

Computes a simple model-based bootstrap confidence interval for success of joint diagonalization procedure. The model-based bootstrap approach assumes normally distributed error terms; the parameters of the noise distribution are estimated with maximum likelihood.

Usage

```
bootstrapBackShift(Ahat, X, ExpInd, nrep, alpha = 0.05, covariance = TRUE,
  baseInd = 1, tolerance = 0.001, verbose = FALSE)
```

Arguments

`Ahat`	Estimated connectivity matrix returned by `backShift`.
`X`	A (nxp)-dimensional matrix (or data frame) with n observations of p variables.
`ExpInd`	Indicator of the experiment or the intervention type an observation belongs to. A numeric vector of length n. Has to contain at least three different unique values.
`nrep`	Number of bootstrap samples.
`alpha`	Significance level for confidence interval.
`covariance`	A boolean variable. If `TRUE`, use only shift in covariance matrix; otherwise use shift in Gram matrix. Set only to `FALSE` if at most one variable has a non-zero shift in mean in the same setting (default is `TRUE`).
`baseInd`	Index for baseline environment against which the intervention variances are measured. Defaults to 1.
`tolerance`	Precision parameter for `ffdiag`: the algorithm stops when the criterium difference between two iterations is less than `tolerance`. Default is 10^(-4).
`verbose`	If `FALSE`, messages are supressed.

Value

A list with the following elements:

- `bootsSumOffDiags` Vector of length `nrep` with sum of off-diagonal elements after joint diagnolization procedure in each of the bootstrap samples.
- `sumOffDiagsBackShift` Sum of off-diagonal elements after joint diagnolization procedure in original estimation.
- `jointDiagSuccess` `TRUE` if `sumOffDiagsBackShift` lies within bootstrap confidence interval.

- `lower` Lower bound of bootstrap confidence interval.
- `upper` Upper bound of bootstrap confidence interval.
- `lowerBasic` `alpha/2` quantile of empirical bootstrap distribution.
- `upperBasic` `1 - alpha/2` quantile of empirical bootstrap distribution.

`computeDiagonalization` *Computes the matrix $\Delta\Sigma\_c, j$ resulting from the joint diagonalization for a given environment (cf. Eq.(7) in the paper). If the joint diagonalization was successful the matrix should be diagonal for all environments j.*

Description

Computes the matrix $\Delta\Sigma_{c,j}$ resulting from the joint diagonalization for a given environment (cf. Eq.(7) in the paper). If the joint diagonalization was successful the matrix should be diagonal for all environments \$j\$.

Usage

```
computeDiagonalization(estConnectivity, X, env, whichEnv, main = NULL)
```

Arguments

`estConnectivity`	Estimate for connectivity matrix returned by `backShift`.
`X`	Data matrix
`env`	Indicator of the experiment or the intervention type an observation belongs to (a numeric vector of length n).
`whichEnv`	Indicator for the environment for which the matrix $\Delta\Sigma_{c,j}$\$ should be computed.
`main`	Optional title for plot; defaults to paste("Env.", whichEnv)

`exampleAdjacencyMatrix` *Example adjacency matrix*

Description

An example for an adjacency matrix A to be used as input to `simulateInterventions`. The entry A_{ij} contains the edge from node i to node j.

Usage

```
data("exampleAdjacencyMatrix")
```

Format

A matrix with 10 rows and 10 columns.

References

Used in simulations in:

Dominik Rothenhaeusler, Christina Heinze, Jonas Peters, Nicolai Meinshausen (2015): backShift: Learning causal cyclic graphs from unknown shift interventions arXiv preprint: http://arxiv.org/abs/1506.02494

Examples

```
data("exampleAdjacencyMatrix")
plotGraphEdgeAttr(estimate = exampleAdjacencyMatrix, plotStabSelec = FALSE,
                  labels = colnames(exampleAdjacencyMatrix),
                  thres.point = 0, thres.stab = NULL, main = "True graph")
```

`generateA` *Generates a connectivity matrix A.*

Description

Generates a connectivity matrix A with cycle product smaller than 1.

Usage

```
generateA(p, expNumNeigh, minCoef, maxCoef, cyclic, verbose = FALSE)
```

Arguments

`p`	Number of variables.
`expNumNeigh`	Expected number of neighbors, to be passed to `randDAG`.
`minCoef`	Minimal edge coefficient. The absolute magnitude of the coefficients will be sampled uniformly at random from the range $[minCoef, maxCoef]$.
`maxCoef`	Maximal edge coefficient. The absolute magnitude of the coefficients will be sampled uniformly at random from the range $[minCoef, maxCoef]$.
`cyclic`	If `TRUE`, connectivity matrix will contain at least one cycle.
`verbose`	If `TRUE`, comments will be printed.

Details

If `expNumNeigh` and `maxCoef` are large, function may fail to find a connectivity matrix with cycle product smaller one. In this case, try to lower these parameters.

Value

A list with two elements

- A Connectivity matrix
- sizeCycle Size of the cycle, if `cyclic` was set to `TRUE`.

`metricsThreshold` *Performance metrics for estimate of connectiviy matrix A.*

Description

Computes various performance metrics for estimate of connectiviy matrix A.

Usage

```
metricsThreshold(trueA, est, thres = seq(0.01, 1, by = 0.01))
```

Arguments

`trueA`	True connectivity matrix
`est`	Estimated connectivity matrix
`thres`	Value at which the point estimate should be thresholded, i.e. edges with coefficients smaller than thres are discarded. Can be a sequence of values.

Value

A data frame with the following columns:

- `Threshold` Value at which point estimate `est` was thresholded.
- `SHD` Structural Hamming distance between `trueA` and `est`.
- `TPR.Recall` True positive rate / recall value
- `FPR` False positive rate
- `Precision` Precision value

Examples

```
# true A
p  <- 3
A <- diag(p)*0
A[1,2] <- 0.8
A[2,3] <- -0.8
A[3,1] <- 0.8

# say an estimated connectivity matrix is given by:
A.est <- matrix(rnorm(p*p, 1e-3, 1e-3), ncol = p)
diag(A.est) <- 0
```

```
A.est[1,2] <- 0.76
A.est[2,3] <- -0.68
A.est[3,1] <- 0.83

# compute metrics with threshold 0.25
metricsThreshold(A, A.est, thres = 0.25)
```

`plotDiagonalization` *Plots the joint diagonalization. I.e. if it was successful the matrices should all be diagonal.*

Description

Plots the joint diagonalization. I.e. if it was successful the matrices should all be diagonal.

Usage

```
plotDiagonalization(estConnectivity, X, env, whichEnv, main = NULL)
```

Arguments

`estConnectivity`	Estimate for connectivity matrix returned by `backShift`.
`X`	Data matrix
`env`	Indicator of the experiment or the intervention type an observation belongs to (a numeric vector of length n).
`whichEnv`	Indicator for the environment to be plotted.
`main`	Optional title for plot; defaults to paste("Env.", whichEnv)

`plotGraphEdgeAttr` *Plotting function to visualize directed graphs*

Description

Given a point estimate of the connectivety matrix or the adjacency matrix, this function visualizes the directed graph using `plot.igraph` from the package `igraph`. If a point estimate is plotted, the edges' intensity reflects the magnitude of the coefficients. If the result is an adjacency matrix estimated by stability selection then the edges' width reflects how often an edge was selected and the intensity reflects the magnitude of the coefficients (if this information is also provided).

Usage

```
plotGraphEdgeAttr(estimate, plotStabSelec, labels, thres.point,
  edgeWeights = NULL, thres.stab = 0.75, main = "", edge.color = "blue",
  ...)
```

Arguments

`estimate`	Estimate of connectivity matrix. This can be a point estimate with entry A_{ij} being the estimated edge weight for the edge from node i to node j. Otherwise, it can be the estimated adjacency matrix by a stability selection procedure as in `backShift`. In this case, the entry A_{ij} indicates how often the edge from node i to node j was selected.
`plotStabSelec`	Set to TRUE if `estimate` results from the stability selection procedure. Otherwise, `estimate` is assumed to be a point estimate.
`labels`	Variable labels to be displayed in plot.
`thres.point`	Value at which the point estimate should be thresholded, i.e. edges with coefficients smaller than `thres.point` are not displayed.
`edgeWeights`	If stability selection result should be visualized, provide edgeWeights as a (pxp)-matrix to display the magnitude of the coefficients as the intensity of the edges.
`thres.stab`	Indicate the threhold value that was used in the stability selection procedure. Used to determine the width of the plotted edges.
`main`	Provide the title of the plot.
`edge.color`	Color of the edges. Defaults to blue.
`...`	Optional arguments passed to the plotting function. Consists of igraph-type options like vertex.label.cex,vertex.label.color, edge.arrow.size or vertex.size etc. @examples # create a matrix A to be visualized p <- 3 A <- diag(p)*0 A[1,2] <- 0.8 A[2,3] <- -0.8 A[3,1] <- 0.8 # add column names to use as labels for nodes colnames(A) <- c("1", "2", "3") # plot plotGraphEdgeAttr(estimate = A, plotStabSelec = FALSE, labels = colnames(A), thres.point = 0, thres.stab = NULL, main = "True graph")

Details

Currently not all options of `igraph` are used; additional arguments are ignored.

`plotInterventionVars` *Plots the estimated intervention variances.*

Description

Plots the estimated intervention variances.

Usage

```
plotInterventionVars(estIntVars, trueIntVars = NULL, scales_facet = "free")
```

Arguments

estIntVars	A (Gxp)-dimensional matrix with the estimated intervention variances returned by backShift (as varianceEnv). G is the number of unique environments, p is the number of variables.
trueIntVars	A (Gxp)-dimensional matrix with the true intervention variances if these are known (for simulations). By default this paramter is set to NULL.
scales_facet	scales argument passed to ggplot's facet_wrap

---

simulateInterventions *Simulate data of a causal cyclic model under shift interventions.*

Description

Simulate data of a causal cyclic model under shift interventions.

Usage

```
simulateInterventions(n, p, A, G, intervMultiplier, noiseMult, nonGauss,
  hiddenVars, knownInterventions, fracVarInt, simulateObs, seed = 1)
```

Arguments

n	Number of observations.
p	Number of variables.
A	Connectivity matrix A. The entry A_{ij} contains the edge from node i to node j.
G	Number of environments, has to be larger than two for backShift.
intervMultiplier	Regulates the strength of the interventions.
noiseMult	Regulates the noise variance.
nonGauss	Set to TRUE to generate non-Gaussian noise.
hiddenVars	Set to TRUE to include hidden variables.
knownInterventions	Set to TRUE if location of interventions should be known.
fracVarInt	If knownInterventions is TRUE, fraction of variables that are intervened on in each environment.
simulateObs	If TRUE, also generate observational data.
seed	Random seed.

Value

A list with the following elements:

- `X` (nxp)-dimensional data matrix
- `environment` Indicator of the experiment or the intervention type an observation belongs to. A numeric vector of length n.
- `interventionVar` (Gxp)-dimensional matrix with intervention variances.
- `interventions` Location of interventions if `knownInterventions` was set to `TRUE`.
- `configs` A list with the following elements:
  - `trueA` True connectivity matrix used to generate the data.
  - `G` Number of environments.
  - `indexObservationalData` Index of observational data
  - `intervMultiplier` Multiplier steering the intervention strength
  - `noiseMult` Multiplier steering the noise level
  - `fracVarInt` If `knownInterventions` was set to `TRUE`, fraction of variables that were intervened on in each environment.
  - `hiddenVars` If `TRUE`, hidden variables exist.
  - `knownInterventions` If `TRUE`, location of interventions is known.
  - `simulateObs` If `TRUE`, environment `1` contains observational data.

References

Dominik Rothenhaeusler, Christina Heinze, Jonas Peters, Nicolai Meinshausen (2015): backShift: Learning causal cyclic graphs from unknown shift interventions. arXiv preprint: http://arxiv.org/abs/1506.02494

Index