Package 'jewel'

March 9, 2021

Title Joint node-wise estimation of Gaussian graphical models from multiple datasets

Version 0.1.0

Author Anna Plaksienko, Claudia Angelini, Daniela De Canditiis

Maintainer Anna Plaksienko <anna@plaxienko.com>

Description jewel is a method for joint estimation of the graph of conditional dependencies between variables given multiple classes of data. Package also allows to estimate regularization parameter with Bayesian information criterion and cross-validation and to generate simulation data. Reference paper to be added.

Depends R (¿= 3.6.0)

Imports Matrix, matrixcalc, MASS, SMUT, igraph, rlist, parallel, purrr

URL https://github.com/annaplaksienko/jewel

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

R topics documented:

`estimateLambdaBIC` *Estimation of the optimal regularization parameter for jewel method with Bayesian information criterion*

Description

Given a grid of regularization parameters, function evaluates Baeysian information criterion (BIC) for each element. Optimal lambda is chosen as the one for which BIC's minimum is obtained. Warm start is implemented.

Usage

```
estimateLambdaBIC(X, lambda, makePlot = TRUE)
```

Arguments

`X`	list of `K` numeric data matrices of size `n_k` by `p` (`n_k` can be different for each matrix)
`lambda`	vector of parameters for which function evaluates BIC
`makePlot`	If makePlot = FALSE, plotting of BIC is disabled. The default value is TRUE.

Value

The following list is returned

- `lambda_opt` - a number, optimal value of regularization parameter according to BIC procedure;
- `BIC` - a vector of BICs for each element of input vector `lambda`.

`estimateLambdaCV` *Estimation of the optimal regularization parameter for jewel method based on cross-validation*

Description

Given a grid of regulariation parameters, function performs cross-validation and estimates the optimal parameter as the one for which the cross-validation error's minimum is obtained. Parallelization over folds and warm start are implemented.

Usage

```
estimateLambdaCV(X, lambda, k_folds = 5, verbose = TRUE, makePlot = TRUE)
```

Arguments

`X`	list of `K` numeric data matrices of size `n_k` by `p` (`n_k` can be different for each matrix)
`lambda`	vector of parameters over which cross-validation is performed
`k_folds`	number of folds in which data is divided. The default value is 5.
`verbose`	If verbose = FALSE, tracing information printing is disabled. The default value is TRUE.
`makePlot`	If makePlot = FALSE, plotting of CV error is disabled. The default value is TRUE.

Value

The following list is returned

- `lambda_opt` - a number, optimal value of regularization parameter according to cross-validation procedure;
- `CV_err` - a vector of cross-validation errors for each element of input vector `lambda`

`evaluatePerformance` *Evaluation of graph estimation methods perfomance if true graph is known*

Description

Function compares adjacency matrices of true and estimated simple graphs and calculates the number of true positives (correctly estimated edges), true negatives (correctly estimated absence of edges), false positives (edges present in the estimator but not in the true graph) and false negatives (failure to identify an edge).

Usage

```
evaluatePerformance(G, G_hat)
```

Arguments

`G`	True graph's adjacency matrix
`G_hat`	Estimated graph's adjacenct matrix. Must have the same dimensions as `G`.

Value

performance - vector of length 4 with TP, TN, FP, FN

`generateData` *Generation of a scale-free graph and corresponding datasets using the graph as their Gaussian graphical model*

Description

Function generates a scale-free graph with `p` vertices and `K` corresponding precision and covariance matrices, all of the size `p` by `p`. Then for each `l`-th element of vector `n` it generates `K` data matrices, each of the size `n_l` by `p`, i.e., for the same underlying graph we can generate several sets of `K` datasets with different sample sizes.

Usage

```
generateData(
  p,
  K,
  n,
  power = 1,
  m = 1,
  a = 0.2,
  b = 0.8,
  makePlot = TRUE,
  verbose = TRUE
)
```

Arguments

`p`	Number of nodes in the true graph
`K`	Number of data matrices
`n`	Vector of the sample sizes for each desired set of `K` data matrices. Can be a vector of one element if one wishes to obtain only one dataset of `K` matrices.
`power`	Power of preferential attachment for Barabasi-Albert algorithm for generation of the scale-free graph. The defaut value is 1.
`m`	Number of edges to add at each time step of Barabasi-Albert algorithm for generation of the scale-free graph. Resulting graph has `mp -(2m -1)` edges. The default value is 1 and graph has `p-1` edges.
`a`	Entries of precision matrices are sampled from the uniform distribution on the interval `[-b,-a] + [a,b]`. The default value is `a = 0.2`.
`b`	Entries of precision matrices are sampled from the uniform distribution on the interval `[-b,-a] + [a,b]`. The default value is `b = 0.8`.
`makePlot`	If makePlot = FALSE, plotting of the generated true graph is disabled. The default value is TRUE.
`verbose`	If verbose = FALSE, tracing information printing is disabled. The default value is TRUE.

Value

The following list is returned

- `trueGraph` - sparse adjacency matrix of the true graph
- `data` - list of lists, for each sample size (`l`-th element of the input vector `n`) one obtains `K` data matrices, each of the size `n_l` by `p`
- `Sigma` - list of `K` covariance matrices of the size `p` by `p`

---

generateLambdaGrid *Generation of the sequence of regularization parameters*

Description

Function generates a uniform in logarithmic space grid of regularization parameters. `Lambda_max` is `max({1 / (n -1)} max(X^TX))`, `lambda_min = lambda_max * eps`.

Usage

```
generateLambdaGrid(X, nlambda = 50, eps = 0.1, scale = TRUE)
```

Arguments

`X`	list of `K` numeric data matrices of size `n_k` by p (`n_k` can be different for each matrix)
`nlambda`	desired number of parameters. The default value is 50.
`eps`	`lambda_min = lambda_max * eps`. The default value is 0.1
`scale`	if TRUE and hence data is scaled, then resulting grid is independent of `X` and goes from `eps` to 1 uniformly in log-scale. The default value is TRUE.

Value

`lambda` - vector of regularization parameters of length `n`

---

jewel *Joint node-wise estimation of Gaussian graphical model from multiple datasets*

Description

Implementation of the jewel method for estimation of the graph of conditional dependencies between the variables given multiple datasets, i.e. when observations of variables are collected under different conditions.

Usage

```
jewel(X, lambda, Theta = NULL, tol = 0.01, maxIter = 10000, verbose = TRUE)
```

Arguments

`X`	List of `K` numeric data matrices of size `n_k` by `p` (`n_k` can be different for each matrix).
`lambda`	Regularization parameter which controls the sparsity of the resulting graph - bigger it is, less edges one gets.
`Theta`	List of `K` starting regression coefficient matrices of size `p` by `p`. If not provided, initialized as all zeros and adjacency matrix is initialized as all ones.
`tol`	Convergence threshold controlling the relative error between iterations. The default value is 0.01.
`maxIter`	Maximum allowed number of iterations. The default value is 10 000.
`verbose`	If verbose = FALSE, tracing information printing is disabled. The default value is TRUE.

Value

The following list is returned

- `EstAdjMat` - adjacency matrix of the estimated graph
- `Theta` - list of `K` estimated covariance matrices of size `p` by `p`
- `residual` - list of `K` matrices of residuals of size `n_k` by `p` (`n_k` can be different for each matrix)
- `BIC` - value of Bayesian information criterion