

Package ‘pulsar’

August 29, 2016

Title Parallel Utilities for Lambda Selection along a Regularization Path

Version 0.2.5

Encoding UTF-8

Description Model selection for penalized graphical models using the Stability Approach to Regularization Selection ('StARS'), with options for speed-ups including Bounded StARS (B-StARS), batch computing, and other stability metrics (e.g., graphlet stability G-StARS).

URL <http://github.com/zdk123/pulsar>, <http://arxiv.org/abs/1605.07072>

BugReports <http://github.com/zdk123/pulsar/issues>

Depends R (>= 3.2.0)

License GPL (>= 2)

Suggests BatchJobs, orca, huge, QUIC, glmnet, network, cluster, testthat, knitr, rmarkdown

Imports methods, parallel, graphics, stats, utils, Matrix

RoxygenNote 5.0.1

VignetteBuilder knitr

NeedsCompilation no

Author Zachary Kurtz [aut, cre],
Christian Müller [aut, ctb]

Maintainer Zachary Kurtz <zdkurtz@gmail.com>

Repository CRAN

Date/Publication 2016-08-04 12:43:42

R topics documented:

pulsar-package	2
batch.pulsar	3
estrada.class	6
gcvec	6
get.opt.index	7

getEnvir	8
getLamPath	8
getMaxCov	9
getTempDir	10
graph.diss	10
natural.connectivity	11
opt.index	12
plot.pulsar	12
print.pulsar	13
print.pulsar.refit	13
pulsar	14
pulsar-function	16
refit	18
update.pulsar	20

Index	22
--------------	-----------

pulsar-package	<i>The pulsar package</i>
----------------	---------------------------

Description

Graphical model selection with the pulsar package

Details

This package provides methods to select a sparse, undirected graphical model by choosing a penalty parameter (λ or λ) among a list of ordered values of λ . We use an implementation of the Stability Approach to Regularization Selection (StARS, see references) inspired by the **huge** package.

However, **pulsar** includes some major differences from other R packages for graphical model estimation and selection (**glasso**, **huge**, **QUIC**, **XMRF**, **clime**, **flare**, etc). The underlying graphical model is computed by passing a function as an argument to **pulsar**. Thus, any algorithm for penalized graphical models can be used in this framework (see **pulsar-function** for more details), including those from the above packages. **pulsar** brings computational experiments under one roof by separating subsampling and calculation of summary criteria from the user-specified core model. The typical workflow in **pulsar** is to perform subsampling first (via the **pulsar**) and then refit the model on the full dataset using **refit**.

Previous StARS implementations can be inefficient for large graphs or when many subsamples are required. **pulsar** can compute upper and lower bounds on the regularization path for the StARS criterion after only 2 subsamples which makes it possible to neglect λ values that are far from the desired StARS regularization parameter, reducing computation time for the rest of the $N - 2$ subsamples (Bounded StARS (B-StARS)).

We also implement additional subsampling-based graph summary criteria which can be used for more informed model selection. For example, we have shown that induced subgraph (graphlet) stability (G-StARS) improves empirical performance over StARS but other criteria are also offered.

Subsampling amounts to running the specified core model for N independent computations. Using the **BatchJobs** framework, we provide a simple wrapper, `batch.pulsar`, for running `pulsar` in embarrassingly parallel mode in an hpc environment. Summary criteria are computed using a Map/Reduce strategy, which lowers memory footprint for large models.

References

Müller, C. L., Bonneau, R. A., & Kurtz, Z. D. (2016). Generalized Stability Approach for Regularized Graphical Models. arXiv: <http://arxiv.org/abs/1605.07072>.

See Also

[pulsar-function](#), [pulsar](#), [batch.pulsar](#)

batch.pulsar	<i>pulsar: batch mode</i>
--------------	---------------------------

Description

Run `pulsar` using stability selection, or another criteria, to select an undirected graphical model over a lambda-path.

Usage

```
batch.pulsar(data, fun = huge::huge, fargs = list(),
  criterion = c("stars"), thresh = 0.1, subsample.ratio = NULL,
  lb.stars = FALSE, ub.stars = FALSE, rep.num = 20, seed = NULL,
  regdir = getTempDir(), regid = basename(regdir), init = "subtwo",
  conffile = ".BatchJobs.R", job.res = list(), progressbars = TRUE,
  cleanup = FALSE)
```

Arguments

<code>data</code>	A $n * p$ matrix of data matrix input to solve for the $p * p$ graphical model
<code>fun</code>	pass in a function that returns a list representing $p * p$ sparse, undirected graphical models along the desired regularization path. The expected inputs to this function are: a data matrix input and a sequence of decreasing lambdas and must return a list or S3 object with a member <i>named</i> path. This should be a list of adjacency matrices for each value of lambda. See pulsar-function for more information.
<code>fargs</code>	arguments to argument fun. Must be a named list and requires at least one member lambda, a numeric vector with values for the penalty parameter.
<code>criterion</code>	A character vector of selection statistics. Multiple criteria can be supplied. Only StARS can be used to automatically select an optimal index for the lambda path. See details for additional statistics.
<code>thresh</code>	threshold (referred to as scalar β in StARS publication) for selection criterion. Only implemented for StARS. <code>thresh=0.1</code> is recommended.

subsample.ratio	determine the size of the subsamples (referred to as $b(n)/n$). Default is $10*\sqrt{n}/n$ for $n > 144$ or 0.8 otherwise. Should be strictly less than 1.
lb.stars	Should the lower bound be computed after the first $N = 2$ subsamples (should result in considerable speedup and only implemented if stars is selected). If this option is selected, other summary metrics will only be applied to the smaller lambda path.
ub.stars	Should the upper bound be computed after the first $N = 2$ subsamples (should result in considerable speedup and only implemented if stars is selected). If this option is selected, other summary metrics will only be applied to the smaller lambda path. This option is ignored if the lb.stars flag is FALSE.
rep.num	number of random subsamples N to take for graph re-estimation. Default is $N = 20$, but more is recommended for non-StARS criteria or if using edge frequencies as confidence scores.
seed	A numeric seed to force predictable subsampling. Default is NULL. Use for testing purposes only.
regdir	directory to store intermediate batch job files
regid	text string representing a unique registry ID for your project
init	text string appended to basename of regdir to store the batch jobs for the initial StARS variability estimate
conffile	path to BatchJobs configuration file
job.res	named list of resources needed for each job (e.g. for intermediate PBS script). The format and members depends on configuration and template. See examples section for a Torque example
progressbars	Flag to show various BatchJobs progress bars. Set FALSE for less output.
cleanup	Flag for removing BatchJob registry files. Recommended FALSE unless you're sure indeterminate data shouldn't be saved.

Details

The options for criterion statistics are:

- stars (Stability approach to regularization selection)
- gcd (Graphlet correlation distance, requires the **orca** package)
- estrada (estrada class) see [estrada.class](#)
- sufficiency (Tandon & Ravikumar's sufficiency statistic)

Value

an S3 object of class `pulsar` with a named member for each stability metric run. Within each of these are:

- `summary`: the summary statistic over `rep.num` graphs at each value of lambda
- `criterion`: the stability criterion used
- `merge`: the raw statistic over the `rep.num` graphs, prior to summarization

- `opt.ind`: index (along the path) of optimal lambda selected by the criterion at the desired threshold. Will return 0 if no optimum is found or NULL if selection for the criterion is not implemented.

If `stars` is included as a criterion then additional arguments include

- `lb.index`: the lambda index of the lower bound at $N = 2$ samples if `lb.stars` flag is set to TRUE
- `ub.index`: the lambda index of the upper bound at $N = 2$ samples if `ub.stars` flag is set to TRUE

`reg`: Registry object. See `BatchJobs::makeRegistry`

`id`: Identifier for mapping graph estimation function. See `BatchJobs::batchMap`

`call`: the original function call

References

Müller, C. L., Bonneau, R., & Kurtz, Z. (2016). Generalized Stability Approach for Regularized Graphical Models. arXiv <http://arxiv.org/abs/1605.07072>

Liu, H., Roeder, K., & Wasserman, L. (2010). Stability approach to regularization selection (stars) for high dimensional graphical models. Proceedings of the Twenty-Third Annual Conference on Neural Information Processing Systems (NIPS).

Zhao, T., Liu, H., Roeder, K., Lafferty, J., & Wasserman, L. (2012). The huge Package for High-dimensional Undirected Graph Estimation in R. The Journal of Machine Learning Research, 13, 1059–1062.

Bischl, B., Lang, M., Mersmann, O., Rahnenführer, J., & Weihs, C. (2015). BatchJobs and Batch-Experiments : Abstraction Mechanisms for Using R in Batch Environments. Journal of Statistical Software, 64(11), 1–25. doi:10.18637/jss.v064.i11

See Also

[pulsar](#)

Examples

```
## Not run:
## Generate the data with huge:
library(huge)
set.seed(10010)
p <- 40 ; n <- 1200
dat <- huge.generator(n, p, "hub", verbose=FALSE, v=.1, u=.3)
lams <- getLamPath(.2, .01, len=40)

## Run batch.pulsar on a Torque cluster

## Get example template and config files
url <- "https://raw.githubusercontent.com/zdk123/pulsar/master/inst/extdata"
download.file(paste(url, "BatchJobsTorque.R", sep="/"), destfile=".BatchJobs.R")
download.file(paste(url, "simpletorque.tml", sep="/"), destfile="simpletorque.tml")
```

```

## Give each job 1gb of memory and a limit of 30 minutes
resources <- list(memory="1GB", nodes="1", walltime="00:30:00")
hugeargs <- list(lambda=lams, verbose=FALSE)
out.p <- batch.pulsar(dat$data, fun=huge::huge, fargs=hugeargs,
                    rep.num=100, criterion=c('stars', 'gcd'),
                    job.res=resources, regdir=file.path(getwd(), "testtorq"))

plot(out.p)

## End(Not run)

```

estrada.class *Estrada class*

Description

Estrada proposes that graphs can be classified into four different classes. We call this the Estrada class. These are: I. Expander-like II. Cluster III. Core-Periphery IV. Mixed.

Usage

```
estrada.class(G, evthresh = 0.001)
```

Arguments

G a $p * p$ adjacency matrix of a Graph
evthresh tolerance for a zero eigenvalue

Value

Estrada class (1 – 4)

References

Estrada, E. (2007). Topological structural classes of complex networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 75(1), 1-12. doi:10.1103/PhysRevE.75.016103

gcvec *Graphlet correlation vector*

Description

Compute graphlet correlations over the desired orbits (default is 11 non-redundant orbits of graphlets of size ≤ 4) for a single graph G

Usage

```
gcvec(G, orbind = c(0, 2, 5, 7, 8, 10, 11, 6, 9, 4, 1) + 1)
```

Arguments

G	a $p * p$ adjacency matrix (dense or sparse) of a graph.
orbind	index vector for which orbits to use for computing pairwise graphlet correlations. Default is from Yaveroğlu et al, 2014 (see References), but 1 offset needed for R-style indexing.

References

- Hočevar, T., & Demšar, J. (2014). A combinatorial approach to graphlet counting. *Bioinformatics* (Oxford, England), 30(4), 559–65. doi:10.1093/bioinformatics/btt717
- Yaveroğlu, Ö. N., Malod-Dognin, N., Davis, D., Levnajic, Z., Janjic, V., Karapandza, R., ... Pržulj, N. (2014). Revealing the hidden language of complex networks. *Scientific Reports*, 4, 4547. doi:10.1038/srep04547

get.opt.index	<i>Get or evaluate an optimal index</i>
---------------	---

Description

If the optimal index for the lambda path is not already assigned, then use a validated method to select the optimal index of the lambda path for alternate criteria (i.e. other than StARS).

Usage

```
get.opt.index(obj, criterion = "gcd", ...)
```

Arguments

obj	the pulsar/batch.pulsar object to evaluate
criterion	a character argument for the desired summary criterion
...	Ignored

Details

Automated optimal index selection is [currently] only implemented for gcd (graphlet stability).

Criterion:

- gcd: Select the minimum gcd summary score within the lower and upper StARS bounds.

Value

index of the lambda path

See Also

[opt.index](#)

getEnvir	<i>Get calling environment</i>
----------	--------------------------------

Description

Generic S3 method for extracting an environment from an S3 object. A getter for an explicitly stored environment from an S3 object or list... probably the environment where the original function that created the object was called from. The default method is a wrapper for `x$envir`.

Usage

```
getEnvir(x)
```

```
## Default S3 method:  
getEnvir(x)
```

Arguments

`x` S3 object to extract the environment

See Also

`getCall`, `environment`, `parent.env`, `eval`

getLamPath	<i>Lambda path</i>
------------	--------------------

Description

Generate a lambda path sequence in descending order, equally or log-spaced.

Usage

```
getLamPath(max, min, len, log = FALSE)
```

Arguments

<code>max</code>	numeric, maximum lambda value
<code>min</code>	numeric, minimum lambda value
<code>len</code>	numeric/int, length of lambda path
<code>log</code>	logical, should the lambda path be log-spaced

Value

numeric vector of lambdas

See Also[getMaxCov](#)**Examples**

```
## Generate the data with huge:
library(huge)
set.seed(10010)
p <- 40 ; n <- 100
dat <- huge.generator(n, p, "hub", verbose=FALSE, v=.1, u=.3)

## Theoretical lamda max is the maximum abs value of the empirical covariance matrix
maxCov <- getMaxCov(dat$data)
lams <- getLamPath(maxCov, 5e-2*maxCov, len=40)
```

`getMaxCov`*Max value of cov*

Description

Get the maximum [absolute] value of a covariance matrix.

Usage

```
getMaxCov(x, cov = isSymmetric(x), abs = TRUE, diag = FALSE)
```

Arguments

<code>x</code>	A matrix/Matrix of data or covariance
<code>cov</code>	Flag if <code>x</code> is a covariance matrix, Set <code>False</code> if <code>x</code> is an <code>n x p</code> data matrix. By default, if <code>x</code> is symmetric, assume it is a covariance matrix.
<code>abs</code>	Flag to get max absolute value
<code>diag</code>	Flag to include diagonal entries in the max

Details

This function is useful to determine the theoretical value for `lambda_max` - for Gaussian data, but may be a useful starting point in the general case as well.

See Also[getLamPath](#)

getTempDir *Generate a string for a temporary directory*

Description

Generate a string to create a random temporary directory, in a platform independent manner. By default, this directory will live under the subdirectory of the per-session temporary directory given by tempdir from base R.

Usage

```
getTempDir(base = tempdir(), len = 6, fsep = .Platform$file.sep)
```

Arguments

base	the base path for the temporary directory.
len	the number of letters to randomly generate the directory name
fsep	the path separator to use (platform dependent)

Details

This function creates a random path intended for temporary directories. E.g. for testing pulsar's batch mode. This function is useful if you need a safe place to store (and delete) files without endangering important directories or R's per session tmp directory, given by tempdir, which may be needed for other uses.

Value

a character vector representing a file path for a randomly generated directory.

See Also

batch.pulsar

graph.diss *Graph dissimilarity*

Description

Dissimilarity matrix of a graph is here defined as the number of neighbors shared by any two nodes.

Usage

```
graph.diss(G, sim = FALSE, loops = FALSE)
```

Arguments

G	a $p * p$ adjacency matrix (dense or sparse) of a graph.
sim	Flag to return Graph similarity instead (1-dissimilarity)
loops	Flag to consider self loops

Value

a $p * p$ dissimilarity matrix

References

Bochkina, N. (2015). Selection of the Regularization Parameter in Graphical Models using a Prior Knowledge of Network Structure, arXiv: 1509.05326.

natural.connectivity *Natural Connectivity*

Description

Compute the natural connectivity of a graph

Usage

```
natural.connectivity(G, eig = NULL, norm = TRUE)
```

Arguments

G	a $p * p$ adjacency matrix (dense or sparse) of a graph. Ignored if eig is given
eig	precomputed list of eigen vals/vectors (output from eigen). If NULL, compute for G.
norm	should the natural connectivity score be normalized

Details

The natural connectivity of a graph is a useful robustness measure of complex networks, corresponding to the average eigenvalue of the adjacency matrix.

Value

numeric natural connectivity score

References

Jun, W., Barahona, M., Yue-Jin, T., & Hong-Zhong, D. (2010). Natural Connectivity of Complex Networks. Chinese Physics Letters, 27(7), 78902. doi:10.1088/0256-307X/27/7/078902

opt.index	<i>Optimal index</i>
-----------	----------------------

Description

Get or set the optimal index of the lambda path, as determined by a given criterion. value must be a numeric/int.

Usage

```
opt.index(obj, criterion = "gcd")

opt.index(obj, criterion = names(value)) <- value
```

Arguments

obj	a pulsar or batch.pulsar object
criterion	a summary statistic criterion for lambda selection. If value is not named, default to gcd.
value	Integer index for optimal lambda by criterion

See Also

[get.opt.index](#)

plot.pulsar	<i>Plot a pulsar S3 object</i>
-------------	--------------------------------

Description

Plot a pulsar S3 object

Usage

```
## S3 method for class 'pulsar'
plot(x, scale = TRUE, invlam = FALSE, loglam = FALSE,
     legends = TRUE, ...)
```

Arguments

x	a pulsar or batch.pulsar object
scale	Flag to scale non-StARS criterion to max StARS value (or 1)
invlam	Flag to plot 1/lambda
loglam	Flag to plot log[lambda]
legends	Flag to plot legends
...	ignored

Details

If both `invlam` and `loglam` are given, `log[1/lambda]` is plotted

`print.pulsar` *Print a pulsar and batch.pulsar S3 object*

Description

Print information about the model, path length, graph dimension, criterion and optimal indices, if defined.

Usage

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

Arguments

`x` a fitted pulsar or batch.pulsar object
`...` ignored

`print.pulsar.refit` *Print a pulsar.refit S3 object*

Description

Print information about the model, path length, graph dimension, criterion and optimal indices and graph sparsity.

Usage

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

Arguments

`x` a pulsar.refit. output from refit
`...` ignored

pulsar *pulsar: serial or parallel mode*

Description

Run pulsar using StARS' edge stability (or other criteria) to select an undirected graphical model over a lambda path.

Usage

```
pulsar(data, fun = huge::huge, fargs = list(), criterion = c("stars"),
       thresh = 0.1, subsample.ratio = NULL, rep.num = 20, seed = NULL,
       lb.stars = FALSE, ub.stars = FALSE, ncores = 1)
```

Arguments

data	A $n * p$ matrix of data matrix input to solve for the $p * p$ graphical model
fun	pass in a function that returns a list representing $p * p$ sparse, undirected graphical models along the desired regularization path. The expected inputs to this function are: a data matrix input and a sequence of decreasing lambdas and must return a list or S3 object with a member <i>named</i> path. This should be a list of adjacency matrices for each value of lambda. See pulsar-function for more information.
fargs	arguments to argument fun. Must be a named list and requires at least one member lambda, a numeric vector with values for the penalty parameter.
criterion	A character vector of selection statistics. Multiple criteria can be supplied. Only StARS can be used to automatically select an optimal index for the lambda path. See details for additional statistics.
thresh	threshold (referred to as scalar β in StARS publication) for selection criterion. Only implemented for StARS. <code>thresh=0.1</code> is recommended.
subsample.ratio	determine the size of the subsamples (referred to as $b(n)/n$). Default is $10 * \sqrt{n}/n$ for $n > 144$ or 0.8 otherwise. Should be strictly less than 1.
rep.num	number of random subsamples N to take for graph re-estimation. Default is $N = 20$, but more is recommended for non-StARS criteria or if using edge frequencies as confidence scores.
seed	A numeric seed to force predictable subsampling. Default is NULL. Use for testing purposes only.
lb.stars	Should the lower bound be computed after the first $N = 2$ subsamples (should result in considerable speedup and only implemented if stars is selected). If this option is selected, other summary metrics will only be applied to the smaller lambda path.

<code>ub.stars</code>	Should the upper bound be computed after the first $N = 2$ subsamples (should result in considerable speedup and only implemented if stars is selected). If this option is selected, other summary metrics will only be applied to the smaller lambda path. This option is ignored if the <code>lb.stars</code> flag is FALSE.
<code>ncores</code>	number of cores to use for subsampling. See <code>batch.pulsar</code> for more parallelization options.

Details

The options for criterion statistics are:

- `stars` (Stability approach to regularization selection)
- `gcd` (Graphlet correlation distance, requires the **orca** package) see [gcvec](#)
- `diss` (Node-node dissimilarity) see [graph.diss](#)
- `estrada` (estrada class) see [estrada.class](#)
- `nc` (natural connectivity) see [natural.connectivity](#)
- `sufficiency` (Tandon & Ravikumar's sufficiency statistic)

Value

an S3 object of class `pulsar` with a named member for each stability metric run. Within each of these are:

- `summary`: the summary statistic over `rep.num` graphs at each value of lambda
- `criterion`: the stability criterion used
- `merge`: the raw statistic over the `rep.num` graphs, prior to summarization
- `opt.ind`: index (along the path) of optimal lambda selected by the criterion at the desired threshold. Will return 0 if no optimum is found or NULL if selection for the criterion is not implemented.

If `stars` is included as a criterion then additional arguments include

- `lb.index`: the lambda index of the lower bound at $N = 2$ samples if `lb.stars` flag is set to TRUE
- `ub.index`: the lambda index of the upper bound at $N = 2$ samples if `ub.stars` flag is set to TRUE

`call`: the original function call

References

- Müller, C. L., Bonneau, R., & Kurtz, Z. (2016). Generalized Stability Approach for Regularized Graphical Models. arXiv. <http://arxiv.org/abs/1605.07072>
- Liu, H., Roeder, K., & Wasserman, L. (2010). Stability approach to regularization selection (stars) for high dimensional graphical models. Proceedings of the Twenty-Third Annual Conference on Neural Information Processing Systems (NIPS).
- Zhao, T., Liu, H., Roeder, K., Lafferty, J., & Wasserman, L. (2012). The huge Package for High-dimensional Undirected Graph Estimation in R. The Journal of Machine Learning Research, 13, 1059–1062.

See Also[batch.pulsar](#)**Examples**

```
## Not run:
## Generate the data with huge:
library(huge)
p <- 40 ; n <- 1200
dat <- huge.generator(n, p, "hub", verbose=FALSE, v=.1, u=.3)
lams <- getLamPath(getMaxCov(dat$data), .01, len=20)

## Run pulsar with huge
hugeargs <- list(lambda=lams, verbose=FALSE)
out.p <- pulsar(dat$data, fun=huge::huge, fargs=hugeargs,
               rep.num=20, criterion='stars')

## Run pulsar in bounded stars mode and include gcd metric:
out.b <- pulsar(dat$data, fun=huge::huge, fargs=hugeargs,
               rep.num=20, criterion=c('stars', 'gcd'),
               lb.stars=TRUE, ub.stars=TRUE)

plot(out.b)

## End(Not run)
```

pulsar-function

Graphical model functions for pulsar

Description

Correctly specify a function for graphical model estimation that is compatible with the pulsar package.

Details

It is easy to construct your own function for penalized model estimation that can be used with this package. The R function must have correctly specified inputs and outputs and is passed into the fun argument to [pulsar](#) or [batch.pulsar](#). Any function that does not follow these rules will fail to give the desired output and may trigger an error.

These packages on CRAN have functions that work out of the box, so you won't need to construct a wrapper:

~function~	~package~
huge	huge
sugm	flare

Inputs:

The function may take arbitrary, named arguments but the first argument must be the data $n * p$ data matrix with the n samples in rows and p features in the columns. At least one argument must be named "lambda", which is expected to be a decreasing numeric vector of penalties. The non-data arguments should be passed into `pulsar` or `batch.pulsar` as a named list (the names must match function arguments exactly) to the `fargs` argument.

Outputs:

The output from the function must be a list or another S3 object inherited from a list. At least one member must be named `path`. This `path` object itself must be a list of $p * p$ adjacency matrices, one for each value of `lambda`. Each cell in the adjacency matrix contains a 1 or TRUE if there is an edge between two nodes or 0/FALSE otherwise. It is highly recommended (though not enforced by `pulsar`) that each adjacency matrix be a column-oriented, compressed, sparse matrix from the `Matrix` package. For example, `dgMatrix/dsMatrix` (general/symmetric numeric Matrix) or the 1-bit `lgMatrix/lscMatrix` classes. The function may return other named outputs, but these will be ignored.

References

Müller, C. L., Bonneau, R. A., & Kurtz, Z. D. (2016). Generalized Stability Approach for Regularized Graphical Models. arXiv: <http://arxiv.org/abs/1605.07072>.

See Also

`pulsar`, `batch.pulsar`, `huge`, `Matrix`

Examples

```
## Generate a hub example
dat <- huge::huge.generator(100, 40, 'hub', verbose=FALSE)

## Simple correlation thresholding
corrthresh <- function(data, lambda) {
  S <- cor(data)
  path <- lapply(lambda, function(lam) {
    tmp <- abs(S) > lam
    diag(tmp) <- FALSE
    as(tmp, 'lsMatrix')
  })
  list(path=path)
}

## Inspect output
lam <- getLamPath(getMaxCov(dat$sigmaHat), 1e-4, 10)
out.cor <- pulsar(dat$data, corrthresh, fargs=list(lambda=lam))
out.cor

## Not run:
## Additional examples
## quic
library(QUIC)
quicr <- function(data, lambda, ...) {
  S <- cov(data)
```

```

est <- QUIC(S, rho=1, path=lambda, msg=0, tol=1e-2, ...)
est$path <- lapply(seq(length(lambda)), function(i) {
  ## convert precision array to adj list
  tmp <- est$X[,i]; diag(tmp) <- 0
  as(tmp!=0, "lgCMatrix")
})
est
}
## clime
library(clime)
climer <- function(data, lambda, tol=1e-5, ...) {
  est <- clime(data, lambda, ...)
  est$path <- lapply(est$Omegalist, function(x) {
    diag(x) <- 0
    as(abs(x) > tol, "dsCMatrix")
  })
  est
}

## inverse cov shrinkage Schafer and Strimmer, 2005
library(corpcor)
icovshrink <- function(data, lambda, tol=1e-3, ...) {
  path <- lapply(lambda, function(lam) {
    tmp <- invcov.shrink(data, lam, verbose=FALSE)
    diag(tmp) <- 0
    as(abs(tmp) > tol, "lsCMatrix")
  })
  list(path=path)
}

## Penalized linear model, only
library(glmnet)
lasso <- function(data, lambda, respind=1, family="gaussian", ...) {
  n <- length(lambda)
  tmp <- glmnet(data[, -respind], data[, respind],
               family=family, lambda=lambda, ...)
  path <- lapply(1:n, function(i) as(tmp$beta[,i, drop=FALSE], "lgCMatrix"))
  list(path=path)
}

## alternative stability selection (DIFFERENT from hdi package)
out <- pulsar(dat$data, lasso, fargs=list(lambda=lam))
mergmat <- do.call('cbind', tmp$stars$merge)
image(mergmat)

## End(Not run)

```

Description

Run the supplied graphical model function on the whole dataset and refit with the selected lambda(s)

Usage

```
refit(obj, criterion)
```

Arguments

obj	a fitted pulsar or batch.pulsar object
criterion	a character vector of criteria for refitting on full data. An optimal index must be defined for each criterion or a message will displayed. If missing (no argument is supplied), try to refit for all pre-specified criteria.

Details

The refit call is evaluated in the environment specified by the pulsar or batch.pulsar object, so if any variables were used for arguments to the original call, unless they are purposefully updated, should not be altered. For example, if the variable for the original data is reassigned, the output of refit will not be on the original dataset.

Value

a pulsar.refit S3 object with members:

- est: the raw output from the graphical model function, fun, applied to the full dataset.
- refit: a named list of adjacency matrices, for each optimal criterion in obj or specified in the criterion argument.
- fun: the original function used to estimate the graphical model along the lambda path.

Examples

```
## Generate the data with huge:
## Not run:
library(huge)
set.seed(10010)
p <- 40 ; n <- 1200
dat <- huge.generator(n, p, "hub", verbose=FALSE, v=.1, u=.3)
lams <- getLamPath(getMaxCov(dat$data), .01, len=20)

## Run pulsar with huge
hugeargs <- list(lambda=lams, verbose=FALSE)
out.p <- pulsar(dat$data, fun=huge::huge, fargs=hugeargs,
               rep.num=20, criterion='stars')

fit <- refit(out.p)

## End(Not run)
```

update.pulsar	<i>Update a pulsar call</i>
---------------	-----------------------------

Description

Update a pulsar model with new or altered arguments. It does this by extracting the call stored in the object, updating the call and (by default) evaluating it in the environment of the original pulsar call.

Usage

```
## S3 method for class 'pulsar'
update(object, ..., evaluate = TRUE)
```

Arguments

object	a n existing pulsar or batch.pulsar object
...	arguments to pulsar to update
evaluate	Flag to evaluate the function. If FALSE, the updated call is returned without evaluation

Details

The update call is evaluated in the environment specified by the pulsar or batch.pulsar object, so if any variables were used for arguments to the original call, unless they are purposefully updated, should not be altered. For example, if the variable for the original data is reassigned, the output of update will not be on the original dataset.

Value

If evaluate = TRUE, the fitted object - the same output as pulsar or batch.pulsar. Otherwise, the updated call.

See Also

eval, [update.pulsar](#), [batch.pulsar](#)

Examples

```
## Not run: p <- 40 ; n <- 1200
dat <- huge.generator(n, p, "hub", verbose=FALSE, v=.1, u=.3)
lams <- getLamPath(getMaxCov(dat$data), .01, len=20)

## Run pulsar with huge
hugeargs <- list(lambda=lams, verbose=FALSE)
out.p <- pulsar(dat$data, fun=huge::huge, fargs=hugeargs,
               rep.num=20, criterion='stars')
```

```
## update call, adding bounds  
out.b <- update(out.p, lb.stars=TRUE, ub.stars=TRUE)  
  
## End(Not run)
```

Index

`batch.pulsar`, [3](#), [3](#), [16](#), [17](#), [20](#)

`estrada.class`, [4](#), [6](#), [15](#)

`gcvec`, [6](#), [15](#)

`get.opt.index`, [7](#), [12](#)

`getEnvir`, [8](#)

`getLamPath`, [8](#), [9](#)

`getMaxCov`, [9](#), [9](#)

`getTempDir`, [10](#)

`graph.diss`, [10](#), [15](#)

`natural.connectivity`, [11](#), [15](#)

`opt.index`, [7](#), [12](#)

`opt.index<- (opt.index)`, [12](#)

`plot.pulsar`, [12](#)

`print.batch.pulsar (print.pulsar)`, [13](#)

`print.pulsar`, [13](#)

`print.pulsar.refit`, [13](#)

`pulsar`, [2](#), [3](#), [5](#), [14](#), [16](#), [17](#), [20](#)

`pulsar-function`, [16](#)

`pulsar-package`, [2](#)

`refit`, [2](#), [18](#)

`update`, [20](#)

`update.pulsar`, [20](#)