Package ‘netprioR’

January 15, 2017

Title A model for network-based prioritisation of genes
Description A model for semi-supervised prioritisation of genes integrating network data, phenotypes and additional prior knowledge about TP and TN gene labels from the literature or experts.
Imports stats, Matrix, dplyr, doParallel, foreach, parallel, sparseMVN, ggplot2, gridExtra, pROC
Depends methods, graphics, R(>= 3.3)
Suggests knitr, BiocStyle, pander
VignetteBuilder knitr
biocViews CellBasedAssays, Preprocessing, Network
Type Package
Lazyload yes
LazyData true
Version 1.0.0
Date 2016-05-08
Author Fabian Schmich
Maintainer Fabian Schmich <fabian.schmich@bsse.ethz.ch>
License GPL-3
URL http://bioconductor.org/packages/netprioR
RoxygenNote 5.0.1
NeedsCompilation no

R topics documented:

  netprioR-package .......................................................... 2
  bandwidth ................................................................. 2
  cmn ................................................................. 3
  conjugate_gradient .................................................... 3
  cuthill_mckee .......................................................... 4
  fit ................................................................. 4
  laplacian ............................................................... 5
  learn ............................................................... 5
  netprioR-class ......................................................... 6
netprioR-package

Package: netprioR

Description

This package provides a model for semi-supervised prioritisation of genes integrating network data, phenotypes and additional prior knowledge about TP and TN gene labels.

Author(s)

Fabian Schmich | Computational Biology Group, ETH Zurich | <fabian.schmich@bsse.ethz.ch>

References

Fabian Schmich et. al (2016).

bandwidth

Description

Compute the bandwidth of a matrix

Usage

bandwidth(x)

Arguments

x Inpute matrix

Value

Bandwidth

Author(s)

Fabian Schmich
**cmn**

*Class Mass Normalization (CMN) from Zhu et al., 2003*

**Description**

Class Mass Normalization (CMN) from Zhu et al., 2003

**Usage**

```
cmn(yhat, l, u)
```

**Arguments**

- `yhat` : Response for labeled (l) and unlabeled (u) genes
- `l` : Indices of labeled genes
- `u` : Indices of unlabeled genes

**Value**

Class normalized yhat

**Author(s)**

Fabian Schmich

---

**conjugate_gradient**  
*Conjugate Gradient Solver*

**Description**

Solves linear equation systems iteratively

**Usage**

```
conjugate_gradient(A, b, x0 = rep(0, ncol(A)), threshold = 1e-15, verbose = FALSE)
```

**Arguments**

- `A` : Matrix
- `b` : Coefficients
- `x0` : Starting solution
- `threshold` : Termination threshold
- `verbose` : Show iterative progress

**Value**

Solution for equation system
Author(s)

Fabian Schmich

cuthill_mckee  
Cuthill McKee (CM) algorithm

Description

Transform sparse matrix into a band matrix

Usage

cuthill_mckee(x)

Arguments

x  
Input matrix

Value

Band matrix

Author(s)

Fabian Schmich

fit  
Fit netprioR model

Description

Fit netprioR model

Usage

fit(object, ...)

## S4 method for signature 'netprioR'
frenchify()

fit(object, refit = FALSE, ...)

Arguments

object  
A netprioR object

...  
Additional arguments

refit  
Flag whether to overwrite existing fit

Value

A netprioR object with fitted model
**laplacian**

*Graph Laplacian*

**Description**

Compute the Laplacian matrix of a graph given its adjacency matrix

**Usage**

```r
laplacian(x, norm = c("none", "sym", "asym"))
```

**Arguments**

- `x` : Adjacency matrix
- `norm` : Type of normalisation

**Value**

Laplacian matrix

**Author(s)**

Fabian Schmich

---

**learn**

*Fit netprioR model*

**Description**

Infer parameters and hidden data using the EM algorithm of netprioR

**Usage**

```r
learn(Yobs, X, G, l, u, a = 0.1, b = 0.1, sigma2 = 1, tau2 = 10, 
esps = 1e-11, max.iter = 500, thresh = 0.001, use.cg = TRUE, 
thresh.cg = 1e-05, nrestarts = 5, max.cores = detectCores(), 
verbose = FALSE)
```
Arguments

- **yobs**: Observed labels (NA, if not observed)
- **X**: Phenotypes
- **G**: Graph Laplacians
- **l**: Indices of labelled instances
- **u**: Indices of unlabelled instances
- **a**: Shape parameter of Gamma prior for W
- **b**: Scale parameter of Gamma prior for W
- **sigma2**: Variance for Gaussian labels
- **tau2**: Variance for Gaussian prior for beta
- **eps**: Small value added to diagonal of Q in order to make it non-singular
- **max.iter**: Maximum number of iterations for EM
- **thresh**: Threshold for termination of EM with respect to change in parameters
- **use.cg**: Flag whether to use conjugate gradient instead of exact computation of expectations
- **thresh.cg**: Threshold for the termination of the conjugate gradient solver
- **nrestarts**: Number of restarts for EM
- **max.cores**: Maximum number of cores to use for parallel computation
- **verbose**: Print verbose output

Value

List containing: Predicted labels Yhat and inferred parameters W and beta

Author(s)

Fabian Schmich

netprioR-class

Description

Class that represents a netprioR model.

Usage

```r
netprioR(networks, phenotypes, labels, ...)
```

```r
## S4 method for signature 'list,matrix,factor'
netprioR(networks, phenotypes, labels,
  fit.model = FALSE, a = 0.1, b = 0.1, sigma2 = 0.1, tau2 = 100,
  eps = 1e-10, max.iter = 500, thresh = 1e-06, use.cg = FALSE,
  thresh.cg = 1e-06, nrestarts = 5, max.cores = detectCores(),
  verbose = TRUE, ...)
```
### netprioR-class

**Arguments**

- **networks**
  - List of N\times N adjacency matrices of gene-gene similarities
- **phenotypes**
  - Matrix of dimension N\times P containing covariates
- **labels**
  - Vector of N\times 1 labels for all genes (NA if no label available)
- **fit.model**
  - Indicator whether to fit the model
- **a**
  - Shape parameter of Gamma prior for W
- **b**
  - Scale parameter of Gamma prior for W
- **sigma2**
  - Variance for Gaussian labels
- **tau2**
  - Variance for Gaussian prior for beta
- **eps**
  - Small value added to diagonal of Q in order to make it non-singular
- **max.iter**
  - Maximum number of iterations for EM
- **thresh**
  - Threshold for termination of EM with respect to change in parameters
- **use.cg**
  - Flag whether to use conjugate gradient instead of exact computation of expectations
- **thresh.cg**
  - Threshold for the termination of the conjugate gradient solver
- **nrestarts**
  - Number of restarts for EM
- **max.cores**
  - Maximum number of cores to use for parallel computation
- **verbose**
  - Print verbose output

**Value**

A netprioR object

**Slots**

- **networks** List of N\times N adjacency matrices of gene-gene similarities
- **phenotypes** Matrix of dimension N\times P containing covariates
- **labels** Vector of N\times 1 labels for all genes. NA if no label available.
- **is.fitted** Flag indicating if model is fitted
- **model** List containing estimated parameters and imputed missing data

**Author(s)**

Fabian Schmich

**Examples**

```r
# runs long-ish
data(simulation)
np <- netprioR(networks = simulation$networks,
               phenotypes = simulation$phenotypes,
               labels = simulation$labels.obs,
               fit.model = TRUE)
summary(np)
```
**norm_kern**

*Normalise kernel*

**Description**

adopted from GeneMania, Mostafavi et al, 2009

**Usage**

`norm_kern(x)`

**Arguments**

- `x` kernel

**Value**

Normalised kernel

**Author(s)**

Fabian Schmich

---

**plot.netprioR**

*Plot method for netprioR objects*

**Description**

Plot method for `netprioR` objects

**Usage**

```
## S3 method for class 'netprioR'
plot(x, which = c("all", "weights", "lik", "scores"), ...)
```

**Arguments**

- `x` A `netprioR` object
- `which` Flag for which plot should be shown, options: weights, lik, scores, all
- `...` Additional parameters for plot

**Value**

Plot of the weights, likelihood, ranks, or all three

**Author(s)**

Fabian Schmich

**Examples**

```
data(simulation)
plot(simulation$model)
```
## ranks

**Description**
Retrieve ranked prioritisation list

**Usage**
ranks(object)

```r
## S4 method for signature 'netprioR'
ranks(object)
```

**Arguments**
- `object` (A `netprioR` object)

**Value**
Ranked list of prioritised genes

**Author(s)**
Fabian Schmich

**Examples**
```r
data(simulation)
ranks(simulation$model)
```

## ROC

**Description**
Compute ROC curve from netprioR model and true labels

**Usage**
ROC(object, ...)

```r
## S4 method for signature 'netprioR'
ROC(object, true.labels, plot = FALSE, ...)
```

**Arguments**
- `object` (A `netprioR` object)
- `...` (Additional arguments)
- `true.labels` (True full set of underlying labels)
- `plot` (Flag whether to plot the AUC curve)
Value

ROC curve with AUC

Author(s)

Fabian Schmich

Examples

```
data(simulation)
ROC(simulation$model, true.labels = simulation$labels.true)
```

---

Description

Simulate labels

Usage

```
simulate_labels(values, sizes, nobs)
```

Arguments

- **values**: Vector of labels for groups
- **sizes**: Vector of group sizes
- **nobs**: Vector of number of observed labels per group

Value

List of Y, Yobs and indices for labeled instances

Author(s)

Fabian Schmich

Examples

```
labels <- simulate_labels(values = c("Positive", "Negative"),
sizes = c(10, 10),
nobs = c(5, 5))
```
**simulate_network_random**

Simulate random networks with predefined number of members for each of the two groups and the number of neighbours for each node

**Description**

Simulate random networks with predefined number of members for each of the two groups and the number of neighbours for each node

**Usage**

```r
simulate_network_random(nmemb, nnei = 1)
```

**Arguments**

- `nmemb`: Vector of number of members for each group
- `nnei`: Number of neighbours for each node

**Value**

Adjacency matrix of graph

**Author(s)**

Fabian Schmich

**Examples**

```r
network <- simulate_network_random(nmemb = c(10, 10), nnei = 1)
```

---

**simulate_network_scalefree**

Simulate scale-free networks

**Description**

Simulate scale-free networks for predefined number of members for each of two groups and a parameter `pclus` that determines how strictly distinct the groups are

**Usage**

```r
simulate_network_scalefree(nmemb, pclus = 1)
```

**Arguments**

- `nmemb`: Vector of numbers of members per group
- `pclus`: Scalar in [0, 1] determining how strictly distinct groups are
Value

Adjacency matrix

Author(s)

Fabian Schmich

Examples

```r
network <- simulate_network_scalefree(nmemb = c(10, 10), pclus = 0.8)
```

---

**simulate_phenotype**  
*Simulate phenotypes correlated to labels pivoted into two groups*

Description

Simulate phenotypes correlated to labels pivoted into two groups

Usage

```r
simulate_phenotype(labels.true, meandiff, sd)
```

Arguments

- `labels.true`: Vector of labels
- `meandiff`: difference of means between positive and negative groups
- `sd`: Standard deviation of the phenotype

Value

Simulated phenotype

Author(s)

Fabian Schmich

Examples

```r
data(simulation)
phenotypes <- simulate_phenotype(labels.true = simulation$labels.true, meandiff = 0.5, sd = 1)
```
Example data: Simulated networks, phenotypes and labels for N = 1000 genes

Description
The data set contains simulated data for N = 1000 genes and P = 1 (univariate) phenotypes. The list of networks contains 2 low noise networks and two high noise networks. The class labels are "Positive" and "Negative".

Usage
data(simulation)

Details
The code used to simulate the data can be found in system.file("example", "data_simulation.R", package = "netprioR")

Value
List of simulated networks, phenotypes and labels for 1000 genes

Retrieve network weights

Description
Retrieve network weights

Usage
weights(object, ...)

## S4 method for signature 'netprioR'
weights(object)

Arguments
object A netprioR object
...
Additional arguments

Value
Estimated network weights

Author(s)
Fabian Schmich
Examples

data(simulation)
weights(simulation$model)
Index

*Topic package

  netprioR-package, 2

bandwidth, 2

cmn, 3
conjugate_gradient, 3
cuthill_mckee, 4

fit, 4
fit,netprioR-method (fit), 4

laplacian, 5
learn, 5

netprioR, 4, 7–9, 13
netprioR (netprioR-class), 6
netprioR,list,matrix,factor-method (netprioR-class), 6
netprioR-class, 6
netprioR-package, 2

norm_kern, 8

plot.netprioR, 8

ranks, 9
ranks,netprioR-method (ranks), 9
ROC, 9
ROC,netprioR-method (ROC), 9

simulate_labels, 10
simulate_network_random, 11
simulate_network_scalefree, 11
simulate_phenotype, 12
simulation, 13

weights, 13
weights,netprioR-method (weights), 13