Package ‘gasper’

August 3, 2020

Type Package

Title Graph Signal Processing

Version 1.0.1

Description Provides the standard operations for signal processing on graphs:
graph Fourier transform, spectral graph wavelet transform,
visualization tools. It also implements a data driven method
for graph signal denoising/regression, for details see
De Loynes, Navarro, Olivier (2019) <arxiv:1906.01882>.

License LGPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports scatterplot3d, rwavelet, Rcpp, ggplot2

LinkingTo Rcpp, RcppArmadillo

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation yes

Author Fabien Navarro [aut, cre],
Basile De Loynes [aut],
Baptiste Olivier [aut]

Maintainer Fabien Navarro <fabien.navarro@math.cnrs.fr>

Repository CRAN

Date/Publication 2020-08-03 08:40:02 UTC

R topics documented:

`adjacency_mat` *Compute the adjacency matrix of the gaussian weighted graph*

Description

Compute the adjacency matrix of the gaussian weighted graph

Usage

```
adjacency_mat(pts, f = function(x) {     exp(-x^2/8) }, s = 0)
```

Arguments

`pts`	coordinates of N points in R^3.
`f`	is a scalar potential ($\exp(-x^2/2t^2)$ for gaussian potential)
`s`	is a threhold to sparisfy the matrix

See Also

`laplacian_mat`

Examples

```
pts <- swissroll(N=100, seed=0, a=1, b=4)
W <- adjacency_mat(pts)
```

`analysis` *Analysis operator.*

Description

Compute the analysis operator for coefficient y.

Usage

```
analysis(y, tf)
```

Arguments

`y`	Graph signal to analyze.
`tf`	frame coefficients.

Value

`coef` Transform coefficients.

`betathresh` *Apply Beta Threshold.*

Description

Apply Beta Threshold.

Usage

```
betathresh(y, t, b)
```

Arguments

`y`	Noisy Data.
`t`	Threshold.
`b`	Thresholding type (b=1: soft, b=2: JS).

Value

`x` Filtered result.

download_graph *Download sparse matrix form the SuiteSparse Matrix Collection.*

Description

If coordinates are associated with the graphs, they are automatically downloaded and added to the output. See https://sparse.tamu.edu/ for the list of groups and graph names.

Usage

```
download_graph(graphname, groupname)
```

Arguments

graphname	Name of the graph to download.
groupname	Name of the group that provides the graph.

Value

graphname a list of dataframe contening W and xy coordinates.

Examples

```
graphname <- "grid1"
groupname <- "AG-Monien"
download_graph(graphname,groupname)
plot_graph(grid1)
```

eigendec *Spectral decomposition of a symetric matrix*

Description

Eigen decomposition of dense symmetric/hermitian matrix M using standard or divide-and-conquer methods. the divide-and-conquer method provides slightly different results than the standard method, but is considerably faster for large matrices

Usage

```
eigendec(M)
```

Arguments

M	a matrix. //@param method "dc" indicates divide-and-conquer method (par). // "std" indicates standard method.

`eigensort` *Spectral decomposition of a symetric matrix.*

Description

Computes eigenvalues and eigenvectors of matrices (output sorted in increasing order).

Usage

```
eigensort(x)
```

Arguments

`x`	Symetric matrix whose spectral decomposition is to be computed.

Examples

```
A <- matrix(1, ncol=2, nrow=2)
dec <- eigensort(A)
```

`full` *Convert symmetric sparse matrix to full matrix.*

Description

Convert a symmetric sparse matrix sA to full matrix A.

Usage

```
full(sA)
```

Arguments

`sA`	Sparse matrix to convert.

Examples

```
sA <- pittsburgh$sA
A <- full(sA)
```

`fullup` *Convert symmetric sparse matrix to full matrix.*

Description

Convert a symmetric sparse matrix sA stored as upper triangular matrix to full matrix A.

Usage

```
fullup(sA)
```

Arguments

`sA`	Sparse upper triangular matrix matrix to convert.

Examples

```
graphname <- "grid1"
groupname <- "AG-Monien"
download_graph(graphname,groupname)
A <- fullup(grid1$sA)
```

`GVN` *Graph Von Neumann Estimator.*

Description

Graph equivalent of the Von Neummann variance estimator.

Usage

```
GVN(y, A, L)
```

Arguments

`y`	Noisy data.
`A`	Adjacency matrix.
`L`	Laplacian matrix.

References

von Neumann, J. (1941). Distribution of the ratio of the mean square successive difference to the variance. *Ann. Math. Statistics*, 35(3), 433–451.

de Loynes, B., Navarro, F., Olivier, B. (2019). Data-driven Thresholding in Denoising with Spectral Graph Wavelet Transform. arXiv preprint arXiv:1906.01882.

Examples

```
data(minnesota)
A <- minnesota$A
L <- laplacian_mat(A)
x <- minnesota$xy[ ,1]
n <- length(x)
f <- sin(x)
sigma <- 0.1
noise <- rnorm(n, sd = sigma)
y <- f + noise
sigma^2
GVN(y, A, L)
```

---

laplacian_mat — *Laplacian matrix.*

Description

Compute the (unormalized) laplacian matrix from the adjacency matrix.

Usage

```
laplacian_mat(W)
```

Arguments

W	Adjacency matrix.

---

matmult — *Matrix multiplication*

Description

Matrix multiplication

Usage

```
matmult(A, B)
```

Arguments

A	a matrix.
B	a matrix.

minnesota *Minnesota road network.*

Description

A dataset containing the minnesota road network (as well as the two signals).

Usage

```
minnesota
```

Format

list of 6 elements

xy coordinates

A adjacency matrix

sA sparse version of A

f1 eta=0.01 and k=2

f2 eta = 0.001 and k=4

labels labels

Source

D. Gleich. The MatlabBGL Matlab library. https://www.cs.purdue.edu/homes/dgleich/packages/matlab_bgl/index.html.

NYCdata *NYC network.*

Description

A dataset containing the NYC network.

Usage

```
NYCdata
```

Format

list of 2 elements

A NYC adjacency matrix (built using exponential weights between two nodes).

f median price observed from miles travelled to the given drop off point.

Source

https://s3.amazonaws.com/nyc-tlc/trip+data/yellow_tripdata_2018-01.csv.

`pittsburgh` *Pittsburgh network.*

Description

A dataset containing the pittsubrgh network.

Usage

```
pittsburgh
```

Format

list of 7 elements

A pittsburgh adjacency matrix

sA pittsburgh sparse adjacency matrix

xy coordinates

f signal used in Trend filtering on graphs

y noisy signal used in Trend filtering on graph

f1 eta=0.01, k=5

geo geometry

Source

The sources come from different codes provided by Yu-Xiang Wang (UC Santa Barbara) and are associated with the article: "Trend Filtering on Graphs, JMLR, 2016". https://sites.cs.ucsb.edu/~yuxiangw/resources.html.

`plot_graph` *Graph plot*

Description

Graph plot

Usage

```
plot_graph(z, size = 0.75)
```

Arguments

z	Graph data.
size	Dot size.

Examples

```
plot_graph(minnesota)
```

`plot_signal` *Plot a signal on top of a given graph*

Description

Plot a signal on top of a given graph

Usage

```
plot_signal(z, f, size = 0.75, limits = range(f))
```

Arguments

`z`	Graph data.
`f`	Signal to plot.
`size`	Dot size.
`limits`	Set colormap limits.

Examples

```
f <- rnorm(2642)
plot_signal(minnesota, f)
```

`randsignal` *Generate random signal with varying regularity.*

Description

Generate f = A^k * x_eta / lambda_1^k, with A the adjacency matrix and x_eta realization of Bernoulli random variables of parameter eta.

Usage

```
randsignal(eta, k, A)
```

Arguments

`eta`	Smoothness parameter.
`k`	Smoothness parameter.
`A`	Adjacency matrix.

Value

`f` output signal.

`rlogo` *R logo graph.*

Description

A dataset containing a graph based on the R logo.

Usage

```
rlogo
```

Format

list of 2 elements

xy coordinates

sA adjacency matrix

`smoothmodulus` *Modulus of smoothness.*

Description

Compute the modulus of smoothness of a graph signal.

Usage

```
smoothmodulus(f, A, L)
```

Arguments

`f`	Signal.
`A`	Adjacency matrix.
`L`	Laplacian matrix.

Examples

```
data(minnesota)
A <- minnesota$A
L <- laplacian_mat(A)
x <- minnesota$xy[ ,1]
f <- sin(x)
smoothmodulus(f, A, L)
```

SUREthresh *Stein's Unbiased Risk Estimate.*

Description

Adaptive Threshold Selection Using Principle of SURE (The irreductible variance term is not included, it does not change the position of the minimum).

Usage

```
SUREthresh(wcn, tresh, diagWWt, b, sigma, hatsigma, policy)
```

Arguments

`wcn`	Noisy wavelet coefficents.
`tresh`	Threshold values.
`diagWWt`	Weights.
`b`	Thresholding type (b=1: soft, b=2: JS).
`sigma`	Sd of the noise.
`hatsigma`	Estimator of the sd (if any).
`policy`	Dependent or uniform.

Value

`res` a dataframe contening SURE, hatSURE and their respective minima.

References

de Loynes, B., Navarro, F., Olivier, B. (2019). Data-driven Thresholding in Denoising with Spectral Graph Wavelet Transform. arXiv preprint arXiv:1906.01882.

SURE_MSEthresh *Stein's Unbiased Risk Estimate.*

Description

Adaptive Threshold Selection Using Principle of SURE (The irreductible variance term is not included, it does not change the position of the minimum).

Usage

```
SURE_MSEthresh(wcn, wcf, tresh, diagWWt, b, sigma, hatsigma, policy)
```

Arguments

`wcn`	Noisy wavelet coefficents.
`wcf`	True wavelet coefficients.
`tresh`	Threshold values.
`diagWWt`	Weights.
`b`	Thresholding type (b=1: soft, b=2: JS).
`sigma`	Sd of the noise.
`hatsigma`	Estimator of the sd (if any).
`policy`	Dependent or uniform.

Details

Note: - the calculation of the MSE is also included for comparison purpose.

Value

`res` a dataframe contening MSE, SURE, hatSURE and their respective minima

References

de Loynes, B., Navarro, F., Olivier, B. (2019). Data-driven Thresholding in Denoising with Spectral Graph Wavelet Transform. arXiv preprint arXiv:1906.01882.

`swissroll` *Swiss roll graph generation*

Description

Map the square $[0, 1]^2$ in swiss roll for all x, y in $[0, 1]^2$, set

$$Sx = \pi\sqrt{(b^2 - a^2)x + a^2)}$$

$$Sy = \pi(b^2 - a^2)y/2$$

Usage

```
swissroll(N = 500, seed = NULL, a = 1, b = 4)
```

Arguments

`N`	Number of points drawn.
`seed`	Optionally specify a RNG seed (for reproducible experiments).
`a, b`	Shape parameters.

Value

N x 3 array for 3d points.

See Also

adjacency_mat

Examples

```
pts <- swissroll(N=500, seed=0, a=1, b=4)
scatterplot3d::scatterplot3d(pts, y=NULL, z=NULL)
```

---

synthesis *Synthesis operator.*

Description

Compute the synthesis operator for coefficient coeff.

Usage

```
synthesis(coeff, tf)
```

Arguments

coeff	Transform coefficients.
tf	Frame coefficients.

Value

y synthesis signal.

---

tight_frame *Tight frame computation.*

Description

Construct tight (Parseval) frame.

Usage

```
tight_frame(evalues, evectors, b = 2)
```

Arguments

`evalues`	Eigenvalues of the Laplacian matrix.
`evectors`	Eigenvectors of the Laplacian matrix.
`b`	Parameter that control the number of scales.

References

Göbel, F., Blanchard, G., von Luxburg, U. (2018). Construction of tight frames on graphs and application to denoising. In Handbook of Big Data Analytics (pp. 503-522). Springer, Cham.

de Loynes, B., Navarro, F., Olivier, B. (2019). Data-driven Thresholding in Denoising with Spectral Graph Wavelet Transform. arXiv preprint arXiv:1906.01882.

Index