Package ‘GRAPE’

January 20, 2025

Type Package

Title Gene-Ranking Analysis of Pathway Expression

Version 0.1.1

Author Michael Klein <michael.klein@yale.edu>

Maintainer Michael Klein <michael.klein@yale.edu>

Imports stats

Description Gene-Ranking Analysis of Pathway Expression (GRAPE) is a tool for summarizing the consensus behavior of biological pathways in the form of a template, and for quantifying the extent to which individual samples deviate from the template. GRAPE templates are based only on the relative rankings of the genes within the pathway and can be used for classification of tissue types or disease subtypes. GRAPE can be used to represent gene-expression samples as vectors of pathway scores, where each pathway score indicates the departure from a given collection of reference samples. The resulting pathway-space representation can be used as the feature set for various applications, including survival analysis and drug-response prediction.
Users of GRAPE should use the following citation:
Klein MI, Stern DF, and Zhao H. GRAPE: A pathway template method to characterize tissue-specific functionality from gene expression profiles.
BMC Bioinformatics, 18:317 (June 2017).

License GPL-2

LazyData TRUE

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-07 22:12:24 UTC

Contents

`getPathwayScores` *Calculate Pathway Scores*

Description

Calculate pathway scores of a single pathway of a set of samples relative to a reference set of samples

Usage

```
getPathwayScores(refmat, newmat, w = w_quad)
```

Arguments

`refmat`	Pathway expression matrix of reference samples. Rows are genes, columns are samples.
`newmat`	Pathway expression matrix of new samples. Rows are genes, columns are samples.
`w`	Weight function. Default is quadratic weight function.

Value

Vector of pathway scores of each sample in newmat.

Examples

```
## Toy example: 50 reference samples
set.seed(10);
refmat <- matrix(rnorm(5*50),nrow=5,ncol=50); rownames(refmat) <- paste0("g",1:5)
### make g2 and g5 larger in refmat
refmat[2,] <- rnorm(50,3,2); refmat[5,] <- rnorm(50,4,4)
### 15 new samples
newmat <- matrix(rnorm(5*15),nrow=5,ncol=15); rownames(newmat) <- paste0("g",1:5)
### make g2 and g3 larger in newmat
newmat[2,] <- rnorm(15,2,3); newmat[3,] <- rnorm(15,4,3)
ps_new <- getPathwayScores(refmat,newmat) ### get pathway scores of new samples
ps_ref <- getPathwayScores(refmat,refmat) ### get pathway scores of reference samples
ps_both <- getPathwayScores(refmat,cbind(refmat,newmat)) ### get pathway scores of both
# > ps_new
# [1]  6.2720  8.5696  9.9904  6.9056  3.7824  8.9344 13.0880 10.2912  3.7824
# 0.0384 13.1136  6.8032  4.8512 12.8512 10.2912
```

`makeBinaryTemplateAndProbabilityTemplate`
Make binary template and probability template

Description

Takes in matrix, where columns are samples and rows are pathway genes, outputs the binary and probability templates

Usage

```
makeBinaryTemplateAndProbabilityTemplate(submat)
```

Arguments

`submat`	A matrix where columns are samples and rows are pathway genes

Value

List containing binary template vector and probability template vector

Examples

```
submat <- cbind(c(1,3,2,1.5),c(2,3,1.5,1.2),c(1.4,4.2,3.5,3.8))
rownames(submat) <- c("gene_A","gene_B","gene_C","gene_D")
temp <- makeBinaryTemplateAndProbabilityTemplate(submat)
bt <- temp$binary_template; pt <- temp$probability_template
cbind(bt,pt)
```

`makeGRAPE_psMat` *Calculate Pathway Space Matrix*

Description

Represents new samples as vectors of pathway scores relative to reference samples

Usage

```
makeGRAPE_psMat(refge, newge, pathway_list, w = w_quad)
```

Arguments

`refge`	Gene expression matrix of reference samples. Rows are genes, columns are samples.
`newge`	Gene expression matrix of new samples. Rows are genes, columns are samples.
`pathway_list`	List of pathways. Each pathway is a character vector consisting of gene names.
`w`	Weight function. Default is quadratic weight function.

Value

Vector of pathway scores of each sample in newmat.

Examples

```
#' ### Make pathway scores mat
set.seed(10)
### 50 reference samples
refge <- matrix(rnorm(10*50),nrow=10,ncol=50); rownames(refge) <- paste0("g",1:10)
refge[c(2,5,8),] <- matrix(rnorm(3*50,mean=2,sd=2))
refge[c(3,4,7),] <- matrix(rnorm(3*50,mean=4,sd=4))
### 6 new samples
newge <- matrix(rnorm(10*6),nrow=10,ncol=6); rownames(newge) <- paste0("g",1:10)
newge[c(2:7),] <- matrix(rnorm(6*6,mean=3,sd=1))
newge[c(1,9),] <- matrix(rnorm(2*6,mean=5,sd=3))
pathway_list <- list(set1=paste0("g",1:4),set2=paste0("g",5:10),set3=paste0("g",c(1,4,8:10)))
psmat <- makeGRAPE_psMat(refge,newge,pathway_list)
# > psmat
# [,1]     [,2]     [,3]     [,4]     [,5]     [,6]
# set1 2.397426 1.406275 2.516492 2.358809 2.555109 2.358809
# set2 0.670354 3.245575 3.962389 2.670354 1.741150 1.579646
# set3 1.536017 2.167373 2.167373 2.167373 2.148305 1.809322
```

---

makePairwiseOrder *Make pairwise order representation of a sample*

Description

Takes in a vector of gene expression values and returns a binary vector consisting of the pairwise rankings for the sample

Usage

```
makePairwiseOrder(samp)
```

Arguments

samp	A vector of gene expression values

Value

Binary vector of the pairwise ranking representation of the samples

Examples

```
samp <- c(1,3,2,1.5)
makePairwiseOrder(samp)
```

`makePairwiseOrderNames` *Make template names from gene names*

Description

Takes in vector of pathway gene names, returns names corresponding to the pairwise binary representation

Usage

```
makePairwiseOrderNames(path_genes)
```

Arguments

`path_genes`	A vector of pathway gene names

Value

Names for the pairwise representation, of the form "gA<gB"

Examples

```
path_genes <- c("gene_A","gene_B","gene_C","gene_D")
makePairwiseOrderNames(path_genes)
```

`predictClassDIRAC` *DIRAC Classification*

Description

Classification of a samples according to dirac distances from templates. Usually applied to the gene expression values for a single pathway.

Usage

```
predictClassDIRAC(trainmat, testmat, train_labels)
```

Arguments

`trainmat`	Matrix of gene expression for set of genes accross training set samples. Each column is a sample.
`testmat`	Matrix of gene expression for set of genes accross test set samples. Each column is a sample.
`train_labels`	Vector of class labels for each sample in the training set.

Value

Predicted class labels for test set

Examples

```
# Toy example of two classes
set.seed(10); path_genes <- c("gA","gB","gC","gD"); nsamps <- 50 # Four genes, 50 samples per class
class_one_samps <- matrix(NA,nrow=length(path_genes),ncol=nsamps) # Class 1
rownames(class_one_samps) <- path_genes
class_one_samps[1,] <- rnorm(ncol(class_one_samps),4,2)
class_one_samps[2,] <- rnorm(ncol(class_one_samps),5,4)
class_one_samps[3,] <- rnorm(ncol(class_one_samps),1,1)
class_one_samps[4,] <- rnorm(ncol(class_one_samps),2,1)
class_two_samps <- matrix(NA,nrow=length(path_genes),ncol=nsamps) # Class 2
rownames(class_two_samps) <- path_genes
class_two_samps[1,] <- rnorm(ncol(class_two_samps),2,3)
class_two_samps[2,] <- rnorm(ncol(class_two_samps),5,2)
class_two_samps[3,] <- rnorm(ncol(class_two_samps),1,1)
class_two_samps[4,] <- rnorm(ncol(class_two_samps),0,1)
all_samps <- cbind(class_one_samps,class_two_samps)
labels <- c(rep(1,nsamps),rep(2,nsamps))
testid <- sample.int(100,20)
trainmat <- all_samps[,-testid]
train_labels <- labels[-testid]
testmat <- all_samps[,testid]
test_labels <- labels[testid]
yhat <- predictClassDIRAC(trainmat,testmat,train_labels)
sum(diag(table(test_labels,yhat)))/length(test_labels) # accuracy
# [1] 0.7
```

`predictClassGRAPE` *GRAPE Classification*

Description

Classification of a samples according to grape distances from templates. Usually applied to the gene expression values for a single pathway.

Usage

```
predictClassGRAPE(trainmat, testmat, train_labels, w = w_quad)
```

Arguments

`trainmat`	Matrix of gene expression for set of genes accross training set samples. Each column is a sample.
`testmat`	Matrix of gene expression for set of genes accross test set samples. Each column is a sample.
`train_labels`	Vector of class labels for each sample in the training set.
`w`	Weight function. Default is quadratic weight function.

Value

Predicted class labels for test set

Examples

```
# Toy example of two classes
set.seed(10); path_genes <- c("gA","gB","gC","gD"); nsamps <- 50 # Four genes, 50 samples per class
class_one_samps <- matrix(NA,nrow=length(path_genes),ncol=nsamps) # Class 1
rownames(class_one_samps) <- path_genes
class_one_samps[1,] <- rnorm(ncol(class_one_samps),4,2)
class_one_samps[2,] <- rnorm(ncol(class_one_samps),5,4)
class_one_samps[3,] <- rnorm(ncol(class_one_samps),1,1)
class_one_samps[4,] <- rnorm(ncol(class_one_samps),2,1)
class_two_samps <- matrix(NA,nrow=length(path_genes),ncol=nsamps) # Class 2
rownames(class_two_samps) <- path_genes
class_two_samps[1,] <- rnorm(ncol(class_two_samps),2,3)
class_two_samps[2,] <- rnorm(ncol(class_two_samps),5,2)
class_two_samps[3,] <- rnorm(ncol(class_two_samps),1,1)
class_two_samps[4,] <- rnorm(ncol(class_two_samps),0,1)
all_samps <- cbind(class_one_samps,class_two_samps)
labels <- c(rep(1,nsamps),rep(2,nsamps))
testid <- sample.int(100,20)
trainmat <- all_samps[,-testid]
train_labels <- labels[-testid]
testmat <- all_samps[,testid]
test_labels <- labels[testid]
yhat <- predictClassGRAPE(trainmat,testmat,train_labels,w_quad)
sum(diag(table(test_labels,yhat)))/length(test_labels) # accuracy
# [1] 0.8
```

---

predictClassPC *PC Classification*

Description

Classification of a samples according to euclidean distances from PC templates. Usually applied to the gene expression values for a single pathway.

Usage

```
predictClassPC(trainmat, testmat, train_labels)
```

Arguments

`trainmat`	Matrix of gene expression for set of genes accross training set samples. Each column is a sample.
`testmat`	Matrix of gene expression for set of genes accross test set samples. Each column is a sample.
`train_labels`	Vector of class labels for each sample in the training set.

Value

Predicted class labels for test set

Examples

```
# Toy example of two classes
set.seed(10); path_genes <- c("gA","gB","gC","gD"); nsamps <- 50 # Four genes, 50 samples per class
class_one_samps <- matrix(NA,nrow=length(path_genes),ncol=nsamps) # Class 1
rownames(class_one_samps) <- path_genes
class_one_samps[1,] <- rnorm(ncol(class_one_samps),4,2)
class_one_samps[2,] <- rnorm(ncol(class_one_samps),5,4)
class_one_samps[3,] <- rnorm(ncol(class_one_samps),1,1)
class_one_samps[4,] <- rnorm(ncol(class_one_samps),2,1)
class_two_samps <- matrix(NA,nrow=length(path_genes),ncol=nsamps) # Class 2
rownames(class_two_samps) <- path_genes
class_two_samps[1,] <- rnorm(ncol(class_two_samps),2,3)
class_two_samps[2,] <- rnorm(ncol(class_two_samps),5,2)
class_two_samps[3,] <- rnorm(ncol(class_two_samps),1,1)
class_two_samps[4,] <- rnorm(ncol(class_two_samps),0,1)
all_samps <- cbind(class_one_samps,class_two_samps)
labels <- c(rep(1,nsamps),rep(2,nsamps))
testid <- sample.int(100,20)
trainmat <- all_samps[,-testid]
train_labels <- labels[-testid]
testmat <- all_samps[,testid]
test_labels <- labels[testid]
yhat <- predictClassPC(trainmat,testmat,train_labels)
sum(diag(table(test_labels,yhat)))/length(test_labels) # accuracy
# [1] 0.55
```

---

w_quad *Quadratic weight function*

Description

Calculates the weights of all input entries. All entries should take values in [0,1].

Usage

```
w_quad(x)
```

Arguments

x	Any number, vector of matrix.

Value

Weight of each element

Examples

```
w_quad(0.95)
w_quad(cbind(c(.7,.8),c(.9,.1)))
```

Index