Package 'DiPALM'

November 4, 2020

Type Package

Title Differential Pattern Analysis via Linear Modeling

Version 1.1

Date 2020-11-03

Author Ryan C. Sartor <sartorry@gmail.com>, Kathleen Greenham <greenham@umn.edu>

Maintainer Ryan C. Sartor <sartorry@gmail.com>

Description Individual gene expression patterns are encoded into a series of eigenvector patterns ('WGCNA' package). Using the framework of linear model-based differential expression comparisons ('limma' package), time-course expression patterns for genes in different conditions are compared and analyzed for significant pattern changes. For reference, see: Greenham K, Sartor RC, Zorich S, Lou P, Mockler TC and McClung CR. eLife. 2020 Sep 30;9(4). <doi:10.7554/eLife.58993>.

License GPL (>= 2)

Imports limma, WGCNA, ggplot2

Suggests edgeR

NeedsCompilation no

Depends R (>= 2.10)

Repository CRAN

Date/Publication 2020-11-04 16:20:10 UTC

R topics documented:

`DiPALM-package` *DiPALM - Differential Pattern Analysis via Linear Modeling %*

Description

Individual gene expression patterns are encoded into a series of eigenvector patterns (WGCNA package). Using the framework of linear model-based differential expression comparisons (limma package), time-course expression patterns for genes in different conditions are compared and analyzed for significant pattern changes.

Author(s)

Ryan C. Sartor & Kathleen Greenham

Maintainer: Ryan C. Sartor <sartorry@gmail.com>

See Also

`limma` package `blockwiseModules` from the WGCNA package

`AdjustPvalue` *Determine p-values based on permutation test results %*

Description

Used to calculate the false discovery rate associated with a specified cutoff. It uses a vector of actual test scores and a vector of permuted scores. The result can be considered a corrected p-value or q-value.

Usage

```
AdjustPvalue(tVal, tVec, pVec)
```

Arguments

`tVal`	The threshold or cutoff value to use (given this value, the function returns the FDR associated with using any score above this one as a positive result).
`tVec`	A numeric vector of actual test result values (used to determine true positives).
`pVec`	A numeric vector of permuted test result values (used to determine false positives).

Details

This function assumes all test result values above the given threshold are true positives (TP) and all permuted result values above the threshold are false positives (FP). It calculates FDR as (FP)/(TP). When applied to individual test results, this works to estimate an adjusted p-value (or q-value) for that result.

Value

A numeric value representing the FDR associated with the given cutoff (tVal).

Author(s)

Ryan C. Sartor

Examples

```
# Returns a vector of adjusted p-values from a vector of test results
data("testData")
AdjPval_kMEs<-sapply(testData$testResults[1:100],function(x)
AdjustPvalue(tVal = x, tVec = testData$testResults, pVec = testData$permutedResults))
```

BlankPlot *Generate a blank plot window %*

Description

Creates a blank plotting window of desired size.

Usage

```
BlankPlot(xrng = c(0, 1), yrng = c(0, 1), Main = "", xlab = "", ylab = "", ...)
```

Arguments

xrng	The range of the x-axis.
yrng	The range of the y-axis.
Main	The main plot title.
xlab	X-axis label.
ylab	Y-axis label.
...	Additional arguments passed to the plot function.

Details

This function is used to initiate a blank plotting window in R. xrng and yrng are used to initiate the size of the plotting region.

Value

Generates a blank plot

Author(s)

Ryan C. Sartor

See Also

plot

Examples

```
BlankPlot(xrng=c(0,10),yrng=c(0,10),Main="Test",xlab="xlab",ylab="ylab")
```

BuildLimmaLM *Fit linear models to time-course expression patterns using the limma package %*

Description

This function fits a linear model (using lmFit form the limma package) to each set of kME values returned from BuildModMembership.

Usage

```
BuildLimmaLM(dataMat, designMat, contrastStr)
```

Arguments

dataMat	a matrix containing module membership values (a single element of the list returned by BuildModMembership). Module membership is a Pearson correlation value for a single gene compared to a module eigengene.
designMat	a linear model design matrix containing binary values (0 and 1) with rows as samples and columns as treatments. Use the model.matrix function in the stats package.
contrastStr	a linear model contrast specifying the desired differential comparison to be made using the linear model (see the limma package documentation for more details).

Details

This function uses the limma package functionality to carry out an analysis of differential patterning. Asking if a gene displays a significantly different expression pattern across time when looking at one condition compared to another. The concept of differential patterning is similar to differential expression. However, the expression level of most genes fluctuates significantly throughout the day. Most of this is due to diurnal or circadian rhythms. These regular changes in expression severely complicate differential expression analysis and give rise to the need for examining differential patterning. By "encoding" the expression pattern of each gene into a vector of module membership (kME) values using the `BuildModMembership` function, the DiPALM analysis provides a way to use the limma method to examine differential patterning.

Value

This function returns an `MArrayLM-class` object. This is a custom class from the `limma` package. It is an object that contains all information and results of individual linear models built for each gene in the input matrix.

Author(s)

Ryan C. Sartor

See Also

`BuildLimmaLM` , `lmFit` , `makeContrasts` , `contrasts.fit` , `eBayes`

Examples

```
data("testData")
# First calculate the module memberships
kMEsList<-BuildModMembership(MeMat=testData$moduleEigengenes, TCsLst=testData$timeCourseList)

# Then build the limma models
LimmaMods<-lapply(kMEsList, function(x)
BuildLimmaLM(dataMat = x, designMat = testData$modelDesign, contrastStr = testData$modelContrast))
```

`BuildModMembership` *Calculate module membership (kME) for all genes in a time-course data set %*

Description

This function calculates module membership (kME) between each gene's expression vector throughout a time-course and the module eigengenes of the data set (from the WGCNA package, using `blockwiseModules`). Module membership is defined as the Pearson correlation between the expression of a single gene and the expression of a single eigengene.

Usage

```
BuildModMembership(MeMat, TCsLst)
```

Arguments

MeMat	A data frame of eigengenes with each column being a different eigengene vector.
TCsLst	A named list of time-course matrices. Each element of the list is a time-course expression matrix (genes as rows and time-points as columns).

Details

The matrix of eigengene vectors, MeMat (see blockwiseModules from the WGCNA package) is used as a convenient way to summarize all the unique expression patterns (across time) that are present in a time-course data set. Any large gene expression data set from a complex organism will contain tens of thousands of genes. However, extensive coordinated regulation is present in all biological systems. Because of this co-regulation, usually only dozens of distinct expression patterns exist and most genes can be categorized by one (or a few) of them. By using a relatively small set of descriptive expression vectors (eigengenes), we can "encode" the pattern of expression for any single gene into a length N vector where N is the number of eigengenes and the values of the vector are the Pearson correlation of the gene to each eigengene (kME). These kME values can then be compared using the limma package to determine differential patterning similar to how expression values are compared to determine differential expression. This function is the first step to calculate all kMEs for each gene in a list of time-course matrices relative to the set of eigengenes for the whole data set.

Value

This function returns a list of kME matrices. The list contains a separate element for each eigengene. Each element is a matrix with a row for each gene and a column for each member of the input TCsLst.

Author(s)

Ryan C. Sartor

See Also

blockwiseModules

Examples

```
data("testData")
kMEsList<-BuildModMembership(MeMat=testData$moduleEigengenes, TCsLst=testData$timeCourseList)
```

exampleData *Example Data: Data for use with the DiPALM vignette %*

Description

A list of data objects that are used to run through the DiPALM example vignette.

Usage

```
data("exampleData")
```

Format

A List of 3 objects:

1. $rawCounts - A data frame [42383 rows X 48 columns] containing raw counts for mRNA expression data on *Brassica rapa* R500.
   - rows : Each row represents the expression of one gene across many different samples.
   - columns: Each column represents a different sample. Both drought-treated and properly watered samples are present, with two replicates of each time-course. Tissue was sampled at 4-hour intervals and column names are encoded as follows: S[replicate number] [D-Drought / W-Watered] [Time - sample] Example: S2D10 is part of the second replcate of the drought data and is the 10th time-point (ZT13, 13th hour after dawn on day two).
2. $moduleEigengenes - A dataframe [12 rows and 90 columns] representing 90 different expression vectors that describe descrete expression patterns found in the whole dataset ($timeCourseList).
   - Rows: each row represents a single time-point.
   - Columns: each column represents a distinct expression pattern seen repeatedly in the full data.
3. $geneAnnotations - A data frame [42383 rows and 5 columns] giving the locations within the *Brassica rapa* R500 genome where each gene is found. This is in SAF format which is used by the Linux Subread software package.
   - rows: Each row represents a single gene.
   - $GeneID: (character string) Unique gene accession.
   - $Chr: (character string) Chromosome number.
   - $Start: (integer) The position where the gene starts (first basepair).
   - $End: (integer) The position where the gene ends (last basepair).
   - $Strand: (character) is the gene on the positive [+] or negative [-] strand.

Details

Raw count data was summarized by mapping RNA-seq counts to the R500 *Brassica rapa* genome using the hisat2 aligner. Mapped reads were counted using the featureCounts function from the Linux Subread software package.

Source

Data is from: Greenham et al., eLife 2017;6:e29655

`ggPlotMultiDensities` *Plot multiple density distributions %*

Description

This function generates one or more smoothed density distributions using `ggplot2` functionality

Usage

```
ggPlotMultiDensities(denslist, main = "", xlab = "", ylab = "Normalized Frequency",
 scale = T, cols = c("red", "blue", "grey50", "black", "skyblue", "orange"),
 ledgx = "topright", ltype = NULL, lwidth = NULL, xrng = NULL, yrng = NULL, Ledge = T)
```

Arguments

`denslist`	A named list of populations. Each element is a vector of numbers. A separate density plot will be generated for each population.
`main`	The main plot title.
`xlab`	The x-axis label.
`ylab`	The y-axis label.
`scale`	When multiple densities are plotted, should they be scaled to represent the relative number of samples in each population.
`cols`	A vector of colors (one for each population in `denslist`).
`ledgx`	The x value from `legend` function. Determines where the legend is plotted.
`ltype`	A vector of integers specifying the line types to be used for each population (repeated if necessary for multiple populations).
`lwidth`	The line width to be used in the plot.
`xrng`	A numeric vector of 2, the x-axis range to be plotted.
`yrng`	A numeric vector of 2, the y-axis range to be plotted.
`Ledge`	A logical: should the legend be plotted?

Details

This function takes in a list of one or more numeric vectors. Each vector is considered a separate population and a smoothed density plot (similar to a histogram) will be generated for each and plotted together. The actual numeric y-axis values are meaningless. A density plot is normalized so that the total area under the curve is equal to 1. When scale = T, the total area under the density plots may not equal one, but instead the total area is scaled to the relative proportions of the total population sizes of each population.

Value

Generates multiple density plots on the same graph.

Author(s)

Ryan C. Sartor

See Also

ggplot2

Examples

```
data(testData)
require(ggplot2)

# Two populations of the same size, unscaled
ggPlotMultiDensities(denslist = list(Test=testData$testResults,Permuted=testData$permutedResults),
main = "Pattern Change Scores", xlab = "Differential Pattern Score",lwidth = 1)

# Two populations of different sizes, scaled
ggPlotMultiDensities(denslist = list(Test=testData$testResults[1:3000],
Permuted=testData$permutedResults), scale=TRUE , main = "Pattern Change Scores (Scaled)",
xlab = "Differential Pattern Score",lwidth = 1)
```

PlotTCs *Plot multiple time-course expression patterns for one gene %*

Description

A plotting function that generates a line plot for a single gene in multiple time-course expression sets. Each time-course is overlaid on the same plot.

Usage

```
PlotTCs(TClst, tgene, main = "", scale = TRUE, xlab = "Time",
    ylab = "", xAxsLabs = colnames(TClst[[1]]), ledgeX = "top",
    colAdj = 0.4, tcols = c("red", "red", "blue", "blue"), tltys = c(1, 1, 2, 2))
```

Arguments

TClst	A named list of time-course matrices. Each element of the list is a time-course expression matrix (Genes as rows and time points as columns).
tgene	A string of a single gene accession (found in the row names of one element of TClst).

main	Main figure title.
scale	Logical: should the expression values of each time-course be scaled and centered?
xlab	The X-axis label.
ylab	The Y-axis label.
xAxsLabs	A vector of names to be associated with each x-axis point.
ledgeX	The x value from legend function. Determines where the legend is plotted.
colAdj	Controls the alpha channel for the line colors using adjustcolor.
tcols	A vector of colors controlling the colors of the lines. Must list one color for each time-course in TClst.
tltys	A vector of integers controlling the line type (lty) of the lines. Must list one lty for each time-course in TClst.

Details

This function is used to view the expression pattern of a single gene across multiple time-course samples.

Value

The function generates a plot with multiple lines.

Author(s)

Ryan C. Sartor

Examples

```
data(testData)
PlotTCs(TClst = testData$timeCourseList, tgene = "BraA05g36370R" ,
scale = TRUE,tcols = c("red","red","blue","blue"), tltys = c(1,2,1,2), ledgeX = "topleft")
```

PlotTCsRibbon *Plot multiple averaged time-course expression patterns for a group of genes %*

Description

A plotting function that generates a ribbon plot representing the summarized expression of a set of genes. A separate ribbon is generated for each time-course. The ribbon represents + & - 1 standard deviation from the mean at each point.

Usage

```
PlotTCsRibbon(TClst, tgenes, main = "", xlab = "Time", ylab = "",
 xAxsLabs = colnames(TClst[[1]]),scale = TRUE, alpha = 0.1, tcols, tltys,
 splits = rep(1, length(TClst)))
```

Arguments

TClst	A named list of time-course matrices. Each element of the list is a time-course expression matrix (Genes as rows and time points as columns).
tgenes	A vector of gene accessions (found in the row names of one element of TClst).
main	The main title of the plot, if splits are defined, multiple plots will be generated and a vector of titles may be defined here.
xlab	The X-axis label.
ylab	The Y-axis label.
xAxsLabs	The time point labels for the x-axis, by default they will be the column names of the first time-course in TClst.
scale	Logical: should the expression values of each time-course be scaled and centered?
alpha	The alpha channel to be used for the ribbons (number from 0 - 1 defining transparency). See adjustcolor.
tcols	A vector of colors controlling the colors of the lines. Must list one color for each time-course in TClst.
tltys	A vector of integers controlling the line type (lty) of the lines. Must list one lty for each time-course in TClst.
splits	A vector of integers defining groups to be plotted on separate plots if desired (see examples).

Details

This can be used to plot the summarized expression pattern of an entire cluster of genes and compare between multiple time courses. Line plots are generated representing the mean expression value at each time point for the group. Ribbons are plotted along the lines in semi-transparent colors. The ribbons represent + and - one standard deviation from the mean expression level of the group. #

Value

This function generates multiple line plots representing the average expression of a group of genes with ribbons representing standard deviation.

Author(s)

Ryan C. Sartor

Examples

```
data("testData")

# Calculate the module memberships
kMEsList<-BuildModMembership(MeMat=testData$moduleEigengenes, TCsLst=testData$timeCourseList)

# Build the limma models
LimmaMods<-lapply(kMEsList, function(x)
BuildLimmaLM(dataMat = x, designMat = testData$modelDesign, contrastStr = testData$modelContrast))

# Get the adjusted Pvalues for each gene
AdjPval_kMEs<-sapply(testData$testResults[rownames(LimmaMods[[1]])],function(x)
AdjustPvalue(tVal = x, tVec = testData$testResults, pVec = testData$permutedResults))

# Cluster the genes base on the limma results
LimmaSums<-do.call(cbind,lapply(LimmaMods,function(x) x$t))
LimmaModsSig<-LimmaSums[names(AdjPval_kMEs)[which(AdjPval_kMEs<0.01)],]
patternCor<-cor(t(LimmaModsSig))
patternTree<-hclust(as.dist(1-patternCor),method = "complete")
patternClusters<-cutree(tree = patternTree, k = 25)
patternClusters<-tapply(rownames(patternCor),INDEX = patternClusters,function(x) x)

#Replicates are grouped together (same color + lty values)
PlotTCsRibbon(TClst = testData$timeCourseList, main="Drought Time-Course",
xAxsLabs = c(seq(1,23,4),seq(1,23,4)), xlab="ZT Time (hours)", tgenes = patternClusters[[15]],
scale = TRUE, tcols = c("red","red","blue","blue"), tltys = c(1,1,1,1))

#Replicates are in different groups (same color but different lty values)
PlotTCsRibbon(TClst = testData$timeCourseList, main="Drought Time-Course",
xAxsLabs = c(seq(1,23,4),seq(1,23,4)), xlab="ZT Time (hours)", tgenes = patternClusters[[15]],
scale = TRUE, tcols = c("red","red","blue","blue"), tltys = c(1,2,1,2))

#Replicates are represented on differnt plots (using the splits argument)
PlotTCsRibbon(TClst = testData$timeCourseList, main=c("Rep1","Rep2"),
xAxsLabs = c(seq(1,23,4),seq(1,23,4)), xlab="ZT Time (hours)", tgenes = patternClusters[[15]],
scale = TRUE, tcols = c("red","red","blue","blue"), tltys = c(1,2,1,2), splits = c(1,2,1,2))
```

`testData` *Test Data: Data for function testing %*

Description

A list of objects that can be used to test individual DiPALM functions.

Usage

```
data("testData")
```

Format

A List of 6 objects:

1. $timeCourseList - A List of 4 matrices containing mRNA expression data on *Brassica rapa* R500.
   - $Drought.R1 (Plants exposed to drought, Replicate 1) A matrix of time-course gene expression data [18428 Genes X 12 Time-points]
   - $Drought.R2 (Plants exposed to drought, Replicate 2) A matrix of time-course gene expression data [18428 Genes X 12 Time-points]
   - $Watered.R1 (Properly watered plants, Replicate 1) A matrix of time-course gene expression data [18428 Genes X 12 Time-points]
   - $Watered.R2 (Properly watered plants, Replicate 2) A matrix of time-course gene expression data [18428 Genes X 12 Time-points]
2. $moduleEigengenes - A dataframe [12 rows and 90 columns] representing 90 different expression vectors that describe descrete expression patterns found in the whole dataset ($timeCourseList).
   - Rows: each row represents a single time-point
   - Columns: each column represents a distinct expression pattern seen repeatedly in the full data
3. $modelDesign - A matrix [4 rows and 2 columns] used as a design matrix for a simple linear model. This matrix provides a mapping between the input instances (timeCourseList) and the two fixed effects fitted in the mode (Drought and Watered). This matrix was created with `model.matrix`.
4. $modelContrast - A character string specifying the contrast to be evaluated using the fitted model for each gene. This example is used to compare watered to drought conditions.
5. $testResults - A named numeric vector with 18428 values from DiPALM test results on actual data.
   - names: Gene accessions
   - values: The summed absolute values of t-values generated from limma model contrasts related to each eigengene pattern. This can be thought of as a DiPALM score for differential patterning.
6. $permutedResults- A named numeric vector with 18428 values from DiPALM test results on permuted data. This vector acts as a null distribution in order to estimate significance of actual test results.
   - names: Gene accessions
   - values: The summed absolute values of t-values generated from limma model contrasts related to each eigengene pattern using permuted expression data. This can be thought of as a null distribution of DiPALM scores for differential patterning.

Source

Data is from: Greenham et al., eLife 2017;6:e29655

Index