Package 'tempoR'

May 27, 2019

Title Characterizing Temporal Dysregulation

Version 1.0.4.4

Description TEMPO (TEmporal Modeling of Pathway Outliers) is a pathway-based outlier detection approach for finding pathways showing significant changes in temporal expression patterns across conditions. Given a gene expression data set where each sample is characterized by an age or time point as well as a phenotype (e.g. control or disease), and a collection of gene sets or pathways, TEMPO ranks each pathway by a score that characterizes how well a partial least squares regression (PLSR) model can predict age as a function of gene expression in the controls and how poorly that same model performs in the disease. TEMPO v1.0.3 is described in Pietras (2018) <doi:10.1145/3233547.3233559>.

Depends R (>= 3.0.2)

Imports doParallel (>= 1.0.10), foreach (>= 1.4.3), parallel (>= 3.0.2), pls (>= 2.5.0), grDevices, graphics, stats, utils

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Christopher Pietras [aut, cre]

Maintainer Christopher Pietras <christopher.pietras@tufts.edu>

Repository CRAN

Date/Publication 2019-05-27 08:30:03 UTC

R topics documented:

`dflatExample` *A subset of the DFLAT gene sets*

Description

20 selected gene sets from the DFLAT biological process gene sets

Usage

```
dflatExample
```

Format

A list of lists of gene identifiers, indexed by gene set name

Source

http://bcb.cs.tufts.edu/dflat//

`gse32472Example` *A subset of the GSE32472 BPD data set*

Description

A dataset of 97 normal or BPD samples at 5 days of life, with gestational age, class, and gene expression for 654 genes

Usage

```
gse32472Example
```

Format

A list

data a matrix of 97 samples and 654 genes

age a list of sample gestational ages, indexed by sample id

bpd a list of statuses, indexed by sample id

ctrl a list of sample ids without BPD

test a list of sample ids with BPD

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE32472

---

gse32472ExampleTempoResults

TEMPO results for the GSE32472 subset example data set

Description

The results data structure generated by tempo.run with default settings on the gse32472Example data set

Usage

```
gse32472ExampleTempoResults
```

Format

A list

ctrl a list of control sample ids used for scoring, identical to gse32472Example$ctrl

test a list of test sample ids used for scoring, identical to gse32472Example$test

train a list of normal sample ids used for model training, identical to gse32472Example$ctrl

scores a data frame of scores and signicance values for each gene set

pred a matrix of age predictions for each sample and gene set

Y age for each sample, identical to gse32472Example$age

genesets the gene sets used in the analysis, identical to dflatExample

reported a list of the gene sets which meet the significance thresholds for reporting

loadCLS *Load a categorical or continuous cls formatted file.*

Description

loadCLS loads a categorical or continuous cls formated file. Both file formats are described at the BROAD site.

Usage

```
loadCLS(target, sampleNames)
```

Arguments

target	a string indicating the location of the .cls file. loadCLS will automatically determine whether the specified file is categorical or continuous.
sampleNames	a list of sample names that correspond to each each entry in the .cls file.

Value

a list indexed by sample id containing the specified categorical or continuous variable.

Examples

```
# The .cls format does not include sample names, so it is necesary to have some already.
# In this case, we get them from the .gct file
sampleNames = names(loadGCT(file.path(path.package("tempoR"),"gse32472Example.gct")))

# Example continouous and categorical .cls files are included in the package
exampleAgesPath = file.path(path.package("tempoR"),"gse32472Example.age.cls")
exampleAges = loadCLS(exampleAgesPath, sampleNames)

exampleClassesPath = file.path(path.package("tempoR"),"gse32472Example.phen.cls")
exampleClasses = loadCLS(exampleClassesPath, sampleNames)
```

loadGCT *Load a Gene Cluster Text formatted file*

Description

loadGCT loads a Gene Cluster Text formatted file from a text file to the data structure used by TEMPO. A .gct file is organized as described at the BROAD site.

Usage

```
loadGCT(target)
```

Arguments

target	a string indicating the location of the .gct file

Value

a matrix with sample ids as row names and gene ids as column names

Examples

```
# An example gene expression data set is included in the package in .gct format
exampleDataPath = file.path(path.package("tempoR"),"gse32472Example.gct")
exampleData = loadGCT(exampleDataPath)
```

---

loadGMT *Load a Gene Matrix Transposed formatted file*

Description

loadGMT loads a Gene Matrix Transposed formatted file from a text file to the data structure used by TEMPO. In the .gmt format, each row represents a gene set, with tab delimited columns. The first column is a gene set name, the second columns and optional description, and the remaining columns contain gene ids for each gene in the gene set. The format is also described at the BROAD site.

Usage

```
loadGMT(target)
```

Arguments

target	a string indicating the location of the .gmt file

Value

a list indexed by gene set name of lists of gene ids

Examples

```
# An example collection of gene sets is included in the package in .gmt format
exampleGeneSetsPath = file.path(path.package("tempoR"),"dflatExample.gmt")
exampleGeneSets = loadGMT(exampleGeneSetsPath)
```

tempo.mkplot *Make a plot for a specified gene sets*

Description

Generate a plot of actual vs. predicted age, broken down by control sample or test sample, for a single specified gene set

Usage

```
tempo.mkplot(results, geneset)
```

Arguments

results	The results object returned by tempo.runInstance.or tempo.run
geneset	The name of a gene set

Examples

```
data("gse32472ExampleTempoResults")
tempo.mkplot(gse32472ExampleTempoResults, "cochlea development")
```

tempo.permutationTest *Permutation testing for TEMPO*

Description

Does permutation testing for the results of an instance of tempo run on the true class assignments. Called by tempo.run.

Usage

```
tempo.permutationTest(mainResult, X, Y, genesets, ctrl, test,
  train = NULL, numPerms = 500, nCores = 24, pMseCutoff = 0.05)
```

Arguments

mainResult	the data structure returned by tempo.runInstance when run using the true class assignments
X	a matrix with sample ids as row names and gene ids as column names
Y	a list indexed by sample ids, containing numerical values, e.g. ages
genesets	a list of lists. Outer list is indexed by gene set name, inner list contains all gene ids in a given gene set
ctrl	a list of sample ids. The list of control samples to use in scoring.

test	a list of sample ids. The list of test samples to use in scoring.
train	a list of sample ids. The list of control samples to train models on. If null, train on ctrl.
numPerms	number of permutations to do in permutation testing. Defaults to 500
nCores	number of thread to spawn for permutation testing. This should likely be set to some number less than or equal to the number of cores on your machine. If nCores is less than 0, nCores will be set to the return value of detectCores.
pMseCutoff	gene sets with a p-value associated with the control mean squared error below this cuttof are not evaluated further.

Value

mainResult, with the score data frame annotated with p-values for ctrlMSE ("pMSE") and score ("p") and fdr for the score p-value ("BH") for each gene set. Note that "BH" will be 2 for any gene set where pMSE > pMseCutoff

Examples

```
data("dflatExample")
data("gse32472Example")
data("gse32472ExampleTempoResults")


# Cross-validation can be run seperately once the initial model-building and scoring is complete.
# Note that running this example may take several minutes.
results = tempo.runInstance(ctrl=gse32472Example$ctrl,
    test=gse32472Example$test,
    genesets=dflatExample,
    X=gse32472Example$data,
    Y=gse32472Example$age)
results = tempo.permutationTest(mainResult=results,
    ctrl=gse32472Example$ctrl,
    test=gse32472Example$test,
    genesets=dflatExample,
    X=gse32472Example$data,
    Y=gse32472Example$age,
    nCores=-1)


# Reporting thresholds, number of permutations, and number of CPU cores used can all be changed.
# This command is suitable for demonstration purposes, but significance values will not be
# meaningful.
results2 = tempo.permutationTest(mainResult=gse32472ExampleTempoResults,
    ctrl=gse32472Example$ctrl,
    test=gse32472Example$test,
    genesets=dflatExample,
    X=gse32472Example$data,
    Y=gse32472Example$age,
    numPerms=2,nCores=2,pMseCutoff = 1)
```

`tempo.run` *The main method.*

Description

For each gene set in genesets, train a model on the control samples of Y~X. Generates scores for each gene set, performs permutation testing, and (optionally) writes a table of results and plots

Usage

```
tempo.run(X, Y, genesets, phen = NULL, ctrl = NULL, test = NULL,
  train = NULL, output = "", numPerms = 500, validation = "LOO",
  minGsSize = 2, nCores = 24, pCutoff = 0.05, fdrCutoff = 0.25,
  pMseCutoff = 0.05)
```

Arguments

`X`	a matrix with sample ids as row names and gene ids as column names.
`Y`	a list indexed by sample ids, containing numerical values.
`genesets`	a list of lists. Outer list is indexed by gene set name, inner list contains all gene ids in a given gene set
`phen`	a list indexed by sample ids, containing phenotypes. If ctrl and test are null, the phenotype of the first non-NA entry in the list is assumed to be the control phenotype; all others are test phenotypes
`ctrl`	a list of sample ids. The list of control samples to use in scoring. If train is null, these are also the training samples. Used only if phen is null.
`test`	a list of sample ids. The list of test samples to use in scoring. Used only if phen is null.
`train`	a list of sample ids. The list of control samples to train models on. If null, samples ids in ctrl are used. Used only if phen is null.
`output`	optional. A prefix to write table of output and plots for each reported gene set
`numPerms`	number of permutations to do in permutation testing. Defaults to 500
`validation`	type of validation to do. Defaults to leave one out, which generates deterministic results. "CV" performs 10-fold cross-validation, which is significantly faster but generates non-deterministic results.
`minGsSize`	minimum acceptable size for a gene set, considering only features which exist in `colnames(X)`
`nCores`	number of thread to spawn for permutation testing. This should likely be set to some number less than or equal to the number of cores on your machine. If nCores is less than 0, nCores will be set to the return value of `detectCores`.
`pCutoff`	report only gene sets with a p-value below this cutoff
`fdrCutoff`	report only gene sets with a FDR below this cutoff
`pMseCutoff`	report only gene sets with the p-value of the control mean squared error below this cutoff

Value

the output from tempo.runInstance annotated with significance for each gene set

Examples

```
data("dflatExample")
data("gse32472Example")


# This runs a simple TEMPO analysis on the example data set with default settings
# (with the exception of nCores, which will instead be automatically set to a suitable
# value) and saves the output in a two temporary files.
# Note that running this example may take several minutes.
results = tempo.run(phen=gse32472Example$bpd,
    genesets=dflatExample,
    X=gse32472Example$data,
    Y=gse32472Example$age,
    output=tempfile(tmpdir = tempdir()),
    nCores=-1)

# If phen is used, the first item in the list is assumed to the control phenotype
# and all other phenotypes test. Specifiy ctrl and test exactly for more control.
# Note that running this example may take several minutes.
results = tempo.run(ctrl=gse32472Example$ctrl,
    test=gse32472Example$test,
    genesets=dflatExample,
    X=gse32472Example$data,
    Y=gse32472Example$age,
    nCores=-1)


# If training models on a held out set of data is desired, train can be specified seperately
# Note that running this example may take several minutes.
results2 = tempo.run(train=gse32472Example$ctrl[1:10],
    ctrl=gse32472Example$ctrl[11:20],
    test=gse32472Example$test,
    genesets=dflatExample,
    X=gse32472Example$data,
    Y=gse32472Example$age,
    nCores=-1)


# Reporting thresholds, number of permutations, and number of CPU cores used can all be changed.
# This command is suitable for demonstration purposes, but significance values will not be
# meaningful.
results3 = tempo.run(phen=gse32472Example$bpd,
    genesets=dflatExample,X=gse32472Example$data,
    Y=gse32472Example$age,output=tempfile(tmpdir = tempdir()),
    numPerms=2,nCores=2,pCutoff=1,fdrCutoff=2,pMseCutoff = 1)
```

`tempo.runInstance` *Build models for all pathways using the control data and test on the test population.*

Description

Build models for all pathways using the control data and test on the test population.

Usage

```
tempo.runInstance(X, Y, genesets, ctrl, test, train = NULL, comps = 10,
  validation = "CV")
```

Arguments

`X`	a matrix with sample ids as row names and gene ids as column names.
`Y`	a list indexed by sample ids, containing numerical values.
`genesets`	a list of lists. Outer list is indexed by gene set name, inner list contains all gene ids in a given gene set
`ctrl`	a list of sample ids. The list of control samples to use in scoring.
`test`	a list of sample ids. The list of test samples to use in scoring.
`train`	a list of sample ids. The list of control samples to train models on. If null, train on ctrl.
`comps`	maximum number of components to use in the plsr model
`validation`	"CV" for 10-fold cross-validation, "LOO" for leave-one-out cross-validation. "CV" is nondeterministic and should not be used where exactly reproducible results are important

Value

a list with the following entries

- `ctrl` a list of the sample names used from the control set used for scoring
- `test` a list of the sample names used from the test set used for scoring
- `train` a list of the sample names used from the control set to train models
- `scores` a data frame with gene set ids as row names and a "ctrlMSE" and "score" entry for each gene set with the MSE of control age predictions in cross-validation and the calulated score, respectively
- `pred` a matrix with gene set labels, where the age prediction for sample j from the models for gene set i are at i,j
- `Y` the continuous variable of interest that models are built with respect to, e.g. age
- `genesets` the gene sets used in the analysis

Examples

```
data("dflatExample")
data("gse32472Example")

# It is possible to run just the model-building and scoring functions and skip
# cross-validation.  This is not recommended for general use, but may be useful
# in some cases
results = tempo.runInstance(ctrl=gse32472Example$ctrl,
    test=gse32472Example$test,
    genesets=dflatExample,
    X=gse32472Example$data,
    Y=gse32472Example$age)
```

`tempo.writeOutput` *Write output*

Description

Write the score data frame and plots of actual vs. predicted age for all reported gene sets to disk.

Usage

```
tempo.writeOutput(results, filename)
```

Arguments

`results`	The results object returned by `tempo.runInstance`.
`filename`	The prefix for the filename. Writes filename.pdf and filename.table (a tab-delimited text file) for all reported gene sets

Examples

```
# In this example, text output will be written to tempfile.table and
# plots printed to tempfile.pdf
data("gse32472ExampleTempoResults")
tempo.writeOutput(gse32472ExampleTempoResults, tempfile(tmpdir = tempdir()))
```

Index