Package 'plgem'

March 11, 2025

Title Detect differential expression in microarray and proteomics datasets with the Power Law Global Error Model (PLGEM)

Version 1.79.0

Author Mattia Pelizzola <mattia.pelizzola@gmail.com> and Norman Pavelka <normanpavelka@gmail.com>

Description The Power Law Global Error Model (PLGEM) has been shown to faithfully model the variance-versus-mean dependence that exists in a variety of genome-wide datasets, including microarray and proteomics data. The use of PLGEM has been shown to improve the detection of differentially expressed genes or proteins in these datasets.

Maintainer Norman Pavelka <normanpavelka@gmail.com>

Imports utils, Biobase (>= 2.5.5), MASS, methods

Depends R (>= 2.10)

License GPL-2

URL http://www.genopolis.it

biocViews ImmunoOncology, Microarray, DifferentialExpression, Proteomics, GeneExpression, MassSpectrometry

git_url https://git.bioconductor.org/packages/plgem

git_branch devel

git_last_commit 4123c5b

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-03-10

Contents

`LPSeset` *ExpressionSet for Testing PLGEM*

Description

This `ExpressionSet` object contains a subset of the microarray data used in References for **PLGEM** set-up and validation. Briefly, it contains normalized gene expression values from a total of 6 hybridizations on MG-U74Av2 Affymetrix GeneChip microarrays. Two experimental conditions are represented in this dataset: the baseline condition ('C') contains data of immature murine dendritic cells (4 replicates); the other condition ('LPS') contains data of the same cells stimulated for 4 hours with LPS (2 replicates).

Usage

```
data(LPSeset)
```

Format

An object of class `ExpressionSet`.

Author(s)

Mattia Pelizzola <mattia.pelizzola@gmail.com>

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17;5:203.; http://www.biomedcentral.com/1471-2105/5/203

Examples

```
data(LPSeset)
library(Biobase)
head(exprs(LPSeset))
```

plgem.deg *Selection of Differentially Expressed Genes/Proteins With PLGEM*

Description

This function selects differentially expressed genes/proteins (DEG) at a given significance level, based on observed **PLGEM** signal-to-noise ratio (STN) values (typically obtained via a call to plgem.obsStn) and pre-computed p-values (typically obtained via a call to plgem.pValue).

Usage

```
plgem.deg(observedStn, plgemPval, delta=0.001, verbose=FALSE)
```

Arguments

observedStn	list containing a matrix of observed PLGEM-STN values; output of function plgem.obsStn.
plgemPval	matrix of p-values; output of function plgem.pValue.
delta	numeric vector; the significance level(s) to be used for the selection of DEG; value(s) must be between 0 and 1 (excluded).
verbose	logical; if TRUE, comments are printed out while running.

Details

This function allows for the selection of DEG by setting a significance cut-off on pre-calculated p-values. The significance level delta roughly represents the false positive rate of the DEG selection, e.g. if a delta of 0.001 is chosen in a microarray dataset with 10,000 genes (none of which is truly differentially expressed), on average 10 genes/proteins are expected to be selected by chance alone.

Value

A list of four elements:

fit	the input plgemFit.
PLGEM.STN	the input matrix of observed PLGEM-STN values (see plgem.obsStn for details).
p-value	the input matrix of p-values (see plgem.pValue for details).
significant	a list with a number of elements equal to the number of different significance levels (delta) used as input. Each element of this list is again a list, whose number of elements correspond to the number of performed comparisons (i.e. the number of conditions in the starting ExpressionSet minus the baseline). Each of these second level elements is a character vector of significant gene/protein names that passed the statistical test at the corresponding significance level.

Author(s)

Mattia Pelizzola <mattia.pelizzola@gmail.com>

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17; 5:203; http://www.biomedcentral.com/1471-2105/5/203.

Pavelka N, Fournier ML, Swanson SK, Pelizzola M, Ricciardi-Castagnoli P, Florens L, Washburn MP. Statistical similarities between transcriptomics and quantitative shotgun proteomics data. Mol Cell Proteomics. 2008 Apr; 7(4):631-44; http://www.mcponline.org/cgi/content/abstract/7/4/631.

See Also

plgem.fit, plgem.obsStn, plgem.resampledStn, plgem.pValue, run.plgem

Examples

```
data(LPSeset)
LPSfit <- plgem.fit(data=LPSeset, fittingEval=FALSE)
LPSobsStn <- plgem.obsStn(data=LPSeset, plgemFit=LPSfit)
set.seed(123)
LPSresampledStn <- plgem.resampledStn(data=LPSeset, plgemFit=LPSfit)
LPSpValues <- plgem.pValue(LPSobsStn, LPSresampledStn)
LPSdegList <- plgem.deg(observedStn=LPSobsStn, plgemPval=LPSpValues,
  delta=0.001)
```

---

plgem.fit *PLGEM Fitting and Evaluation*

Description

Function for fitting and evaluating goodness of fit of a **PLGEM** on a set of replicated samples defined in an ExpressionSet.

Usage

```
plgem.fit(data, covariate=1, fitCondition=1, p=10, q=0.5,
  trimAllZeroRows=FALSE, zeroMeanOrSD=c("replace", "trim"), fittingEval=FALSE,
  plot.file=FALSE, prefix=NULL, gPar=setGpar(), verbose=FALSE)
```

Arguments

data	an object of class ExpressionSet; see Details for important information on how the phenoData slot of this object will be interpreted by the function.
covariate	integer, numeric or character; specifies the covariate to be used to fit the **PLGEM**. See Details for how to specify the covariate.
fitCondition	integer, numeric or character; specifies the condition to be used to fit the **PLGEM**. See Details for how to specify the fitCondition.
p	integer (or coercible to integer); number of intervals used to partition the expression value range.
q	numeric in [0,1]; the quantile of standard deviation used for **PLGEM** fitting.
trimAllZeroRows	logical; if TRUE, rows in the data set containing only zero values are trimmed before fitting **PLGEM**.
zeroMeanOrSD	either NULL or character; what should be done if a row with non-positive mean or zero standard deviation is encountered before fitting **PLGEM**? Current options are one of "replace" or "trim". Partial matching is used to switch between the options and setting the value to NULL will cause the default behaviour to be enforced, i.e. to "replace" (see Details).
fittingEval	logical; if TRUE, the fitting is evaluated generating a diagnostic plot.
plot.file	logical; if TRUE, a png file is written on the current working directory.
prefix	optional character to use as a prefix of the file name to be written.
gPar	optional list of graphical parameters to define plotting boundaries in PLGEM fitting evaluation plots. If left unspecified suitable boundaries will be determined from the data. The recommended way to set these parameters if via a call to setGPar().
verbose	logical; if TRUE, comments are printed out while running.

Details

plgem.fit fits a Power Law Global Error Model (**PLGEM**) to an ExpressionSet and optionally evaluates the quality of the fit. This **PLGEM** aims to find the mathematical relationship between standard deviation and mean gene expression values (or protein abundance levels) in a set of replicated microarray (or proteomics) samples, according to the following power law:

$$\log(modeledSpread) = PLGEMslope * \log(mean) + PLGEMintercept$$

It has been demonstrated that this model fits to Affymetrix GeneChip datasets, as well as to datasets of normalized spectral counts obtained by mass spectrometry-based proteomics. Technically, two replicates are required and sufficient to fit a **PLGEM**. Having 3 or more replicates, of course, improves the fitting and is recommended (see References for details).

The phenoData slot of the ExpressionSet given as input is expected to contain the necessary information to distinguish the various experimental conditions from one another. The columns of the pData are referred to as 'covariates'. There has to be at least one covariate defined in the input ExpressionSet. The sample attributes according to this covariate must be distinct for samples that

are to be treated as distinct experimental conditions and identical for samples that are to be treated as replicates.

There is a couple different ways to specify the `covariate`: If an `integer` or a `numeric` is given, it will be taken as the covariate number (in the same order in which the covariates appear in the `colnames` of the `pData`). If a `character` is given, it will be taken as the covariate name itself (in the same way the covariates are specified in the `colnames` of the `pData`). By default, the first covariate appearing in the `colnames` of the `pData` is used.

Similarly, there is a couple different ways to specify on which experimental condition to fit the model. The available 'condition names' are taken from `unique(as.character(pData(data)[, covariate]))`. If `fitCondition` is given as a `character`, it will be taken as the condition name itself. If `fitCondition` is given as an `integer` or a `numeric` value, it will be taken as the condition number (in the same order of appearance as in the 'condition names'). By default, the first condition name is used.

Setting `trimAllZeroRows=TRUE` is especially useful in proteomics data sets, where there is no guarantee of identifying a protein across all experimental conditions. Since **PLGEM** is fitted only to the data corresponding to a single experimental condition (as defined by `fitCondition`), it is possible to generate a non-negligible number of rows containing only zero values, even if there were no such rows in the original (complete) data set containing all experimental conditions.

Setting `zeroMeanOrSD="replace"` (the current default, for backward compatibility) will cause the function to replace zero or negative means with the smallest positive mean found in the data set and to replace zero standard deviations with the smallest non-zero standard deviation found in the data set. Setting `zeroMeanOrSD="trim"` is the current recommended option, especially for spectral counting proteomics data sets that are typically characterized by a high data granularity or for microarray data sets with a small number of replicates. In both cases, there are chances for data values for a same gene or protein to be identical across replicates (and therefore with zero standard deviation) by chance alone. Note that setting `trimAllZeroRows=TRUE` does not guarantee that there will be no rows with zero mean or zero standard deviation.

If argument `fittingEval` is set to `TRUE`, a graphical control of the goodness of the **PLGEM** fitting is produced and a plot containing four panels is generated. The top-left panel shows the power law, characterized by a 'SLOPE' and an 'INTERCEPT'. The top-right panel represents the distribution of model residuals. The bottom-left reports the contour plot of ranked residuals. The bottom-right panel finally shows the relationship between the distribution of observed residuals and the normal distribution. A good fit normally gives a horizontal symmetric rank-plot and a near normal distribution of residuals.

Warnings are issued if the fitted PLGEM slope is above 1 or under 0.5, if the adjusted r^2 is below 0.95 or if the Pearson correlation coefficient is below 0.85. These are the ranges of values inside which most GeneChip MAS5 dataset and NSAF proteomics dataset have been empirically observed to lie (see References).

Value

A `list` of six elements (see Details):

`SLOPE`	the slope of the fitted PLGEM.
`INTERCEPT`	the intercept of the fitted PLGEM.
`DATA.PEARSON`	the Pearson correlation coefficient between the $\log(sd)$ and the $\log(mean)$ in the original data.

ADJ.R2.MP	the adjusted r^2 of PLGEM fitted on the modelling points.
COVARIATE	a character indicating the covariate used for fitting.
FIT.CONDITION	a character indicating the condition used for fitting.

Author(s)

Mattia Pelizzola <mattia.pelizzola@gmail.com>

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17; 5:203; http://www.biomedcentral.com/1471-2105/5/203.

Pavelka N, Fournier ML, Swanson SK, Pelizzola M, Ricciardi-Castagnoli P, Florens L, Washburn MP. Statistical similarities between transcriptomics and quantitative shotgun proteomics data. Mol Cell Proteomics. 2008 Apr; 7(4):631-44; http://www.mcponline.org/cgi/content/abstract/7/4/631.

See Also

setGpar, plgem.obsStn, plgem.resampledStn, plgem.pValue, plgem.deg, run.plgem

Examples

```
data(LPSeset)
LPSfit <- plgem.fit(data=LPSeset, fittingEval=TRUE)
as.data.frame(LPSfit)
```

---

plgem.obsStn *Computation of Observed PLGEM-STN Statistics*

Description

This function computes observed signal-to-noise ratio (STN) values using **PLGEM** fitting parameters (obtained via a call to function plgem.fit) to detect differential expression in an ExpressionSet, containing either microarray or proteomics data.

Usage

```
plgem.obsStn(data, plgemFit, covariate=1, baselineCondition=1,
  verbose=FALSE)
```

Arguments

data	an object of class ExpressionSet; see Details for important information on how the phenoData slot of this object will be interpreted by the function.
plgemFit	list; the output of function plgem.fit.
covariate	integer, numeric or character; specifies the covariate to be used to distinguish the various experimental conditions from one another. See Details for how to specify the covariate.
baselineCondition	integer, numeric or character; specifies the condition to be treated as the baseline. See Details for how to specify the baselineCondition.
verbose	logical; if TRUE, comments are printed out while running.

Details

The phenoData slot of the ExpressionSet given as input is expected to contain the necessary information to distinguish the various experimental conditions from one another. The columns of the pData are referred to as 'covariates'. There has to be at least one covariate defined in the input ExpressionSet. The sample attributes according to this covariate must be distinct for samples that are to be treated as distinct experimental conditions and identical for samples that are to be treated as replicates.

There is a couple different ways how to specify the covariate: If an integer or a numeric is given, it will be taken as the covariate number (in the same order in which the covariates appear in the colnames of the pData). If a character is given, it will be taken as the covariate name itself (in the same way the covariates are specified in the colnames of the pData). By default, the first covariate appearing in the colnames of the pData is used.

Similarly, there is a couple different ways how to specify which experimental condition to treat as the baseline. The available 'condition names' are taken from unique(as.character(pData(data)[, covariate])). If baselineCondition is given as a character, it will be taken as the condition name itself. If baselineCondition is given as an integer or a numeric value, it will be taken as the condition number (in the same order of appearance as in the 'condition names'). By default, the first condition name is used.

PLGEM-STN values are a measure of the degree of differential expression between a condition and the baseline:

$$STN = \frac{mean_{condition} - mean_{baseline}}{modeledSpread_{condition} + modeledSpread_{baseline}},$$

where:

$$\log(modeledSpread) = PLGEMslope * \log(mean) + PLGEMintercept$$

plgem.obsStn determines the observed PLGEM-STN values for each gene or protein in data; see References for details.

Value

A list of two elements:

fit	the input plgemFit.
PLGEM.STN	a matrix of observed PLGEM-STN values. The rownames of this matrix are identical to the rownames of data. The colnames represent the different experimental conditions that were compared to the baseline.

Author(s)

Mattia Pelizzola <mattia.pelizzola@gmail.com>

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17; 5:203; http://www.biomedcentral.com/1471-2105/5/203.

Pavelka N, Fournier ML, Swanson SK, Pelizzola M, Ricciardi-Castagnoli P, Florens L, Washburn MP. Statistical similarities between transcriptomics and quantitative shotgun proteomics data. Mol Cell Proteomics. 2008 Apr; 7(4):631-44; http://www.mcponline.org/cgi/content/abstract/7/4/631.

See Also

plgem.fit, plgem.resampledStn, plgem.pValue, plgem.deg, run.plgem

Examples

```
data(LPSeset)
LPSfit <- plgem.fit(data=LPSeset)
LPSobsStn <- plgem.obsStn(data=LPSeset, plgemFit=LPSfit)
head(LPSobsStn[["PLGEM.STN"]])
```

---

plgem.pValue *Computation of PLGEM p-values*

Description

This function computes p-values for observed PLGEM signal-to-noise ratio (STN) values (typically obtained via a call to plgem.obsStn) from resampled STN values (typically obtained via a call to plgem.resampledStn).

Usage

```
plgem.pValue(observedStn, plgemResampledStn, verbose=FALSE)
```

Arguments

observedStn	list containing a matrix of observed PLGEM-STN values; output of function plgem.obsStn.
plgemResampledStn	list containing a matrix of resampled PLGEM-STN values; output of function plgem.resampledStn.
verbose	logical; if TRUE, comments are printed out while running.

Details

The p-value of each given observed STN value is computed based on the quantile that the given value occupies in the corresponding distribution of resampled PLGEM-STN values, based on the following relationship:

$$P = min(2 * quantile, 2 * (1 - quantile))$$

Value

A matrix with the same dimensions and dimnames as the input observedStn$PLGEM.STN, where each entry represents the p-value of the corresponding observed PLGEM-STN value.

Author(s)

Mattia Pelizzola <mattia.pelizzola@gmail.com>

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17; 5:203; http://www.biomedcentral.com/1471-2105/5/203.

Pavelka N, Fournier ML, Swanson SK, Pelizzola M, Ricciardi-Castagnoli P, Florens L, Washburn MP. Statistical similarities between transcriptomics and quantitative shotgun proteomics data. Mol Cell Proteomics. 2008 Apr; 7(4):631-44; http://www.mcponline.org/cgi/content/abstract/7/4/631.

See Also

plgem.fit, plgem.obsStn, plgem.resampledStn, plgem.deg, run.plgem

Examples

```
data(LPSeset)
LPSfit <- plgem.fit(data=LPSeset)
LPSobsStn <- plgem.obsStn(data=LPSeset, plgemFit=LPSfit)
head(LPSobsStn[["PLGEM.STN"]])
set.seed(123)
LPSresampledStn <- plgem.resampledStn(data=LPSeset, plgemFit=LPSfit)
```

```
LPSpValues <- plgem.pValue(LPSobsStn, LPSresampledStn)
head(LPSpValues)
```

---

plgem.resampledStn *Computation of Resampled PLGEM-STN Statistics*

Description

This function computes resampled signal-to-noise ratio (STN) values using **PLGEM** fitting parameters (obtained via a call to function plgem.fit) to detect differential expression in an ExpressionSet, containing either microarray or proteomics data.

Usage

```
plgem.resampledStn(data, plgemFit, covariate=1, baselineCondition=1,
  iterations="automatic", verbose=FALSE)
```

Arguments

data	an object of class ExpressionSet; see Details for important information on how the phenoData slot of this object will be interpreted by the function.
plgemFit	list; the output of function plgem.fit.
covariate	integer, numeric or character; specifies the covariate to be used to distinguish the various experimental conditions from one another. See Details for how to specify the covariate.
baselineCondition	integer, numeric or character; specifies the condition to be treated as the baseline. See Details for how to specify the baselineCondition.
verbose	logical; if TRUE, comments are printed out while running.
iterations	number of iterations for the resampling step; if "automatic" it is automatically determined.

Details

The phenoData slot of the ExpressionSet given as input is expected to contain the necessary information to distinguish the various experimental conditions from one another. The columns of the pData are referred to as 'covariates'. There has to be at least one covariate defined in the input ExpressionSet. The sample attributes according to this covariate must be distinct for samples that are to be treated as distinct experimental conditions and identical for samples that are to be treated as replicates.

There is a couple different ways how to specify the covariate: If an integer or a numeric is given, it will be taken as the covariate number (in the same order in which the covariates appear in the colnames of the pData). If a character is given, it will be taken as the covariate name itself (in the same way the covariates are specified in the colnames of the pData). By default, the first covariate appearing in the colnames of the pData is used.

Similarly, there is a couple different ways how to specify which experimental condition to treat as the baseline. The available 'condition names' are taken from `unique(as.character(pData(data)[, covariate]))`. If `baselineCondition` is given as a `character`, it will be taken as the condition name itself. If `baselineCondition` is given as an `integer` or a `numeric` value, it will be taken as the condition number (in the same order of appearance as in the 'condition names'). By default, the first condition name is used.

PLGEM-STN values are a measure of the degree of differential expression between a condition and the baseline:

$$STN = \frac{mean_{condition} - mean_{baseline}}{modeledSpread_{condition} + modeledSpread_{baseline}},$$

where:

$$\log(modeledSpread) = PLGEMslope * \log(mean) + PLGEMintercept$$

`plgem.resampledStn` determines the resampled PLGEM-STN values for each gene or protein in `data` using a resampling approach; see References for details. The number of iterations should be chosen depending on the number of available replicates of the condition used for fitting the model.

Value

A `list` of two elements:

`RESAMPLED.STN`	`matrix` of resampled PLGEM-STN values, with `rownames` identical to those in `data`, and `colnames` representing the different number of replicates found in the different comparisons; see References for details.
`REPL.NUMBER`	the number of replicates found for each experimental condition; see References for details.

Author(s)

Mattia Pelizzola <mattia.pelizzola@gmail.com>

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17; 5:203; http://www.biomedcentral.com/1471-2105/5/203.

Pavelka N, Fournier ML, Swanson SK, Pelizzola M, Ricciardi-Castagnoli P, Florens L, Washburn MP. Statistical similarities between transcriptomics and quantitative shotgun proteomics data. Mol Cell Proteomics. 2008 Apr; 7(4):631-44; http://www.mcponline.org/cgi/content/abstract/7/4/631.

See Also

`plgem.fit`, `plgem.obsStn`, `plgem.pValue`, `plgem.deg`, `run.plgem`

Examples

```
data(LPSeset)
LPSfit <- plgem.fit(data=LPSeset)
LPSobsStn <- plgem.obsStn(data=LPSeset, plgemFit=LPSfit)
set.seed(123)
LPSresampledStn <- plgem.resampledStn(data=LPSeset, plgemFit=LPSfit)
plot(density(LPSresampledStn[["RESAMPLED.STN"]], bw=0.01), col="black", lwd=2,
  xlab="PLGEM STN values",
  main="Distribution of observed\nand resampled PLGEM-STN values")
lines(density(LPSobsStn[["PLGEM.STN"]], bw=0.01), col="red")
legend("topright", legend=c("resampled", "observed"), col=c("black", "red"),
  lwd=2:1)
```

---

plgem.write.summary *Write the Result of a PLGEM Analysis to the Working Directory*

Description

This function writes the output of function plgem.deg or run.plgem to a series of files in the current working directory.

Usage

```
plgem.write.summary(x, prefix=NULL, verbose=FALSE)
```

Arguments

x	list; the output of either plgem.deg or run.plgem (see corresponding help papes for details).
prefix	optional character to use as a prefix of the file names to be written.
verbose	logical; if TRUE, comments are printed out while running.

Details

This function writes three types of files to the current working directory:

1) A comma-separated text file containing the PLGEM fitting parameters;

2) A comma-separated text file containing the observed STN values and their associated p-values for all performed comparisons; (STN values and p-values from different comparisons appear in different columns, with column headers reflecting the underlying comparison)

3) One or more plain text files containing the identifiers of the significant genes or proteins, with filenames reflecting the specific comparisons that were performed (i.e. which experimental condition was compared to the baseline) and the specific significance threshold that were used in the DEG selection step.

Before files are written, the function checks for existence of files with identical names in the working directory and prompts the user to decide whether to abort the writing process or to overwrite the existing files.

Value

The function returns no value. It is called for its side effect to write files to the working directory.

Author(s)

Mattia Pelizzola <mattia.pelizzola@gmail.com>

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17; 5:203; http://www.biomedcentral.com/1471-2105/5/203.

Pavelka N, Fournier ML, Swanson SK, Pelizzola M, Ricciardi-Castagnoli P, Florens L, Washburn MP. Statistical similarities between transcriptomics and quantitative shotgun proteomics data. Mol Cell Proteomics. 2008 Apr; 7(4):631-44; http://www.mcponline.org/cgi/content/abstract/7/4/631.

See Also

plgem.deg, run.plgem

Examples

```
## Not run:
  data(LPSeset)
  LPSdegList <- run.plgem(LPSeset, fitting.eval=FALSE)
  plgem.write.summary(LPSdegList, prefix="test", verbose=TRUE)
## End(Not run)
```

---

run.plgem *Wrapper for Power Law Global Error Model (PLGEM) analysis method*

Description

This function automatically performs **PLGEM** fitting and evaluation, determination of observed and resampled PLGEM-STN values, and selection of differentially expressed genes/proteins (DEG) using the **PLGEM** method.

Usage

```
run.plgem(esdata, signLev=0.001, rank=100, covariate=1,
  baselineCondition=1, Iterations="automatic", trimAllZeroRows=FALSE,
  zeroMeanOrSD=c("replace", "trim"), fitting.eval=TRUE,
  plotFile=FALSE, writeFiles=FALSE, Prefix=NULL, Verbose=FALSE)
```

Arguments

esdata	an object of class ExpressionSet; see Details for important information on how the phenoData slot of this object will be interpreted by the function.
signLev	numeric vector; significance level(s) for the DEG selection. Value(s) must be in (0,1).
rank	integer (or coercible to integer); the number of genes or proteins to be selected according to their PLGEM-STN rank. Only used if number of available replicates is too small to perform resampling (see Details).
covariate	integer, numeric or character; specifies the covariate to be used to distinguish the various experimental conditions from one another. See Details for how to specify the covariate.
baselineCondition	integer, numeric or character; specifies the condition to be treated as the baseline. See Details for how to specify the baselineCondition.
Iterations	number of iterations for the resampling step; if "automatic" it is automatically determined.
trimAllZeroRows	logical; if TRUE, rows in the data set containing only zero values are trimmed before fitting **PLGEM**. See help page of function plgem.fit for details.
zeroMeanOrSD	either NULL or character; what should be done if a row with non-positive mean or zero standard deviation is encountered before fitting **PLGEM**? Current options are one of "replace" or "trim". Partial matching is used to switch between the options and setting the value to NULL will cause the default behaviour to be enforced, i.e. to "replace". See help page of function plgem.fit for details.
fitting.eval	logical; if TRUE, the fitting is evaluated generating a diagnostic plot.
plotFile	logical; if TRUE, the generated plot is written on a file.
writeFiles	logical; if TRUE, the generated list of DEG is written on disk file(s).
Prefix	optional character to use as a prefix of the file names to be written if writeFiles=TRUE.
Verbose	logical; if TRUE, comments are printed out while running.

Details

The phenoData slot of the ExpressionSet given as input is expected to contain the necessary information to distinguish the various experimental conditions from one another. The columns of the pData are referred to as 'covariates'. There has to be at least one covariate defined in the input ExpressionSet. The sample attributes according to this covariate must be distinct for samples that are to be treated as distinct experimental conditions and identical for samples that are to be treated as replicates.

There is a couple different ways how to specify the covariate: If an integer or a numeric is given, it will be taken as the covariate number (in the same order in which the covariates appear in the colnames of the pData). If a character is given, it will be taken as the covariate name itself (in the same way the covariates are specified in the colnames of the pData). By default, the first covariate appearing in the colnames of the pData is used.

Similarly, there is a couple different ways how to specify which experimental condition to treat as the baseline. The available 'condition names' are taken from `unique(as.character(pData(data)[, covariate]))`. If `baselineCondition` is given as a `character`, it will be taken as the condition name itself. If `baselineCondition` is given as an `integer` or a `numeric` value, it will be taken as the condition number (in the same order of appearance as in the 'condition names'). By default, the first condition name is used.

The model is fitted on the most replicated condition. When more conditions exist with the maximum number of replicates, the condition providing the best fit is chosen (based on the adjusted r^2). If there is again a tie, the first one is arbitrarily taken.

If less than 3 replicates are provided for the condition used for fitting, then the selection is based on ranking according to the observed PLGEM-STN values. In this case the first `rank` genes or proteins are selected for each comparison.

Otherwise DEG are selected comparing the observed and resampled PLGEM-STN values at the `signLev` significance level(s), based on p-values obtained via a call to function `plgem.pValue`. See References for details.

Value

A `list` of four elements:

`fit`	the input `plgemFit`.
`PLGEM.STN`	a `matrix` of observed PLGEM-STN values (see `plgem.obsStn` for details).
`p-value`	a `matrix` of p-values (see `plgem.pValue` for details).
`significant`	a `list` with a number of elements equal to the number of different significance levels (`delta`) used as input. If ranking method is used due to insufficient number of replicates (see Details), this list will be of length 1 and named `firstXXX`, where `XXX` is the number provided by argument `rank`. Each element of this list is again a list, whose number of elements correspond to the number of performed comparisons (i.e. the number of conditions in the starting `ExpressionSet` minus the baseline). Each of these second level elements is a `character` vector of significant gene/protein names that passed the statistical test at the corresponding significance level.

Author(s)

Mattia Pelizzola <mattia.pelizzola@gmail.com>

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17; 5:203; http://www.biomedcentral.com/1471-2105/5/203.

Pavelka N, Fournier ML, Swanson SK, Pelizzola M, Ricciardi-Castagnoli P, Florens L, Washburn MP. Statistical similarities between transcriptomics and quantitative shotgun proteomics data. Mol Cell Proteomics. 2008 Apr; 7(4):631-44; http://www.mcponline.org/cgi/content/abstract/7/4/631.

See Also

plgem.fit, plgem.obsStn, plgem.resampledStn, plgem.pValue, plgem.deg, plgem.write.summary

Examples

```
data(LPSeset)
set.seed(123)
LPSdegList <- run.plgem(esdata=LPSeset, fitting.eval=FALSE)
```

---

setGpar *Set graphical parameters for PLGEM fitting evaluation plots*

Description

Function to set graphical parameters for **PLGEM** fitting evaluation plots produced by function plgem.fit.

Usage

```
setGpar(minLnM=NULL, maxLnM=NULL, minLnSD=NULL, maxLnSD=NULL,
  minRes=NULL, maxRes=NULL)
```

Arguments

minLnM	minimum 'ln(mean)' value in upper left plot.
maxLnM	maximum 'ln(mean)' value in upper left plot.
minLnSD	minimum 'ln(sd)' value in upper left plot.
maxLnSD	maximum 'ln(sd)' value in upper left plot.
minRes	minimum 'residual' value in upper right plot.
maxRes	maximum 'residual' value in upper right plot.

Details

A call to setGpar() is the recommended way to set graphical parameters in plgem.fit. If parameters are left unspecified, suitable defaults will be found by plgem.fit.

Value

A list of six elements:

minLnM	minimum 'ln(mean)' value in upper left plot.
maxLnM	maximum 'ln(mean)' value in upper left plot.
minLnSD	minimum 'ln(sd)' value in upper left plot.
maxLnSD	maximum 'ln(sd)' value in upper left plot.
minRes	minimum 'residual' value in upper right plot.
maxRes	maximum 'residual' value in upper right plot.

Author(s)

Norman Pavelka <normanpavelka@gmail.com>

References

Pavelka N, Pelizzola M, Vizzardelli C, Capozzoli M, Splendiani A, Granucci F, Ricciardi-Castagnoli P. A power law global error model for the identification of differentially expressed genes in microarray data. BMC Bioinformatics. 2004 Dec 17; 5:203; http://www.biomedcentral.com/1471-2105/5/203.

Pavelka N, Fournier ML, Swanson SK, Pelizzola M, Ricciardi-Castagnoli P, Florens L, Washburn MP. Statistical similarities between transcriptomics and quantitative shotgun proteomics data. Mol Cell Proteomics. 2008 Apr; 7(4):631-44; http://www.mcponline.org/cgi/content/abstract/7/4/631.

See Also

plgem.fit, run.plgem

Examples

```
setGpar(minLnM=-1, maxLnM=8)
```

Index