Package ‘GESTr’

February 19, 2015

Version 0.1

Date 2013-02-21

Title Gene Expression State Transformation

Author Ed Curry

Maintainer Ed Curry <e.curry@imperial.ac.uk>

Depends R (>= 2.15.0), mclust, gtools

Description The Gene Expression State Transformation (GESTr) models the states of expression of genes across a compendium of samples in order to provide a universal scale of gene expression for all genes. TranSAM is a modification of the SAM approach designed to utilise GESTr-transformed gene expression data.

License GPL (>= 2)

URL http://www.r-project.org, http://www1.imperial.ac.uk/medicine/people/e.curry/

NeedsCompilation no

Repository CRAN

Date/Publication 2013-03-07 13:16:24

R topics documented:

ABIdata *Gene Expression Data Matrix*

Description

93 ABI Human Genome Survey Microarray v2 samples, from a survey of gene expression in 31 different tissues

Usage

```
data(GESTr)
```

Format

Numeric matrix containing log2 normalised expression data for first 1000 genes across 93 samples.

Source

http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE7905

findReplicateGroups *Find Groups of Replicate Samples Within Dataset*

Description

Function to find groups of entities within a distance matrix that are similar to some statistical threshold

Usage

```
findReplicateGroups(distMatrix,theta=0.05)
```

Arguments

`distMatrix` A numeric matrix containing pair-wise distances between each sample of the datset.

`theta` A numeric value specifying the quantile of the Normal distribution (approximating the distribution of pairwise distances) to be used to call samples suitable similar to be considered replicates.

Details

Approximates all pair-wise distances with a Normal distribution, then find each group of samples for which all distances are less than the specified quantile of the approximated distribution.

Value

List with a separate element for each replicate group: the individual groups are represented by a vector of corresponding sample indices.

Author(s)

Ed Curry <`e.curry@imperial.ac.uk`>

Examples

```
### internal function ###
```

`fitGMM` *Fitting Gaussian Mixture Models*

Description

Fit Gaussian Mixture Model to a vector of gene expression data values.

Usage

```
fitGMM(exprVals,RDparameters,rejectNull=0.05)
```

Arguments

`exprVals`	A numeric vector of expression values for one gene
`RDparameters`	A list of Rocke-Durbin error model parameters, generated by the `fitRockeDurbin` function
`rejectNull`	Numeric value specifying significance level for rejecting null hypothesis that the expression values arise from a single value and expected errors

Details

Uses `Mclust` to fit a Gaussian Mixture Model (with an unspecified number of components) to a vector of expression values for one gene across a dataset. Required to create models used by `GESTr`.

Value

An object of class `Mclust`

Author(s)

Ed Curry <`e.curry@imperial.ac.uk`>

Examples

```
### internal function ###
```

`fitRockeDurbin` *Fit Rocke-Durbin Error Model*

Description

Estimates Rocke-Durbin error model parameters for a gene expression dataset.

Usage

```
fitRockeDurbin(x,theta)
```

Arguments

`x` Numeric data input array to be transformed into universal gene expression 'state' scale. Probes/genes should be in rows, samples/conditions in columns.

`theta` Numeric value specifying Normal distribution quantile for identifying groups samples similar enough to be treated as replicates

Details

Estimates error model parameters for Rocke-Durbin model of microarray measurement errors. The model comprises three parameters to be estimated: the background measurement level, an intensity-dependent error term and an intensity-independent error term. This function finds groups of suitably-similar samples from the dataset to be treated as replicates, and estimates model parameters from constitutively-low and constitutively-high measurements.

Value

A list with 3 named components:

`alpha`	background measurement
`sd_epsilon`	standard deviation of intensity-dependent errors
`sd_eta`	standard deviation of intensity-independent errors

Author(s)

Ed Curry <e.curry@imperial.ac.uk>

Examples

```
## Not run: data(GESTr)
## Not run: RDparameters <- fitRockeDurbin(ABIdata,theta=0.05)
```

GESTr *Gene Expression State Transformation*

Description

Implements the Gene Expression State Transformation (GESTr), a means of modelling the states of expression of genes across a compendium of samples in order to provide a universal scale of gene expression for all genes.

Usage

```
GESTr(x,dist.theta=0.05,merge.overlap=0.1,verbose=FALSE)
```

Arguments

`x` Numeric data input array to be transformed into universal gene expression 'state' scale. Probes/genes should be in rows, samples/conditions in columns.

`dist.theta` Numeric value indicating normal distribution quantile to be used for calling replicate group similarity, used in estimating Rocke-Durbin error model parameters.

`merge.overlap` Numeric value indicating minimum proportion of overlapping support of GMM components to be merged together

`verbose` Boolean indicating whether to provide progress reports during procedure.

Details

Implementation of the Gene Expression State Transformation. The Gene Expression State Transformation (GESTr) is a process by which structural components are identified within the distributions of measurements for each gene across a data compendium, and are then used to transform the expression level measurements into a standardised scale. Any value in this scale has the same biological interpretation, regardless of the gene, and reflects the state of expression in the sample as defined by the levels observed across the compendium. Each gene's expression level distribution across the compendium is modelled using a Gaussian Mixture Model (GMM), without specifying the number of components a priori. Components with substantially overlapping support are merged, so that the final model components reflect distinct states of expression. A classifier is constructed to compute probabilities that any observed measurement arose from each underlying expression state, and finally probabilities of expression-state membership are combined into a linear scale representing the probability of a highly-active transcriptional state of the gene.

Value

Transformed representation of `x`, such that all values lie in range (0,1) and represent the probability of a highly-active transcriptional state of the gene relative to its distribution across the whole dataset.

Author(s)

Ed Curry `<e.curry@imperial.ac.uk>`

Examples

```
## Not run: data(GESTr)
## Not run: transformed.x <- GESTr(ABIdata)
```

getMonotonicConfidences

Ensure monotonicity of two-class gene expression state confidence assignments

Description

Ensures monotonicity of a gene's expression state confidence assignments, in terms of the underlying gene expression values.

Usage

```
getMonotonicConfidences(ClassScoreList)
```

Arguments

`ClassScoreList`	List with 2 elements, each a numeric vector of probabilities of state membership. One (vector) element for low-expression state and one for high-expression state.

Details

Processes two vectors of class membership scores, returning values corrected so as to ensure monotonicity.

Value

List with 2 elements, each a numeric vector of probabilities of state membership. One (vector) element for low-expression state and one for high-expression state.

Author(s)

Ed Curry <e.curry@imperial.ac.uk>

Examples

```
### internal function ###
```

getMonotonicConfidences_multiclass

Ensure monotonicity of multi-class gene expression state confidence assignments

Description

Ensures monotonicity of a gene's expression state confidence assignments, in terms of the underlying gene expression values.

Usage

```
getMonotonicConfidences_multiclass(ClassScoreList,exprs,verbose=FALSE)
```

Arguments

`ClassScoreList`	List where each element is a numeric vector of probabilities of state membership. One (vector) element per state in the model.
`exprs`	A numeric vector of expression values for one gene
`verbose`	Logical value specifying whether to print output when correction required.

Details

Processes two vectors of class membership scores, returning values corrected so as to ensure monotonicity.

Value

List with 2 elements, each a numeric vector of probabilities of state membership. One (vector) element for low-expression state and one for high-expression state.

Author(s)

Ed Curry <e.curry@imperial.ac.uk>

Examples

```
### internal function ###
```

mergeComponents *Merge GMM Components with Highly Overlapping Support*

Description

Function to merge GMM components with predominantly overlapping support

Usage

```
mergeComponents(model,overlap)
```

Arguments

`model`	An object of class `Mclust`, produced by the function `fitGMM`
`overlap`	A numeric value specifying the classification score threshold: minimum value of `model$z` for a value to be considered as supported by a given mixture component.

Details

Identifies GMM components with sufficiently overlapping support, if for those components the average E-M classification score across all points with a maximum score corresponding to any of those components is greater than the specified threshold.

Value

List where each element is a vector of indices specifying components from input `model` to merge together in gene expression state estimation.

Author(s)

Ed Curry <e.curry@imperial.ac.uk>

Examples

```
### internal function ###
```

TranSAM — *Gene Expression State Transformed Significance Analysis of Microarrays*

Description

Implements TranSAM, a method to use the GESTr-transformed representation of gene expression data to identify genes with _biologically_ significant variation: that is, statistically significant differential expression across biologically distinct states of expression observed in the compendium used for reference (calculating the GESTr models).

Usage

```
TranSAM(x,samples1,samples2,minChange=0.2,var_filter=0.01,maxFDR=1,changeStep=0.1,scoreFun="magChan
```

Arguments

`x` Numeric data array of transformed gene expression 'state' values. Output from `GESTr` function. Probes/genes should be in rows, samples/conditions in columns.

`samples1` Numeric vector of indices for first group of samples

`samples2` Numeric vector of indices for second group of samples

`minChange` Numeric value specifying minimum value of the observed difference between the groups, compared to the expected difference as estimated from the balanced permutations

`var_filter` Numeric value specifying a minimum standard deviation in gene expression state values for a gene to be included in the analysis

`maxFDR` Numeric value specifying maximum allowed group-wise False Discovery Rate, the function will iterate over successively greater minimum observed differences until estimated FDR is below maxFDR

`changeStep` Numeric value specifying step-wise increase of `minChange` filter at each iteration

`scoreFun` Character specifying method of scoring. `"dstat"` uses a regularized t-statistic, making it an analogue of the Significance Analysis of Microarrays (SAM) approach. Any other value uses the absolute difference between the median expression state value of the gene in question across the two groups.

Details

The TranSAM algorithm constructs balanced permutations of the input data and uses these to estimate the false-discovery rates of identifying genes as belonging to different expression states in the two specified sample groups. The balanced permutations are constructed so that an equal number of samples from each specified group are in each partition, and thus can be used to approximate a distribution of expected variation in gene expression state across the groups if the specified grouping were to have no biological relevance (in terms of gene expression profiles).

Value

A Data Frame with columns:\

`genes` The rownames of input `x` corresponding to the genes with significant differential expression between the specified group

`obs.exp.ratios` The calculated scoring statistic for differential expression (in terms of observed value compared to expected)

`change` The difference in median gene expression state values for the gene across the two groups

`FDR.estimate` Estimated Family-Wise Error Rate (FWER) across all genes at least as differentially expressed as the selected gene. This is analogous to FDRor the q-value

Author(s)

Ed Curry `<e.curry@imperial.ac.uk>`

Examples

```
## load data and run GESTr on a subset of this to create transformed data
data(GESTr)
selected.columns <- sort(c(sample(1:ncol(ABIdata),30),which(colnames(ABIdata) %in% c("GSM194513","GSM194514","G
transformed.x <- GESTr(ABIdata[1:20,selected.columns])

## choose samples for analysis
thy.adult <- which(colnames(transformed.x) %in% c("GSM194513","GSM194514","GSM194515"))
thy.fetal <- which(colnames(transformed.x) %in% c("GSM194516","GSM194517","GSM194518"))

## run TranSAM on selected samples
ts.out <- TranSAM(transformed.x[,c(thy.adult,thy.fetal)],samples1=1:3,samples2=4:6)
```

Index