frma

October 25, 2011

GNUSE *GNUSE*

Description

Computes the generalized normalized unscaled standard error (a measure of microarray quality).

Usage

```
GNUSE(object, medianSE=NULL, type=c("plot", "values", "stats", "density"), ...
```

Arguments

`object`	an ExpressionSet or frmaExpressionSet containing standard errors produced by frma
`medianSE`	median standard errors to be used. If NULL, these are obtained from the correct frmavecs package.
`type`	the desired output
`...`	additional graphical parameters for types plot or density

Value

If type is plot, boxplots of GNUSE values are displayed. If type is values, the GNUSE values are returned. If type is stats, the median, IQR, 95th, and 99th percentiles are reported. If type is density, a density plots of GNUSE values are displayed.

Author(s)

Matthew N. McCall

Examples

```
library(frma)
library(frmaExampleData)
data(AffyBatchExample)
object <- frma(AffyBatchExample)
GNUSE(object, type="stats")
```

`barcode` *Gene Expression Barcode*

Description

This function converts expression values produced via fRMA to a gene expression barcode.

Usage

```
barcode(object, platform=NULL, mu=NULL, tau=NULL, cutoff=6.5, output="binary")
```

Arguments

`object` a vector or matrix of expression values or an ExpressionSet or frmaExpressionSet produced by frma

`platform` the platform of the input data. One of GPL96, GPL570, GPL1261. Required if object is a vector or matrix.

`mu` the mean of the unexpressed distribution. If NULL then precomputed values are used.

`tau` the standard deviation of the unexpressed distribution. If NULL then precomputed values are used.

`cutoff` the lod score cutoff used if output is binary.

`output` the desired values to be returned. Options are: p-value, z-score, lod, or binary.

Value

A matrix containing the type of output specified by the output parameter. The option *binary* creates a gene expression barcode where 1s denote expressed genes and 0s denote unexpressed genes. The option *p-value* returns the p-values for the expression values under the unexpressed distribution. The option *lod* returns the LOD scores for expression values under the unexpressed distribution. The option *z-score* returns the z-scores for the expression values under the unexpressed distribution.

Author(s)

Matthew N. McCall

Examples

```
library(frma)
library(frmaExampleData)
data(AffyBatchExample)
object <- frma(AffyBatchExample)
bc <- barcode(object)
```

`frma` *Frozen Robust Multi-Array Analysis*

Description

This function preprocesses an AffyBatch or ExonFeatureSet object using the fRMA method.

Usage

```
frma(object, background="rma", normalize="quantile",
    summarize="robust_weighted_average", target="core",
    input.vecs=list(normVec=NULL, probeVec=NULL, probeVarBetween=NULL,
    probeVarWithin=NULL, probesetSD=NULL), output.param=NULL,
    verbose=FALSE)
```

Arguments

`object` an AffyBatch

`background` type of background correction to perform: either "none" or "rma".

`normalize` type of normalization to perform: either "none" or "quantile".

`summarize` type of summarization to perform: one of "median_polish", "average", "median", "weighted_average", "robust_weighted_average", "random_effect".

`target` summarization level for exon arrays. Must be one of: probeset, core, full, extended.

`input.vecs` a list of vectors to be used in preprocessing. If necessary elements are NULL, the correct package with the default vectors is loaded if it has been installed. These packages are of the form: <platform>frmavecs.

`output.param` a vector of output elements to return. By default only the expression values and standard errors (if applicable) are returned. Additional options are: "residuals", "weights", "random_effects"

`verbose` logical value. If TRUE then some messages are displayed while the function runs.

Value

The function returns an ExpressionSet if output.param=NULL or an frmaExpressionSet otherwise.

Author(s)

Matthew N. McCall

Examples

```
library(frmaExampleData)
data(AffyBatchExample)
object <- frma(AffyBatchExample)
```

frmaExpressionSet-class *Class to Contain and Describe High-Throughput Expression Level Assays*

Description

This is a class representation for fRMA-preprocessed expression data. `frmaExpressionSet` class is derived from `ExpressionSet`, and requires a matrix named `exprs` and optionally matrices named `se.exprs`, `weights`, and `residuals`.

Extends

Extends class `ExpressionSet`.

Creating Objects

```
new("frmaExpressionSet", phenoData = new("AnnotatedDataFrame"), featureData
= new("AnnotatedDataFrame"), experimentData = new("MIAME"), annotation
= character(0), exprs = new("matrix"), weights = new("matrix"), residuals
= new("matrix"), randomeffects = new("matrix")
```

This creates a `frmaExpressionSet` with `assayData` implicitly created to contain `exprs`. The only required named argument is `exprs`. Three optional named matrices, `weights`, `residuals`, and `randomeffects` can be added to the object.

```
new("frmaExpressionSet", assayData = assayDataNew(exprs=new("matrix"),se.exprs=n
phenoData = new("AnnotatedDataFrame"), featureData = new("AnnotatedDataFrame"),
experimentData = new("MIAME"), annotation = character(0), weights =
new("matrix"), residuals = new("matrix"), randomeffects = new("matrix")
```

This creates a `frmaExpressionSet` with `assayData` provided explicitly. In this form, the only required named argument is `assayData`. Three optional named matrices, `weights`, `residuals`, and `randomeffects` can be added to the object.

Slots

`weights`: weights used in the summarization step

`residuals`: residuals from fitting the probe-level model

`randomeffects`: random effect estimates from fitting the probe-level model using random effect summarization

Inherited from `ExpressionSet`:

`assayData`: Contains matrices with equal dimensions, and with column number equal to `nrow(phenoData)`. `assayData` must contain a matrix `exprs` with rows representing features and columns representing samples. It may also contain a matrix `se.exprs` containing standard errors.

`phenoData`: See `eSet`

`annotation`: See `eSet`

`featureData`: See `eSet`

`experimentData`: See `eSet`

Methods

Class-specific methods:

`se.exprs(frmaExpressionSet)` Access elements named `se.exprs` in the `AssayData-class` slot.

`weights(frmaExpressionSet)` Access elements named `weights`

`residuals(frmaExpressionSet)` Access elements named `residuals`

`randomeffects(frmaExpressionSet)` Access elements named `randomeffects`

For derived methods (see `ExpressionSet`).

See Also

`eSet-class`, `ExpressionSet-class`, `frma`.

Examples

```
# create an instance of frmaExpressionSet
new("frmaExpressionSet")
```

Index