Package ‘antiProfiles’

May 29, 2024

Maintainer  Hector Corrada Bravo <hcorrada@gmail.com>
Author       Hector Corrada Bravo, Rafael A. Irizarry and Jeffrey T. Leek
Version      1.44.0
License      Artistic-2.0
Title        Implementation of gene expression anti-profiles
Description  Implements gene expression anti-profiles as described in Corrada
Depends      R (>= 3.0), matrixStats (>= 0.50.0), methods (>= 2.14), locfit
             (>= 1.5)
URL          https://github.com/HCBravoLab/antiProfiles
biocViews    GeneExpression,Classification
Suggests     antiProfilesData, RColorBrewer
Collate      'AllClasses.r' 'AllGenerics.r' 'show-methods.r'
             'antiProfiles-package.r' 'ap-methods.r' 'apCounts-methods.r'
             'apReorder-methods.r' 'apStats.r' 'accessors.r' 'ev.R'
RoxygenNote  5.0.1

Contents

antiProfiles-package ........................................ 2
AntiProfile-class ........................................... 2
AntiProfileStats-class ...................................... 3
apCount ....................................................... 4
apReorder ..................................................... 5
### Description


### Author(s)

Hector Corrada Bravo <hcorrada@gmail.com>

### AntiProfile-class

A gene expression anti-profile

### Description

This class encapsulates anti-profiles. Objects of this class should be built from `AntiProfileStats` objects using the `buildAntiProfile` method. Anti-profile scores can be computed for new samples using the `apCount` method.

### Usage

#### ## Accessors

```r
getProbesetIds(object)
getNormalRegions(object)
```

### Arguments

- `object` Object of class (or inheriting from) `AntiProfile`

### Accessors

In the following code `object` is a `AntiProfile` object.

- `getProbesetIds`: vector of Affymetrix ids for probesets included in the anti-profile (character)
- `getNormalRegions`: median and upper boundary of normal expression regions (numeric matrix of dimension s-by-2, where s is the size of the anti-profile)
Author(s)

Hector Corrada Bravo <hcorrada@gmail.com>

See Also

AntiProfileStats for the class of objects from which anti-profiles are built. buildAntiProfile for the method used to construct objects of this class. apCount for the function used to calculate anti-profile scores from objects of this class.

Examples

```r
if (require(antiProfilesData)) {
  data(apColonData)
  colonStats = apStats(exprs(apColonData), pData(apColonData)$Status)
  colonAP = buildAntiProfile(colonStats, tissueSpec=FALSE, sigsize=10)
  show(colonAP)

  head(getProbesetIds(colonAP))
  head(getNormalRegions(colonAP))
}
```

AntiProfileStats-class

Statistics used to build anti-profiles

Description

This class stores statistics required to build anti-profiles. Objects of this class should be built using the `apStats` function. To build anti-profiles from objects of this class, use the `buildAntiProfile` function.

Usage

```r
## Accessors
getProbeStats(object)
```

Arguments

- **object**: Object of class AntiProfileStats

Accessors

In the following code `object` is a AntiProfileStats object.

getProbeStats: Gets probeset statistics. A data.frame with columns

- **affyid**: Affymetrix probeset id (character)
- **SD0**: Normal expression standard deviation aggregated over tissue types (numeric)
- **SD1**: Cancer expression standard deviation aggregated over tumor types (numeric)
apCount

stat: The log2-variance ratio statistic (numeric)
meds0: Median normal expression aggregated over tissue types (numeric)
mads0: Median absolute deviation of normal expression aggregate over tissue types (numeric)

Author(s)

Hector Corrada Bravo <hcorrada@gmail.com>

See Also

apStats to construct objects of this class, buildAntiProfile to build anti-profiles from objects of this class.

Examples

if (require(antiProfilesData)) {
  data(apColonData)
  colonStats = apStats(exprs(apColonData), pData(apColonData)$Status)
  show(getProbeStats(colonStats))
}

---

### apCount

*Obtain the anti-profile score for a set of samples*

Description

This function applies the given anti-profile to a new set of samples. Rownames in the expression matrix are used to match probenames in the AntiProfile object.

Arguments

- **fit**: an object of class AntiProfile as produced by the buildAntiProfile method
- **expr**: a matrix of gene expression, rownames are used as identifiers

Value

A numeric vector of anti-profile scores

Author(s)

Hector Corrada Bravo <hcorrada@gmail.com>
Examples

```r
if (require(antiProfilesData)) {
  data(apColonData)

  # compute statistics
  colonStats = apStats(exprs(apColonData), pData(apColonData)$Status)

  # create an anti-profile, ignoring tissue-specificity of probesets, with 10 probesets
  ap = buildAntiProfile(colonStats, tissueSpec=FALSE, sigsize=10)

  # get counts for the original dataset
  counts = apCount(ap, exprs(apColonData))
}
```

---

**apReorder**  
*Reorder an AntiProfileStats object*

**Description**

Reorders given AntiProfileStats object using provided ordering o

**Arguments**

- **stats**  
  An object of class AntiProfileStats

- **o**  
  A numeric vector giving new probe ordering

**Value**

A reordered AntiProfileStats object

**Author(s)**

Hector Corrada Bravo <hcorrada@gmail.com>

**Examples**

```r
if (require(antiProfilesData)) {
  data(apColonData)
  colonStats = apStats(exprs(apColonData), pData(apColonData)$Status)
  o = sample(seq(len=nrow(slot(colonStats,"probes"))))
  newStats = apReorder(colonStats, o)
}
```
apStats

Compute statistics used to construct antiProfile

Description

This function calculates normal ranges of expressions and variance ratios for all probesets. To create an anti-profile, call buildAntiProfile on the output of this function.

Usage

```r
apStats(e, cl, tiss = NULL, minL = 10, cutoff = 5,
        OnCutoff = 2.54)
```

Arguments

- `e`: matrix of gene expression, with one column per sample
- `cl`: vector of normal/cancer indicators as 0/1
- `tiss`: vector of tissue types for each sample
- `minL`: minimum number of samples of a given tissue/class to compute stats
- `cutoff`: median absolute deviation multiplier used to determine proportion of samples within normal range of expression
- `OnCutoff`: gene expression barcode z-score to determine if a gene is expressed

Value

An object of class SuccsStats

Author(s)

Hector Corrada Bravo <hcorrada@gmail.com>

See Also

AntiProfileStats for the type of object returned. buildAntiProfile to construct anti-profiles with objects returned by this function.

Examples

```r
if (require(antiProfilesData)) {
  data(apColonData)
  colonStats = apStats(exprs(apColonData), pData(apColonData)$Status)
}
```
buildAntiProfile

Create an anti-profile from a AntiProfileStats object

Description
This function creates anti-profile using statistics stored in a AntiProfileStats object.

Arguments
- stats: an object of class AntiProfileStats as produced by the apStats function.
- tissueSpec: use tissue-specific regions of normal expression.
- tissueFilter: use only tissue-specific genes in the anti-profile.
- sigsize: desired size of signature, if NULL, computed from statCutoff.
- cutoff: median absolute deviation multiplier used to define normal regions of expression.
- statCutoff: cutoff used to include probesets in anti-profile.

Value
an object of class AntiProfilesSig or AntiProfilesTissueSig depending on the tissueSpec argument.

Author(s)
Hector Corrada Bravo <hcorrada@gmail.com>

Examples
if (require(antiProfilesData)) {
  # create an anti-profile, ignoring tissue-specificity of probesets, with 10 probesets
  data(apColonData)
  colonStats = apStats(exprs(apColonData), pData(apColonData)$Status)
  ap = buildAntiProfile(colonStats, tissueSpec=FALSE, sigsize=10)
}

ev

Compute expression variability measure

Description
This function computes expression variability in a way that removes dependence on mean expression. It uses a local polynomial likelihood method to estimate variance as gamma distributed around given mean expression for each probeset. This function makes this calculation using all samples in argument. To calculate expression variability for samples in different groups, call this function for each subset of columns separately.
Usage

`ev(x, cutoff = NULL, plot = FALSE, ...)`

Arguments

- `x`: matrix of gene expression, with one column per sample
- `cutoff`: minimum expression value to be included in computation (for frma normalized data, we find 2.54 to be a good value for determining if a probeset is expressed in a given sample (default NULL))
- `plot`: make a plot of local likelihood model using smoothScatter (default=FALSE)
- `...`: arguments passed to smoothScatter

Value

numeric vector of length equal to number of rows of `x`

Author(s)

Hector Corrada Bravo <hcorrada@gmail.com>

References


See Also

frma for normalization

Examples

```r
if (require(antiProfilesData)) {
  data(apColonData)
  e <- exprs(apColonData)[,pData(apColonData)$Status==1]
  ev <- ev(e, cutoff=2.54)
}
```
Usage

## Accessors

## S4 method for signature 'TissueSpecAntiProfile'
getProbesetIds(object)
## S4 method for signature 'TissueSpecAntiProfile'
getNormalRegions(object)
getNormalTissueRegions(object)

Arguments

object Object of class TissueSpecAntiProfile

Accessors

In the following code object is a TissueSpecAntiProfile object.

getProbesetIds: vector of Affymetrix ids for probesets included in the anti-profile (character)
getNormalRegions: median and upper boundary of normal expression regions (numeric matrix of
dimension s-by-2, where s is the size of the anti-profile)
getNormalTissueRegions: median and upper boundary of normal expression regions (numeric
array of dimension s-by-2-by-t, where s is the size of the anti-profile, and t the number of
normal tissues used in the anti-profile)

Author(s)

Hector Corrada Bravo <hcorrada@gmail.com>

See Also

AntiProfileStats for the class of objects from which anti-profiles are built. buildAntiProfile
for the method used to construct objects of this class. apCount for the function used to calculate
anti-profile scores from objects of this class. #'

Examples

if (require(antiProfilesData)) {
  data(apColonData)
  # fake tissues
tissue=rep(c("colon","lung"), len=length(sampleNames(apColonData)))
tissStats=apStats(exprs(apColonData), pData(apColonData)$Status, tiss=tissue, minL=3)
tissAP=buildAntiProfile(tissStats, sigsize=10)
  show(tissAP)

  head(getProbesetIds(tissAP))
  head(getNormalRegions(tissAP))
  head(getNormalTissueRegions(tissAP))
}
Index

AntiProfile, 2
AntiProfile-class, 2
antiProfiles-package, 2
AntiProfileStats, 2, 3, 6, 8, 9
AntiProfileStats-class, 3
apCount, 2, 3, 4, 8, 9
apCount,AntiProfile,matrix-method
(apCount), 4
apCount,TissueSpecAntiProfile,matrix-method
(apCount), 4
apReorder, 5
apReorder,AntiProfileStats,numeric-method
(apReorder), 5
apStats, 3, 4, 6
buildAntiProfile, 2–4, 6, 7, 8, 9
buildAntiProfile,AntiProfileStats-method
(buildAntiProfile), 7

ev, 7

getNormalRegions (AntiProfile-class), 2
getNormalRegions,AntiProfile-method
(AntiProfile-class), 2
getNormalRegions,TissueSpecAntiProfile-method
(TissueSpecAntiProfile-class), 8
getNormalTissueRegions
(TissueSpecAntiProfile-class), 8
getNormalTissueRegions,TissueSpecAntiProfile-method
(TissueSpecAntiProfile-class), 8
getProbesetIds (AntiProfile-class), 2
getProbesetIds,AntiProfile-method
(AntiProfile-class), 2
getProbesetIds,TissueSpecAntiProfile-method
(TissueSpecAntiProfile-class), 8
getProbeStats (AntiProfileStats-class), 3
getProbeStats,AntiProfileStats-method
(AntiProfileStats-class), 3
TissueSpecAntiProfile, 9
TissueSpecAntiProfile-class, 8