

# Package ‘frma’

April 15, 2020

**Version** 1.38.0

**Date** 2017-11-08

**Title** Frozen RMA and Barcode

**Description** Preprocessing and analysis for single microarrays and microarray batches.

**Author** Matthew N. McCall <mccallm@gmail.com>, Rafael A. Irizarry <rafa@jhu.edu>, with contributions from Terry Therneau

**Maintainer** Matthew N. McCall <mccallm@gmail.com>

**Depends** R (>= 2.10.0), Biobase (>= 2.6.0)

**Imports** Biobase, MASS, DBI, affy, methods, oligo, oligoClasses, preprocessCore, utils, BiocGenerics

**Suggests** hgu133afmavecs, frmaExampleData

**biocViews** Software, Microarray, Preprocessing

**License** GPL (>= 2)

**URL** <http://bioconductor.org>

**git\_url** <https://git.bioconductor.org/packages/frma>

**git\_branch** RELEASE\_3\_10

**git\_last\_commit** efeb489

**git\_last\_commit\_date** 2019-10-29

**Date/Publication** 2020-04-14

## R topics documented:

barcode . . . . .	2
frma . . . . .	3
frmaExpressionSet-class . . . . .	4
GNUSE . . . . .	5

<b>Index</b>	<b>7</b>
--------------	----------

---

`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**

<code>object</code>	a vector or matrix of expression values or an ExpressionSet or frmaExpressionSet produced by frma
<code>platform</code>	the platform of the input data. One of GPL96, GPL570, GPL571, GPL1261, GPL6244, GPL6246. Required if object is a vector or matrix and either mu or tau is NULL.
<code>mu</code>	the mean of the unexpressed distribution. If NULL then precomputed values are used if possible.
<code>tau</code>	the standard deviation of the unexpressed distribution. If NULL then precomputed values are used if possible.
<code>cutoff</code>	the lod score cutoff used if output is binary.
<code>output</code>	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, ExonFeatureSet, or GeneFeatureSet object using the fRMA method.

## Usage

```
frma(object, background="rma", normalize="quantile",
      summarize="robust_weighted_average", target="probeset",
      input.vecs=NULL, output.param=NULL, verbose=FALSE)
```

## Arguments

object	an AffyBatch, ExonFeatureSet, or GeneFeatureSet
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 and gene arrays. Must be one of: probeset, core, full (exon only), extended (exon only).
input.vecs	a list of vectors to be used in preprocessing. If NULL, the correct package with pre-made 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 preprocessed with fRMA*

---

## 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", exprs = new("matrix"), se.exprs = new("matrix"), weights=new("matrix"), residuals=new("matrix"),
    = new("AnnotatedDataFrame"), featureData = new("AnnotatedDataFrame"), experimentData = new("MIAME"), annotation = new("character"), ...)
```

This creates a frmaExpressionSet with assayData implicitly created to contain `exprs` and `se.exprs`. The only required named arguments 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=new("matrix")), weights=new("matrix"),
    = new("AnnotatedDataFrame"), featureData = new("AnnotatedDataFrame"), experimentData = new("MIAME"), annotation = new("character"), ...)
```

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

`se.exprs`: standard errors for the expression estimates

`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")
```

---

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**

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

**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")
```

# Index

## \*Topic **classes**

frmaExpressionSet-class, 4

## \*Topic **manip**

barcode, 2

frma, 3

GNUSE, 5

as.ExpressionSet, frmaExpressionSet-method  
(frmaExpressionSet-class), 4

barcode, 2

class:frmaExpressionSet  
(frmaExpressionSet-class), 4

frma, 3

frmaExpressionSet  
(frmaExpressionSet-class), 4

frmaExpressionSet-class, 4

frmaExpressionSet-methods  
(frmaExpressionSet-class), 4

GNUSE, 5

initialize, frmaExpressionSet-method  
(frmaExpressionSet-class), 4

randomeffects  
(frmaExpressionSet-class), 4

randomeffects, frmaExpressionSet-method  
(frmaExpressionSet-class), 4

randomeffects<-  
(frmaExpressionSet-class), 4

randomeffects<-, frmaExpressionSet-method  
(frmaExpressionSet-class), 4

residuals, frmaExpressionSet-method  
(frmaExpressionSet-class), 4

residuals<- (frmaExpressionSet-class), 4

residuals<-, frmaExpressionSet-method  
(frmaExpressionSet-class), 4

se.exprs, ExpressionSet-method  
(frmaExpressionSet-class), 4

se.exprs, frmaExpressionSet-method  
(frmaExpressionSet-class), 4

se.exprs<- (frmaExpressionSet-class), 4

se.exprs<-, ExpressionSet-method

(frmaExpressionSet-class), 4

se.exprs<-, frmaExpressionSet-method

(frmaExpressionSet-class), 4

weights, frmaExpressionSet-method  
(frmaExpressionSet-class), 4

weights<- (frmaExpressionSet-class), 4

weights<-, frmaExpressionSet-method  
(frmaExpressionSet-class), 4