

# oligo

April 20, 2011

---

basecontent                      *Sequence Base Contents*

---

## Description

Function to compute the amounts of each nucleotide in a sequence.

## Usage

```
basecontent(seq)
```

## Arguments

seq                      character vector of length n containing a valid sequence (A/T/C/G)

## Value

matrix with n rows and 4 columns with the counts for each base.

## Examples

```
sequences <- c("ATATATCCCCG", "TTCCGAGC")
basecontent(sequences)
```

---

basicRMA                      *Simplified Interface to RMA*

---

## Description

Simple interface to RMA.

## Usage

```
basicRMA(pmMat, pnVec, normalize = TRUE, background = TRUE, bgversion = 2, destr
```

**Arguments**

pmMat	Matrix of intensities to be processed.
pnVec	Probeset names.
normalize	Logical flag: normalize?
background	Logical flag: background adjustment?
bgversion	Version of background correction.
destructive	Logical flag: use destructive methods?
verbose	Logical flag: verbose.
...	Not currently used.

**Value**

Matrix.

**Examples**

```
set.seed(1)
pms <- matrix(rnorm(1000), nc=20)
colnames(pms) <- paste("sample", 1:20, sep="")
pns <- rep(letters[1:10], each=5)
res <- basicRMA(pms, pns, length(unique(pns)), TRUE, TRUE)
res[, 1:3]
```

---

boxplot-methods      *Boxplot*

---

**Description**

Boxplot for observed (log-)intensities in a FeatureSet-like object (ExpressionFeatureSet, ExonFeatureSet, SnpFeatureSet, TilingFeatureSet) and ExpressionSet.

**Usage**

```
## S4 method for signature 'FeatureSet':
boxplot(x, which=c("pm", "mm", "bg", "both", "all"), transfo=log2, nsample=10000)

## S4 method for signature 'ExpressionSet':
boxplot(x, which, transfo=identity, nsample=10000, ...)
```

**Arguments**

x	a FeatureSet-like object or ExpressionSet object.
which	character defining what probe types are to be used in the plot.
transfo	a function to transform the data before plotting. See 'Details'.
nsample	number of units to sample and build the plot.
...	arguments to be passed to the default boxplot method.

**Details**

The 'transfo' argument will set the transformation to be used. For raw data, 'transfo=log2' is a common practice. For summarized data (which are often in log2-scale), no transformation is needed (therefore 'transfo=identity').

**Note**

The boxplot methods for `FeatureSet` and `Expression` use a `sample` (via `sample`) of the probes/probesets to produce the plot. Therefore, the user interested in reproducibility is advised to use `set.seed`.

**See Also**

[hist](#), [image](#), [sample](#), [set.seed](#)

---

chromosome

*Accessor for chromosome information*

---

**Description**

Returns chromosome information.

**Usage**

```
pmChr(object)
```

**Arguments**

object      `TilingFeatureSet` or `SnpcallSet` object

**Details**

`chromosome()` returns the chromosomal information for all probes and `pmChr()` subsets the output to the PM probes only (if a `TilingFeatureSet` object).

**Value**

Vector with chromosome information.

---

darkColors *Create set of colors, interpolating through a set of preferred colors.*

---

### Description

Create set of colors, interpolating through a set of preferred colors.

### Usage

```
darkColors(n)
seqColors(n)
```

### Arguments

n integer determining number of colors to be generated

### Details

darkColors is based on the Dark2 palette in RColorBrewer, therefore useful to describe qualitative features of the data.

seqColors is based on Blues and generates a gradient of blues, therefore useful to describe quantitative features of the data.

### Examples

```
x <- 1:10
y <- 1:10
cols1 <- darkColors(10)
cols2 <- seqColors(10)
plot(x, y, col=cols1, xlim=c(1, 11))
points(x+1, y, col=cols2)
```

---

getX *Accessors for physical array coordinates.*

---

### Description

Accessors for physical array coordinates.

### Usage

```
getX(object, type)
getY(object, type)
```

### Arguments

object FeatureSet object  
 type 'character' defining the type of the probes to be queried. Valid options are 'pm', 'mm', 'bg'

**Value**

A vector with the requested coordinates.

**Examples**

```
## Not run:
x <- read.celfiles(list.celfiles())
theXpm <- getX(x, "pm")
theYpm <- getY(x, "pm")

## End(Not run)
```

crlmm

*Genotype Calls***Description**

Performs genotype calls via CRLMM (Corrected Robust Linear Model with Maximum-likelihood based distances).

**Usage**

```
crlmm(filenamees, outdir, batch_size=40000, balance=1.5,
       minLLRforCalls=c(5, 1, 5), recalibrate=TRUE,
       verbose=TRUE, pkgname, reference=TRUE)
justCRLMM(filenamees, batch_size = 40000, minLLRforCalls = c(5, 1, 5),
           recalibrate = TRUE, balance = 1.5, phenoData = NULL, verbose = TRUE,
           pkgname = NULL, tmpdir=tempdir())
```

**Arguments**

filenamees	character vector with the filenames.
outdir	directory where the output (and some tmp files) files will be saved.
batch_size	integer defining how many SNPs should be processed at a time.
recalibrate	Logical - should recalibration be performed?
balance	Control parameter to balance homozygotes and heterozygotes calls.
minLLRforCalls	Minimum thresholds for genotype calls.
verbose	Logical.
phenoData	phenoData object or NULL
pkgname	alt. pdInfo package to be used
reference	logical, defaulting to TRUE ...
tmpdir	Directory where temporary files are going to be stored at.

**Value**

SnpCallSetPlus object.

---

```
getAffinitySplineCoefficients
```

*Estimate affinity coefficients.*

---

**Description**

Estimate affinity coefficients using sequence information and splines.

**Usage**

```
getAffinitySplineCoefficients(intensities, sequences)
```

**Arguments**

`intensities` Intensity matrix  
`sequences` Probe sequences

**Value**

Matrix with estimated coefficients.

**See Also**

`getBaseProfile`

---

```
getBaseProfile
```

*Compute and plot nucleotide profile.*

---

**Description**

Computes and, optionally, plots nucleotide profile, describing the sequence effect on intensities.

**Usage**

```
getBaseProfile(coefs, probeLength = 25, plot = FALSE, ...)
```

**Arguments**

`coefs` affinity spline coefficients.  
`probeLength` length of probes  
`plot` logical. Plots profile?  
`...` arguments to be passed to `matplot`.

**Value**

Invisibly returns a matrix with estimated effects.

---

getContainer	<i>Get container information for NimbleGen Tiling Arrays.</i>
--------------	---

---

**Description**

Get container information for NimbleGen Tiling Arrays. This is useful for better identification of control probes.

**Usage**

```
getContainer(object, probeType)
```

**Arguments**

object	A <code>TilingFeatureSet</code> or <code>TilingFeatureSet</code> object.
probeType	String describing which probes to query ('pm', 'bg')

**Value**

'character' vector with container information.

---

getCrlmmSummaries	<i>Function to get CRLMM summaries saved to disk</i>
-------------------	--

---

**Description**

This will read the summaries written to disk and return them to the user as a `SnpCallSetPlus` or `SnpCnvCallSetPlus` object.

**Usage**

```
getCrlmmSummaries(tmpdir)
```

**Arguments**

tmpdir	directory where CRLMM saved the results to.
--------	---

**Value**

If the data were from SNP 5.0 or 6.0 arrays, the function will return a `SnpCnvCallSetPlus` object. It will return a `SnpCallSetPlus` object, otherwise.

---

 getNetAffx

*NetAffx Biological Annotations*


---

### Description

Gets NetAffx Biological Annotations saved in the annotation package (Exon and Gene ST Affymetrix arrays).

### Usage

```
getNetAffx(object, type = "probeset")
```

### Arguments

object	'ExpressionSet' object (eg., result of rma())
type	Either 'probeset' or 'transcript', depending on what type of summaries were obtained.

### Details

This retrieves NetAffx annotation saved in the (pd) annotation package - annotation(object). It is only available for Exon ST and Gene ST arrays.

The 'type' argument should match the summarization target used to generate 'object'. The 'rma' method allows for two targets: 'probeset' (target='probeset') and 'transcript' (target='core', target='full', target='extended').

### Value

'AnnotatedDataFrame' that can be used as featureData(object)

### Author(s)

Benilton Carvalho

---

 getNgsColorsInfo

*Helper function to extract color information for filenames on Nimble-Gen arrays.*


---

### Description

This function will (try to) extract the color information for NimbleGen arrays. This is useful when using read.xlsfiles2 to parse XYS files for Tiling applications.

### Usage

```
getNgsColorsInfo(path = ".", pattern1 = "_532", pattern2 = "_635", ...)
```



**Arguments**

<code>path</code>	path where to look for files
<code>pattern1</code>	pattern to match files supposed to go to the first channel
<code>pattern2</code>	pattern to match files supposed to go to the second channel
<code>...</code>	extra arguments for <code>list.xlsfiles</code>

**Details**

Many NimbleGen samples are identified following the pattern `sampleID_532.XYS / sampleID_635.XYS`. The function suggests sample names if all the filenames follow the standard above.

**Value**

A data.frame with, at least, two columns: 'channel1' and 'channel2'. A third column, 'sample-Names', is returned if the filenames follow the `sampleID_532.XYS / sampleID_635.XYS` standard.

**Author(s)**

Benilton Carvalho <bcarvalh@jhsp.edu>

---

`getPlatformDesign` *Retrieve Platform Design object*

---

**Description**

Retrieve platform design object.

**Usage**

```
getPlatformDesign(object)
getPD(object)
```

**Arguments**

<code>object</code>	FeatureSet object
---------------------	-------------------

**Details**

Retrieve platform design object.

**Value**

`platformDesign` or `PDInfo` object.

---

 hist-methods

*Density estimate*


---

### Description

Plot the density estimates for each sample

### Usage

```
## S4 method for signature 'FeatureSet':
hist(x, transfo=log2, which=c("pm", "mm", "bg", "both", "all"),
      nsample=10000, ...)

## S4 method for signature 'ExpressionSet':
hist(x, transfo=identity, nsample=10000, ...)
```

### Arguments

x	FeatureSet or ExpressionSet object
transfo	a function to transform the data before plotting. See 'Details'.
nsample	number of units to sample and build the plot.
which	set of probes to be plotted ("pm", "mm", "bg", "both", "all").
...	arguments to be passed to <code>matplot</code>

### Details

The 'transfo' argument will set the transformation to be used. For raw data, 'transfo=log2' is a common practice. For summarized data (which are often in log2-scale), no transformation is needed (therefore 'transfo=identity').

### Note

The hist methods for FeatureSet and Expression use a sample (via `sample`) of the probes/probesets to produce the plot (unless `nsample > nrow(x)`). Therefore, the user interested in reproducibility is advised to use `set.seed`.

---

 image-methods

*Display a pseudo-image of a microarray chip*


---

### Description

Produces a pseudo-image (`graphics::image`) for each sample.

**Usage**

```
## S4 method for signature 'FeatureSet':
image(x, which, transfo=log2, ...)
```

**Arguments**

x	FeatureSet object
which	integer indices of samples to be plotted (optional).
transfo	function to be applied to the data prior to plotting.
...	parameters to be passed to image

---

mmindex	<i>Accessors for PM, MM or background probes indices.</i>
---------	---

---

**Description**

Extracts the indexes for PM, MM or background probes.

**Usage**

```
mmindex(object, ...)
pmindex(object, ...)
bgindex(object, ...)
```

**Arguments**

object	FeatureSet or DBPDInfo object
...	Extra arguments, not yet implemented

**Details**

The indices are ordered by 'fid', i.e. they follow the order that the probes appear in the CEL/XYS files.

**Value**

A vector of integers representing the rows of the intensity matrix that correspond to PM, MM or background probes.

**Examples**

```
## How pm() works
## Not run:
x <- read.celfiles(list.celfiles())
pms0 <- pm(x)
pmi <- pmindex(x)
pms1 <- exprs(x)[pmi,]
identical(pms0, pms1)

## End(Not run)
```

mm

*Accessors and replacement methods for the PM/MM/BG matrices.***Description**

Accessors and replacement methods for the PM/MM/BG matrices.

**Usage**

```
mm(object, subset = NULL)
pm(object, subset = NULL, ...)
bg(object, subset = NULL)
mm(object) <- value
pm(object) <- value
bg(object) <- value
```

**Arguments**

object	FeatureSet object.
subset	Not implemented yet.
value	matrix object.
...	Extra arguments for future implementation.

**Details**

For all objects but `TilingFeatureSet`, these methods will return matrices. In case of `TilingFeatureSet` objects, the value is a 3-dimensional array (probes x samples x channels).

**Examples**

```
if (require(maqcExpression4plex) & require(pd.hg18.60mer.expr)) {
  xysPath <- system.file("extdata", package="maqcExpression4plex")
  xysFiles <- list.xysfiles(xysPath, full.name=TRUE)
  ngsExpressionFeatureSet <- read.xysfiles(xysFiles)
  pm(ngsExpressionFeatureSet)[1:10,]
}
```

justSNPRMA

*Summarization of SNP data***Description**

This function implements the SNPRMA method for summarization of SNP data. It works directly with the CEL files, saving memory.

**Usage**

```
justSNPRMA(filenamees, verbose = TRUE, phenoData = NULL, normalizeToHapmap = TRUE)
```

**Arguments**

filenames      character vector with the filenames.  
verbose         logical flag for verbosity.  
phenoData      a phenoData object or NULL  
normalizeToHapmap      Normalize to Hapmap? Should always be TRUE, but it's kept here for future use.

**Value**

SnpQSet or a SnpCnvQSet, depending on the array type.

**Examples**

```
## snprmaResults <- justSNPRMA(list.celfiles())
```

---

list.xysfiles      *List XYS files*

---

**Description**

Lists the XYS files.

**Usage**

```
list.xysfiles(...)
```

**Arguments**

...             parameters to be passed to [list.files](#)

**Details**

The functions interface [list.files](#) and the user is asked to check that function for further details.

**Value**

Character vector with the filenames.

**See Also**

[list.files](#)

**Examples**

```
list.xysfiles()
```

---

MAplot-methods      *MA plots*

---

### Description

Create MA plots using a reference array (if one channel) or using channel2 as reference (if two channel).

### Methods

**object = "FeatureSet"** ExpressionFeatureSet

---

oligo-package      *The oligo package: a tool for low-level analysis of oligonucleotide arrays*

---

### Description

The **oligo** package provides tools to preprocess different oligonucleotide arrays types: expression, tiling, SNP and exon chips. The supported manufacturers are Affymetrix and NimbleGen.

It offers support to large datasets (when the **bigmemory** is loaded) and can execute preprocessing tasks in parallel (if, in addition to **bigmemory**, the **snow** package is also loaded).

### Details

The package will read the raw intensity files (CEL for Affymetrix; XYS for NimbleGen) and allow the user to perform analyses starting at the feature-level.

Reading in the intensity files require the existence of data packages that contain the chip specific information (X/Y coordinates; feature types; sequence). These data packages are built using the **pdInfoBuilder** package.

For Affymetrix SNP arrays, users are asked to download the already built annotation packages from BioConductor. This is because these packages contain metadata that are not automatically created. The following annotation packages are available:

50K Xba - pd.mapping50kxba.240 50K Hind - pd.mapping50khind.240 250K Sty - pd.mapping250k.sty  
 250K Nsp - pd.mapping250k.nsp GenomeWideSnp 5 (SNP 5.0) - pd.genomewidesnp.5 GenomeWideSnp  
 6 (SNP 6.0) - pd.genomewidesnp.6

For users interested in genotype calls for SNP 5.0 and 6.0 arrays, we strongly recommend the use of the **crlmm** package, which implements a more efficient version of CRLMM.

### Author(s)

Benilton Carvalho - <carvalho@bclab.org>

### References

Carvalho, B.; Bengtsson, H.; Speed, T. P. & Irizarry, R. A. Exploration, Normalization, and Genotype Calls of High Density Oligonucleotide SNP Array Data. *Biostatistics*, 2006.

---

plotM-methods      *Methods for Log-Ratio plotting*

---

**Description**

The plotM methods are meant to plot log-ratios for different classes of data.

**Methods**

**object = "SnpQSet", i = "character"** Plot log-ratio for SNP data for sample i.

**object = "SnpQSet", i = "integer"** Plot log-ratio for SNP data for sample i.

**object = "SnpQSet", i = "numeric"** Plot log-ratio for SNP data for sample i.

**object = "TilingQSet", i = "missing"** Plot log-ratio for Tiling data for sample i.

---

pmAllele      *Access the allele information for PM probes.*

---

**Description**

Accessor to the allelic information for PM probes.

**Usage**

```
pmAllele(object)
```

**Arguments**

object      SnpFeatureSet or PDInfo object.

---

pmFragmentLength      *Access the fragment length for PM probes.*

---

**Description**

Accessor to the fragment length for PM probes.

**Usage**

```
pmFragmentLength(object)
```

**Arguments**

object      PDInfo object.

---

pmPosition	<i>Accessor to position information</i>
------------	---

---

**Description**

pmPosition will return the genomic position for the (PM) probes.

**Usage**

```
pmPosition(object)
pmOffset(object)
```

**Arguments**

object            AffySNPPDInfo, TilingFeatureSet or SnpCallSet object

**Details**

pmPosition will return genomic position for PM probes on a tiling array.

pmOffset will return the offset information for PM probes on SNP arrays.

---

pmStrand	<i>Accessor to the strand information</i>
----------	---

---

**Description**

Returns the strand information on SNP arrays for PM probes (0 - sense / 1 - antisense).

**Usage**

```
pmStrand(object)
```

**Arguments**

object            AffySNPPDInfo object



summarize

*Tools for microarray preprocessing***Description**

Preprocess microarray data. Includes background correction, normalization and summarization methods.

**Usage**

```
backgroundCorrect(object, method="rma", copy=TRUE, verbose=TRUE, ...)
summarize(object, probes=rownames(object), method="medianpolish", verbose=TRUE,
normalize(object, method="quantile", copy=TRUE, verbose=TRUE, ...)
normalizeToTarget(object, target, method="quantile", copy=TRUE, verbose=TRUE)
```

**Arguments**

object	Object containing probe intensities to be preprocessed.
method	String determining which method to use at that preprocessing step.
target	Vector with the target distribution
probes	Character vector that identifies the name of the probes represented by the rows of object.
copy	Logical flag determining if data must be copied before processing (TRUE), or if data can be overwritten (FALSE).
verbose	Logical flag for verbosity.
...	Arguments to be passed to methods.

**Details**

Number of rows of `object` must match the length of `probes`. Currently, only the following methods are implemented: - backgroundCorrection: `rma.background` - normalize: `quantile` - summarization: `median-polish`

**Value**

Expression matrix with `length(unique(probes))` rows and `ncol(object)` columns.

**Examples**

```
ns <- 100
nps <- 1000
np <- 10
intensities <- matrix(rnorm(ns*nps*np, 8000, 400), nc=ns)
ids <- rep(as.character(1:nps), each=np)
bgCorrected <- backgroundCorrect(intensities)
normalized <- normalize(bgCorrected)
expression <- summarize(normalized, probes=ids)
intensities[1:20, 1:3]
expression[1:20, 1:3]
target <- rnorm(np*nps)
normalizedToTarget <- normalizeToTarget(intensities, target)
```

---

probeNames	<i>Accessor to feature names</i>
------------	----------------------------------

---

**Description**

Accessor to PM feature names.

**Usage**

```
probeNames(object, subset = NULL)
```

**Arguments**

object	FeatureSet or DBPDInfo
subset	not implemented yet.

---

read.celfiles	<i>Parser to CEL files</i>
---------------	----------------------------

---

**Description**

Reads CEL files.

**Usage**

```
read.celfiles(..., filenames, pkgname, phenoData, featureData,
experimentData, protocolData, notes, verbose=TRUE, sampleNames,
rm.mask=FALSE, rm.outliers=FALSE, rm.extra=FALSE, checkType=TRUE)
```

```
read.celfiles2(channel1, channel2, pkgname, phenoData, featureData,
experimentData, protocolData, notes, verbose=TRUE, sampleNames,
rm.mask=FALSE, rm.outliers=FALSE, rm.extra=FALSE, checkType=TRUE)
```

**Arguments**

...	names of files to be read.
filenames	a character vector with the CEL filenames.
channel1	a character vector with the CEL filenames for the first 'channel' on a Tiling application
channel2	a character vector with the CEL filenames for the second 'channel' on a Tiling application
pkgname	alternative data package to be loaded.
phenoData	phenoData
featureData	featureData
experimentData	experimentData
protocolData	protocolData

notes	notes
verbose	logical
sampleNames	character vector with sample names (usually better descriptors than the file-names)
rm.mask	logical. Read masked?
rm.outliers	logical. Remove outliers?
rm.extra	logical. Remove extra?
checkType	logical. Check type of each file? This can be time consuming.

### Details

When using 'affyio' to read in CEL files, the user can read compressed CEL files (CEL.gz). Additionally, 'affyio' is much faster than 'affxparser'.

The function guesses which annotation package to use from the header of the CEL file. The user can also provide the name of the annotation package to be used (via the `pkgname` argument). If the annotation package cannot be loaded, the function returns an error. If the annotation package is not available from BioConductor, one can use the `pdInfoBuilder` package to build one.

### Value

ExpressionFeatureSet  
     if Expression arrays  
 ExonFeatureSet  
     if Exon arrays  
 SnpFeatureSet  
     if SNP arrays  
 TilingFeatureSet  
     if Tiling arrays

### See Also

[list.celfiles](#), [read.xysfiles](#)

### Examples

```
if(require(pd.mapping50k.xba240) & require(hapmap100kxba)){
  celPath <- system.file("celFiles", package="hapmap100kxba")
  celFiles <- list.celfiles(celPath, full.name=TRUE)
  affySnpFeatureSet <- read.celfiles(celFiles)
}
```

---

readSummaries

*Read summaries generated by crlmm*

---

### Description

This function read the different summaries generated by crlmm.

**Usage**

```
readSummaries(type, tmpdir)
```

**Arguments**

type	type of summary of character class: 'alleleA', 'alleleB', 'alleleA-sense', 'alleleA-antisense', 'alleleB-sense', 'alleleB-antisense', 'calls', 'llr', 'conf'.
tmpdir	directory containing the output saved by crlmm

**Details**

On the 50K and 250K arrays, given a SNP, there are probes on both strands (sense and antisense). For this reason, the options 'alleleA-sense', 'alleleA-antisense', 'alleleB-sense' and 'alleleB-antisense' should be used **\*\*only\*\*** with such arrays (XBA, HIND, NSP or STY).

On the SNP 5.0 and SNP 6.0 platforms, this distinction does not exist in terms of algorithm (note that the actual strand could be queried from the annotation package). For these arrays, options 'alleleA', 'alleleB' are the ones to be used.

The options `calls`, `llr` and `conf` will return, respectively, the CRLMM calls, log-likelihood ratios (for devel purpose **\*\*only\*\***) and CRLMM confidence calls matrices.

**Value**

Matrix with values of summaries.

---

read.xysfiles	<i>Parser to XYS files</i>
---------------	----------------------------

---

**Description**

NimbleGen provides XYS files which are read by this function.

**Usage**

```
read.xysfiles(..., filenames, pkgname, phenoData, featureData,
experimentData, protocolData, notes, verbose=TRUE, sampleNames,
checkType=TRUE)
```

```
read.xysfiles2(channel1, channel2, pkgname, phenoData, featureData,
experimentData, protocolData, notes, verbose=TRUE, sampleNames,
checkType=TRUE)
```

**Arguments**

...	file names
filenames	character vector with filenames.
channel1	a character vector with the XYS filenames for the first 'channel' on a Tiling application
channel2	a character vector with the XYS filenames for the second 'channel' on a Tiling application

pkgname	character vector with alternative PD Info package name
phenoData	phenoData
featureData	featureData
experimentData	experimentData
protocolData	protocolData
notes	notes
verbose	verbose
sampleNames	character vector with sample names (usually better descriptors than the file-names)
checkType	logical. Check type of each file? This can be time consuming.

### Details

The function will read the XYS files provided by NimbleGen Systems and return an object of class `FeatureSet`.

The function guesses which annotation package to use from the header of the XYS file. The user can also provide the name of the annotation package to be used (via the `pkgname` argument). If the annotation package cannot be loaded, the function returns an error. If the annotation package is not available from BioConductor, one can use the `pdInfoBuilder` package to build one.

### Value

`ExpressionFeatureSet`  
if Expression arrays  
`TilingFeatureSet`  
if Tiling arrays

### See Also

[list.xysfiles](#), [read.celfiles](#)

### Examples

```
if (require(maqcExpression4plex) & require(pd.hg18.60mer.expr)) {
  xysPath <- system.file("extdata", package="maqcExpression4plex")
  xysFiles <- list.xysfiles(xysPath, full.name=TRUE)
  ngsExpressionFeatureSet <- read.xysfiles(xysFiles)
}
```

### Description

Robust Multichip Average preprocessing methodology. This strategy allows background subtraction, quantile normalization and summarization (via median-polish).

**Usage**

```

## S4 method for signature 'ExonFeatureSet':
rma(object, background=TRUE, normalize=TRUE, subset=NULL, target="core")
## S4 method for signature 'ExpressionFeatureSet':
rma(object, background=TRUE, normalize=TRUE, subset=NULL)
## S4 method for signature 'GeneFeatureSet':
rma(object, background=TRUE, normalize=TRUE, subset=NULL, target="core")
## S4 method for signature 'SnpCnvFeatureSet':
rma(object, background=TRUE, normalize=TRUE, subset=NULL)

```

**Arguments**

object	Exon/Expression/Gene/SnpCnv-FeatureSet object.
background	Logical - perform RMA background correction?
normalize	Logical - perform quantile normalization?
subset	To be implemented.
target	Level of summarization (only for Exon/Gene arrays)

**Methods**

`signature(object = "ExonFeatureSet")` When applied to an `ExonFeatureSet` object, `rma` can produce summaries at different levels: probeset (as defined in the PGF), core genes (as defined in the core.mps file), full genes (as defined in the full.mps file) or extended genes (as defined in the extended.mps file). To determine the level for summarization, use the `target` argument.

`signature(object = "ExpressionFeatureSet")` When used on an `ExpressionFeatureSet` object, `rma` produces summaries at the probeset level (as defined in the CDF or NDF files, depending on the manufacturer).

`signature(object = "GeneFeatureSet")` When applied to a `GeneFeatureSet` object, `rma` can produce summaries at different levels: probeset (as defined in the PGF) and 'core genes' (as defined in the core.mps file). To determine the level for summarization, use the `target` argument.

`signature(object = "SnpCnvFeatureSet")` If used on a `SnpCnvFeatureSet` object (ie., SNP 5.0 or SNP 6.0 arrays), `rma` will produce summaries for the CNV probes. Note that this is an experimental feature for internal (and quick) assessment of CNV probes. We recommend the use of the 'crlmm' package, which contains a Copy Number tool specifically designed for these data.

**References**

Rafael. A. Irizarry, Benjamin M. Bolstad, Francois Collin, Leslie M. Cope, Bridget Hobbs and Terence P. Speed (2003), Summaries of Affymetrix GeneChip probe level data *Nucleic Acids Research* 31(4):e15

Bolstad, B.M., Irizarry R. A., Astrand M., and Speed, T.P. (2003), A Comparison of Normalization Methods for High Density Oligonucleotide Array Data Based on Bias and Variance. *Bioinformatics* 19(2):185-193

Irizarry, RA, Hobbs, B, Collin, F, Beazer-Barclay, YD, Antonellis, KJ, Scherf, U, Speed, TP (2003) Exploration, Normalization, and Summaries of High Density Oligonucleotide Array Probe Level Data. *Biostatistics*. Vol. 4, Number 2: 249-264

**See Also**[snprma](#)**Examples**

```
if (require(maqcExpression4plex) & require(pd.hg18.60mer.expr)) {
  xysPath <- system.file("extdata", package="maqcExpression4plex")
  xysFiles <- list.xysfiles(xysPath, full.name=TRUE)
  ngsExpressionFeatureSet <- read.xysfiles(xysFiles)
  summarized <- rma(ngsExpressionFeatureSet)
  show(summarized)
}
```

---

runDate	<i>Date of scan</i>
---------	---------------------

---

**Description**

Retrieves date information in CEL/XYS files.

**Usage**

```
runDate(object)
```

**Arguments**

object            'FeatureSet' object.

---

sequenceDesignMatrix	<i>Create design matrix for sequences</i>
----------------------	---

---

**Description**

Creates design matrix for sequences.

**Usage**

```
sequenceDesignMatrix(seqs)
```

**Arguments**

seqs              character vector of 25-mers.

**Details**

This assumes all sequences are 25bp long.

The design matrix is often used when the objective is to adjust intensities by sequence.

**Value**

Matrix with length(seqs) rows and 75 columns.

**Examples**

```
genSequence <- function(x)
  paste(sample(c("A", "T", "C", "G"), 25, rep=TRUE), collapse="", sep="")
seqs <- sapply(1:10, genSequence)
X <- sequenceDesignMatrix(seqs)
Y <- rnorm(10, mean=12, sd=2)
Ydemean <- Y-mean(Y)
X[1:10, 1:3]
fit <- lm(Ydemean~X)
coef(fit)
```

---

mmSequence

*Probe Sequences*


---

**Description**

Accessor to the (PM/MM/background) probe sequences.

**Usage**

```
mmSequence(object)
pmSequence(object, ...)
bgSequence(object, ...)
```

**Arguments**

```
object      FeatureSet, AffySNPPDInfo or DBPDInfo object
...         additional arguments
```

**Value**

A DNASTringSet containing the PM/MM/background probe sequence associated to the array.

---

snprma

*Preprocessing SNP Arrays*


---

**Description**

This function preprocess SNP arrays.

**Usage**

```
snprma(object, verbose = TRUE, normalizeToHapmap = TRUE)
```



**Arguments**

object            SnpFeatureSet object  
verbose           Verbosity flag. logical  
normalizeToHapmap  
                  internal

**Value**

A SnpQSet object.

# Index

## \*Topic **IO**

read.celfiles, 18  
read.xlsfiles, 20

## \*Topic **classif**

crlmm, 5  
getNetAffx, 8  
runDate, 23

## \*Topic **file**

list.xlsfiles, 13

## \*Topic **hplot**

boxplot-methods, 2  
darkColors, 4  
hist-methods, 10  
image-methods, 10

## \*Topic **manip**

basecontent, 1  
basicRMA, 1  
chromosome, 3  
getAffinitySplineCoefficients,  
6  
getBaseProfile, 6  
getContainer, 7  
getCrlmmSummaries, 7  
getNgsColorsInfo, 8  
getPlatformDesign, 9  
getX, 4  
justSNPRMA, 12  
mm, 12  
mmindex, 11  
mmSequence, 24  
pmAllele, 15  
pmFragmentLength, 15  
pmPosition, 16  
pmStrand, 16  
probeNames, 18  
readSummaries, 19  
sequenceDesignMatrix, 23  
snprma, 24  
summarize, 17

## \*Topic **methods**

boxplot-methods, 2  
hist-methods, 10  
MAplot-methods, 14

plotM-methods, 15  
rma-methods, 21

## \*Topic **package**

oligo-package, 14

backgroundCorrect (*summarize*), 17

backgroundCorrect, ExpressionFeatureSet-method  
(*summarize*), 17

backgroundCorrect, FeatureSet-method  
(*summarize*), 17

backgroundCorrect, ff\_matrix-method  
(*summarize*), 17

backgroundCorrect, matrix-method  
(*summarize*), 17

backgroundCorrect-methods  
(*summarize*), 17

basecontent, 1

basicRMA, 1

bg (*mm*), 12

bg, FeatureSet-method (*mm*), 12

bg, TilingFeatureSet-method (*mm*),  
12

bg<- (*mm*), 12

bg<-, FeatureSet, matrix-method  
(*mm*), 12

bg<-, TilingFeatureSet, array-method  
(*mm*), 12

bgindex (*mmindex*), 11

bgindex, DBPDInfo-method  
(*mmindex*), 11

bgindex, FeatureSet-method  
(*mmindex*), 11

bgSequence (*mmSequence*), 24

bgSequence, DBPDInfo-method  
(*mmSequence*), 24

bgSequence, ExonFeatureSet-method  
(*mmSequence*), 24

bgSequence, FeatureSet-method  
(*mmSequence*), 24

bgSequence, GeneFeatureSet-method  
(*mmSequence*), 24

boxplot, ExpressionSet-method  
(*boxplot-methods*), 2

- boxplot, FeatureSet-method  
(*boxplot-methods*), 2
- boxplot-methods, 2
- chromosome, 3
- chromosome<- (*chromosome*), 3
- chromosome<-, AnnotatedDataFrame, character-method  
(*chromosome*), 3
- cleanPlatformName  
(*read.celfiles*), 18
- crlmm, 5
- darkColors, 4
- getAffinitySplineCoefficients, 6
- getBaseProfile, 6
- getContainer, 7
- getContainer, TilingFeatureSet-method  
(*getContainer*), 7
- getContainer-methods  
(*getContainer*), 7
- getCrlmmSummaries, 7
- getNetAffx, 8
- getNetAffx, ExpressionSet-method  
(*getNetAffx*), 8
- getNetAffx-methods (*getNetAffx*), 8
- getNgsColorsInfo, 8
- getPD (*getPlatformDesign*), 9
- getPlatformDesign, 9
- getPlatformDesign, FeatureSet-method  
(*getPlatformDesign*), 9
- getX, 4
- getX, DBPDInfo-method (*getX*), 4
- getX, FeatureSet-method (*getX*), 4
- getX-methods (*getX*), 4
- getY (*getX*), 4
- getY, DBPDInfo-method (*getX*), 4
- getY, FeatureSet-method (*getX*), 4
- getY-methods (*getX*), 4
- hist, 3
- hist, ExpressionSet-method  
(*hist-methods*), 10
- hist, FeatureSet-method  
(*hist-methods*), 10
- hist-methods, 10
- image, 3
- image, FeatureSet-method  
(*image-methods*), 10
- image-methods, 10
- justCRLMM (*crlmm*), 5
- justSNPRMA, 12
- list.celfiles, 19
- list.files, 13
- list.xysfiles, 13, 21
- MPlot (*MPlot-methods*), 14
- MPlot, FeatureSet-method  
(*MPlot-methods*), 14
- MPlot-methods, 14
- mm, 12
- mm, FeatureSet-method (*mm*), 12
- mm, TilingFeatureSet-method (*mm*),  
12
- mm<- (*mm*), 12
- mm<-, FeatureSet, matrix-method  
(*mm*), 12
- mm<-, TilingFeatureSet, array-method  
(*mm*), 12
- mmindex, 11
- mmindex, DBPDInfo-method  
(*mmindex*), 11
- mmindex, FeatureSet-method  
(*mmindex*), 11
- mmSequence, 24
- mmSequence, AffySNPPDInfo-method  
(*mmSequence*), 24
- mmSequence, DBPDInfo-method  
(*mmSequence*), 24
- mmSequence, FeatureSet-method  
(*mmSequence*), 24
- normalize (*summarize*), 17
- normalize, ff\_matrix-method  
(*summarize*), 17
- normalize, matrix-method  
(*summarize*), 17
- normalize-methods (*summarize*), 17
- normalizeToTarget (*summarize*), 17
- normalizeToTarget, ff\_matrix-method  
(*summarize*), 17
- normalizeToTarget, matrix-method  
(*summarize*), 17
- normalizeToTarget-methods  
(*summarize*), 17
- oligo-package, 14
- plotM (*plotM-methods*), 15
- plotM, SnpQSet, character-method  
(*plotM-methods*), 15
- plotM, SnpQSet, integer-method  
(*plotM-methods*), 15
- plotM, SnpQSet, numeric-method  
(*plotM-methods*), 15

- plotM, TilingQSet, missing-method  
(*plotM-methods*), 15
- plotM-methods, 15
- pm (*mm*), 12
- pm, FeatureSet-method (*mm*), 12
- pm, TilingFeatureSet-method (*mm*),  
12
- pm<- (*mm*), 12
- pm<-, FeatureSet, matrix-method  
(*mm*), 12
- pm<-, TilingFeatureSet, array-method  
(*mm*), 12
- pmAllele, 15
- pmAllele, AffySNPPDInfo-method  
(*pmAllele*), 15
- pmAllele, SnpFeatureSet-method  
(*pmAllele*), 15
- pmChr (*chromosome*), 3
- pmChr, ExonFeatureSet-method  
(*chromosome*), 3
- pmChr, FeatureSet-method  
(*chromosome*), 3
- pmChr, GeneFeatureSet-method  
(*chromosome*), 3
- pmFragmentLength, 15
- pmFragmentLength, AffySNPPDInfo-method  
(*pmFragmentLength*), 15
- pmindex (*mmindex*), 11
- pmindex, DBPDInfo-method  
(*mmindex*), 11
- pmindex, FeatureSet-method  
(*mmindex*), 11
- pmOffset (*pmPosition*), 16
- pmOffset, AffySNPPDInfo-method  
(*pmPosition*), 16
- pmPosition, 16
- pmPosition, ExpressionPDInfo-method  
(*pmPosition*), 16
- pmPosition, FeatureSet-method  
(*pmPosition*), 16
- pmPosition, TilingFeatureSet-method  
(*pmPosition*), 16
- pmPosition, TilingPDInfo-method  
(*pmPosition*), 16
- pmSequence (*mmSequence*), 24
- pmSequence, AffySNPPDInfo-method  
(*mmSequence*), 24
- pmSequence, DBPDInfo-method  
(*mmSequence*), 24
- pmSequence, FeatureSet-method  
(*mmSequence*), 24
- pmStrand, 16
- pmStrand, AffySNPPDInfo-method  
(*pmStrand*), 16
- probeNames, 18
- probeNames, DBPDInfo-method  
(*probeNames*), 18
- probeNames, ExonFeatureSet-method  
(*probeNames*), 18
- probeNames, FeatureSet-method  
(*probeNames*), 18
- probeNames, GeneFeatureSet-method  
(*probeNames*), 18
- read.celfiles, 18, 21
- read.celfiles2 (*read.celfiles*), 18
- read.xysfiles, 19, 20
- read.xysfiles2 (*read.xysfiles*), 20
- readSummaries, 19
- rma (*rma-methods*), 21
- rma, ExonFeatureSet-method  
(*rma-methods*), 21
- rma, ExpressionFeatureSet-method  
(*rma-methods*), 21
- rma, GeneFeatureSet-method  
(*rma-methods*), 21
- rma, SnpCnvFeatureSet-method  
(*rma-methods*), 21
- rma-methods, 21
- runDate, 23
- runDate, FeatureSet-method  
(*runDate*), 23
- runDate-methods (*runDate*), 23
- sample, 3
- seqColors (*darkColors*), 4
- sequenceDesignMatrix, 23
- set.seed, 3
- snprma, 23, 24
- summarize, 17
- summarize, ff\_matrix-method  
(*summarize*), 17
- summarize, matrix-method  
(*summarize*), 17
- summarize-methods (*summarize*), 17