

# Package ‘OLIN’

December 13, 2024

**Version** 1.85.0

**Date** 2022-05-06

**Title** Optimized local intensity-dependent normalisation of two-color microarrays

**Author** Matthias Futschik <mfutschik@ualg.pt>

**Maintainer** Matthias Futschik <mfutschik@ualg.pt>

**Depends** R (>= 2.10), methods, locfit, marray

**Imports** graphics, grDevices, limma, marray, methods, stats

**Suggests** convert

**Description** Functions for normalisation of two-color microarrays by optimised local regression and for detection of artefacts in microarray data

**biocViews** Microarray, TwoChannel, QualityControl, Preprocessing, Visualization

**License** GPL-2

**URL** <http://olin.sysbiolab.eu>

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

**git\_branch** devel

**git\_last\_commit** f6d5168

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2024-12-12

## Contents

anovaint . . . . .	2
anovapin . . . . .	4
anovaplate . . . . .	5
anovaspacial . . . . .	6
backgroundCorrect2 . . . . .	7
bas . . . . .	9

colorbar.mxy . . . . .	10
colorbar.mxy.abs . . . . .	11
colorbar.sig . . . . .	12
fdr.int . . . . .	13
fdr.int2 . . . . .	14
fdr.spatial . . . . .	15
fdr.spatial2 . . . . .	17
fgbg.visu . . . . .	18
ino . . . . .	19
lin . . . . .	21
m2v . . . . .	22
ma.matrix . . . . .	24
ma.vector . . . . .	25
mxy.abs.plot . . . . .	26
mxy.plot . . . . .	27
mxy2.plot . . . . .	28
oin . . . . .	30
olin . . . . .	31
p.int . . . . .	34
p.int2 . . . . .	36
p.spatial . . . . .	37
p.spatial2 . . . . .	39
sig.mask . . . . .	40
sigint.plot . . . . .	41
sigint.plot2 . . . . .	43
sigxy.plot . . . . .	44
sigxy.plot2 . . . . .	45
sw . . . . .	46
sw.olin . . . . .	47
sw.xy . . . . .	48
v2m . . . . .	49

## **Index** **51**

---

anovaint	<i>One-factorial ANOVA assessing intensity-dependent bias</i>
----------	---

---

### **Description**

This function performs an one-factorial analysis of variance assessing intensity-dependent bias for a single array. The predictor variable is the average logged intensity of both channels and the response variable is the logged fold-change.

### **Usage**

```
anovaint(obj, index, N=10)
```

**Arguments**

obj	object of class “marrayRaw” or “marrayNorm”
index	index of array to be tested
N	number of (intensity) levels for ANOVA

**Details**

The function `anovaint` performs a one-factorial ANOVA for objects of class “marrayRaw” or “marrayNorm”. The predictor variable is the average logged intensity of both channels  $A = 0.5 * (\log_2(\text{Ch1}) + \log_2(\text{Ch2}))$ .  $\text{Ch1}$ ,  $\text{Ch2}$  are the fluorescence intensities of channel 1 and channel 2, respectively. The response variable is the logged fold-change  $M = (\log_2(\text{Ch2}) - \log_2(\text{Ch1}))$ . The A-scale is divided in N intervals generating N levels of factor A. Note that N should divide the total number of spots approx. equally. The null hypothesis is the equality of mean(M) of the different levels (intervals). The model formula used is  $M \sim (A - 1)$  (without an intercept term).

**Value**

The return value is a list of summary statistics of the fitted model as produced by `summary.lm`. For example, the squared multiple correlation coefficient  $R^2$  equals the proportion of the variation of M that can be explained by the variation of A (based on the chosen ANOVA model.)

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[anova](#), [summary.lm](#), [anovaspatial](#), [marrayRaw](#), [marrayNorm](#)

**Examples**

```
# CHECK RAW DATA FOR INTENSITY-DEPENDENT BIAS
data(sw)
print(anovaint(sw, index=1, N=10))

# CHECK DATA NORMALISED BY OLIN FOR INTENSITY-DEPENDENT BIAS
data(sw.olin)
print(anovaint(sw.olin, index=1, N=10))
```

---

`anovapin`*One-factorial ANOVA assessing pin-dependent bias.*

---

### Description

This function performs an one-factorial analysis of variance assessing pin-dependent bias for a single array

### Usage

```
anovapin(obj, index)
```

### Arguments

<code>obj</code>	object of class “marrayRaw” or “marrayNorm”
<code>index</code>	index of array to be tested

### Details

The function `anovapin` performs a one-factorial ANOVA for objects of class “marrayRaw” or “marrayNorm”. The predictor variable is the pin index; the response variable is the logged fold-change  $M = (\log_2(\text{Ch2}) - \log_2(\text{Ch1}))$ . The null hypothesis is equal mean( $M$ ) of groups of spots printed by the same pin i.e. a spot’s  $M$  does not depend on the pin used from printing the spot. The model formula used is  $M \sim (\text{pin.index} - 1)$  (without an intercept term).

### Value

The return value is a list of summary statistics of the fitted model as produced by `summary.lm`. For example, the squared multiple correlation coefficient  $R^2$  equals the proportion of the variation of  $M$  that can be explained by the variation of pin index (based on the chosen ANOVA model.)

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### See Also

[anova](#), [summary.lm](#)

### Examples

```
# CHECK RAW DATA FOR INTENSITY-DEPENDENT BIAS
data(sw)
print(anovapin(sw, index=1))
```

```
# CHECK DATA NORMALISED BY OLIN FOR INTENSITY-DEPENDENT BIAS
data(sw.olin)
```

```
print(anovapin(sw.olin,index=1))
```

---

anovaplate

*One-factorial ANOVA assessing pin-dependent bias.*

---

## Description

This function performs an one-factorial analysis of variance assessing microtiter plate-dependent bias for a single array

## Usage

```
anovaplate(obj,index)
```

## Arguments

obj	object of class “marrayRaw” or “marrayNorm”
index	index of array to be tested

## Details

The function `anovapin` performs a one-factorial ANOVA for objects of class “marrayRaw” or “marrayNorm”. The predictor variable is the corresponding plate index as stored in the `maPlate` slot of `obj`; the response variable is the logged fold-change  $M=(\log_2(\text{Ch}_2)-\log_2(\text{Ch}_1))$ . The null hypothesis is equal mean( $M$ ) of groups of spots derived from the same microtiter plate i.e. a spot’s  $M$  does not dependent on the plate of origin. The model formula used is  $M \sim (\text{plate.index} - 1)$  (without an intercept term).

## Value

The return value is a list of summary statistics of the fitted model as produced by `summary.lm`. For example, the squared multiple correlation coefficient  $R^2$  equals the proportion of the variation of  $M$  that can be explained by the variation of plate index (based on the chosen ANOVA model.)

## Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

## See Also

[anova](#), [summary.lm](#)

**Examples**

```
# CHECK RAW DATA FOR INTENSITY-DEPENDENT BIAS
data(sw)
print(anovapin(sw,index=1))

# CHECK DATA NORMALISED BY OLIN FOR INTENSITY-DEPENDENT BIAS
data(sw.olin)
print(anovapin(sw.olin,index=1))
```

---

anovaspatial

*One-factorial ANOVA assessing spatial bias*


---

**Description**

This function performs an one-factorial analysis of variance to test for spatial bias for a single array. The predictor variable is the average logged intensity of both channels and the response variable is the logged fold-change.

**Usage**

```
anovaspatial(obj,index,xN=5,yN=5,visu=FALSE)
```

**Arguments**

obj	object of class “marrayRaw” or “marrayNorm”
index	index of array (within obj) to be tested
xN	number of intervals in x-direction
yN	number of intervals in y-direction
visu	If visu=TRUE, results are visualised (see below)

**Details**

The function `anovaspatial` performs a one-factorial ANOVA for objects of class “marrayRaw” or “marrayNorm”. The predictor variable is the average logged intensity of both channels ( $A = 0.5 * (\log_2(\text{Ch1}) + \log_2(\text{Ch2}))$ ).  $\text{Ch1}$ ,  $\text{Ch2}$  are the fluorescence intensities of channel 1 and channel 2, respectively. The response variable is the logged fold-change ( $M = (\log_2(\text{Ch2}) - \log_2(\text{Ch1}))$ ). The spot locations on the array is divided into  $xN$  intervals in x-direction and  $yN$  intervals in y-direction. This division defines  $(xN \times yN)$  rectangular spatial blocks on the array, and thus,  $(xN \times yN)$  levels (or treatments) for  $A$ . Note that values chosen for  $xN$  and  $yN$  should divide the array columns and rows approx. equally. The null hypothesis is the equality of mean( $M$ ) of the different levels. The model formula used by `anovaspatial` is  $M \sim (A - 1)$  (without an intercept term).

**Value**

The return value is a list of summary statistics of the fitted model as produced by `summary.lm`. For example, the squared multiple correlation coefficient  $R^2$  equals the proportion of the variation of  $M$  that can be related to the spot location (based on the chosen ANOVA.) Optionally, the distribution of p-values (as derived by t-test and stated in the summary statistics) can be visualised.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[anova](#), [summary.lm](#), [anovaint](#), [marrayRaw](#), [marrayNorm](#)

**Examples**

```
# CHECK RAW DATA FOR SPATIAL BIAS
data(sw)
print(anovaspatial(sw, index=1, xN=8, yN=8, visu=TRUE))

# CHECK DATA NORMALISED BY OLIN FOR SPATIAL BIAS
data(sw.olin)
print(anovaspatial(sw.olin, index=1, xN=8, yN=8, visu=TRUE))
# note the different scale of the colour bar
```

---

backgroundCorrect2      *Background correction*

---

**Description**

Background correction based on [backgroundCorrect](#) of the `limma` package.

**Usage**

```
backgroundCorrect2(object, method="subtract", offset=0)
```

**Arguments**

object	object of class <a href="#">marrayRaw</a>
method	method for background correction: “none”, “subtract”, “half”, “minimum”, “movingmin”, “edwards” or “normexp”.
offset	numeric value to add to intensities

## Details

This function is a wrapper function for [backgroundCorrect](#) with following methods implemented:

- “none”: no background correction
- “subtract”: simple subtraction of background intensities
- “movingmin”: background intensities are first averaged over 3x3 grids of neighbouring spots and subsequently subtracted
- “minimum”: zero or negative intensities after background correction are set equal to half the minimum of positive corrected intensities
- “edwards”: background correction based on log-linear interpolation
- “normexp”: background correction based on fitting procedure

For further details and references, please refer to its help page. An alternative Bayesian model for background correction ([kooperberg](#)) is also implemented in the `limma` package.

## Value

Background correct object of class `marrayRaw`.

## Author(s)

Matthias Futschik

## See Also

[backgroundCorrect](#), [kooperberg](#)

## Examples

```
# Loading data
data(sw)

#No background correction
sw.none <- backgroundCorrect2(sw,method="none")
plot(maA(sw.none)[,1],maM(sw.none)[,1])

# Simple subtraction
sw.sub <- backgroundCorrect2(sw,method="sub")
points(maA(sw.sub)[,1],maM(sw.sub)[,1],col="red")
```



---

bas *Between-array scaling*

---

**Description**

This function performs an between-array scaling

**Usage**

```
bas(obj, mode="var")
```

**Arguments**

obj	object of "marrayNorm"
mode	mode of scaling. Default option is scaling of arrays to have the same within-array variance of logged ratios ( <i>var</i> ). Alternatively, <i>mad</i> or <i>qq</i> can be used (see details)

**Details**

The function *bas* adjust the scale of logged ratios ( $M = (\log_2(\text{Ch}_2) - \log_2(\text{Ch}_1))$ ) between the different arrays stored in *obj*.

Following schemes (*mode*) are implemented:

- *mode="var"*: Logged ratios *M* are scaled to show the same (within-array) variance for all arrays in the batch stored in *obj*. The variance is calculated using *var*.
- *mode="mad"*: The same procedure as for *mode="var"* is applied using, however, median absolute deviation (*mad*) as robust estimate for within-array variance.
- *mode="qq"*: The *quantile scaling* is using the same procedure as the quantile normalisation described by Bolstad et al. (2003). In brief: Given *X* is the matrix with logged ratios (column corresponding to arrays, rows to genes)
  1. Sort each column of *X* (independently) producing *X<sub>s</sub>*,
  2. Replace values in each row of *X<sub>s</sub>* by the mean value of the row producing *X<sub>sm</sub>*,
  3. Rearrange the ordering for each column of matrix *X<sub>sm</sub>*, so that it has the columns have same ordering as for the original matrix *X*.

The last step yields the scaled logged ratios *M*.

**Note**

Between-array scaling should only be performed if it can be assumed that the different arrays have a similar distribution of logged ratios. This has to be checked on a case-by-case basis. Caution should be taken in the interpretation of results for arrays hybridised with biologically divergent samples, if between-array scaling is applied.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**References**

Bolstad et al., A comparison of normalization methods for high density oligonucleotide array data based on variance and bias, *Bioinformatics*, 19: 185-193, 2003

**See Also**

[marrayNorm](#), [var](#), [mad](#)

**Examples**

```
# DISTRIBUTION OF M BEFORE SCALING
data(sw.olin)

col <- c("red", "blue", "green", "orange")
M <- maM(sw.olin)

plot(density(M[,4]), col=col[4], xlim=c(-2,2))
for (i in 1:3){
  lines(density(M[,i]), col=col[i])
}

# SCALING AND VISUALISATION
sw.olin.s <- bas(sw.olin, mode="var")

M <- maM(sw.olin.s)

plot(density(M[,4]), col=col[4], xlim=c(-2,2))
for (i in 1:3){
  lines(density(M[,i]), col=col[i])
}
```

---

colorbar.mxy

*Generates a colour bar*

---

**Description**

Generates colour bar for MXY plots

**Usage**

```
colorbar.mxy(color.lim,
             col=c(rgb(0,(100:0)/100,0),rgb(0,0,0),rgb((1:100)/100,green=0,blue=0)),
             ylab="", ylablim=FALSE)
```

**Arguments**

color.lim	limits for colour range
col	colour palette to be used
ylab	label of ordinate of color bar
ylablim	If TRUE, the axis annotation consists only of the limits of the colour range.

**Details**

The function `colorbar.mxy` produces a colour bar for MXY plots. The default colours used range from green (for the lower limit of the colour range) to red (for its upper limit). For visualisation, values below or above the limits for the colour range (as given by `color.lim`) are set to the lower or upper limit, respectively.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[mxy.plot](#), [colorbar.sig](#)

**Examples**

```
data(sw)

# GENERATING LAYOUT
mat <- matrix(1:2,ncol=2,nrow=1,byrow=TRUE)
l <- layout(mat,widths=c(5,1))

# CHOOSING LIMITS OF COLOUR RANGE
color.lim <- c(-2,2)

# PLOTTING
Mtmp <- v2m(maM(sw)[,1],Ngc=maNgc(sw),Ngr=maNgr(sw),Nsc=maNsc(sw),Nsr=maNsr(sw),
           visu=TRUE,color.lim=color.lim)
colorbar.mxy(color.lim=color.lim,ylablim=FALSE,ylab="M")
```

---

colorbar.mxy.abs      *Generates a colour bar*

---

**Description**

Generates colour bar for 2D plots of absolute values

**Usage**

```
colorbar.mxy.abs(color.lim,color="red", ylab="",ylablim=FALSE)
```

**Arguments**

color.lim	limits for colour range
color	colour to be used: “red” or “green”
ylab	label of y-axis
ylablim	If TRUE, the axis annotation consists only of the limits of the colour range.

**Details**

The function `colorbar.mxy.abs` is a modification of `colorbar.mxy` to provide colour-bars for MXY plots of absolute values. It is used in function `mxy.abs.plot`. Further details can be found at [colorbar.mxy](#).

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[mxy.abs.plot](#), [colorbar.mxy](#), [colorbar.sig](#)

---

colorbar.sig

*Generates a colour bar for spatial significance plots*

---

**Description**

This function generates a colour bar for the visualisation of significance of spatial bias.

**Usage**

```
colorbar.sig(color.lim=c(-3,3))
```

**Arguments**

color.lim	limits for color bar
-----------	----------------------

**Details**

The function `colorbar.sig` produces a colour bar for 2D-plots generated by `sigxy.plot`. The colours used range from green (for the lower limit of the colour range) to red (for its upper limit). For visualisation, values below or above the limits for the colour range (as given by `color.lim`) are set to the lower or upper limit, respectively. The function `colorbar.sig` is similar to more general function `colorbar.mxy`. It differs, however, in its axis annotation. Since it is used to present the significance in a log<sub>10</sub>-scale, annotation of axis ticks consists of negative values in both direction.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[sigxy.plot](#), [colorbar.mxy](#)

---

 fdr.int

 Assessment of the significance of intensity-dependent bias
 

---

**Description**

This function assesses the significance of intensity-dependent bias by an one-sided random permutation test. The observed average values of logged fold-changes within an intensity neighbourhood are compared to an empirical distribution generated by random permutation. The significance is given by the false discovery rate.

**Usage**

```
fdr.int(A,M,delta=50,N=100,av="median")
```

**Arguments**

A	vector of average logged spot intensity
M	vector of logged fold changes
delta	integer determining the size of the neighbourhood. The actual window size is (2 * delta+1).
N	number of random permutations performed for generation of empirical distribution
av	averaging of M within neighbourhood by <i>mean</i> or <i>median</i> (default)

**Details**

The function `fdr.int` assesses significance of intensity-dependent bias using a one-sided random permutation test. The null hypothesis states the independence of A and M. To test if M depends on A, spots are ordered with respect to A. This defines a neighbourhood of spots with similar A for each spot. Next, a test statistic is defined by calculating the *median* or *mean* of M within a symmetrical spot's intensity neighbourhood of chosen size (2 \* delta+1). An empirical distribution of the test statistic is produced by calculating for N random intensity orders of spots. Comparing this empirical distribution of  $\bar{M}$  with the observed distribution of  $\bar{M}$ , the independence of M and A is assessed. If M is independent of A, the empirical distribution of  $\bar{M}$  can be expected to be distributed around its mean value. The false discovery rate (FDR) is used to assess the significance of observing positive deviations of  $\bar{M}$ . It indicates the expected proportion of false positives among rejected null hypotheses. It is defined as  $FDR = q * T/s$ , where  $q$  is the fraction of  $\bar{M}$  larger than chosen threshold  $c$  for the empirical distribution,  $s$  is the number of neighbourhoods with  $\bar{M} > c$  for the distribution derived from the original data and  $T$  is the total number of neighbourhoods in the original data. Varying threshold  $c$  determines the FDR for each spot neighbourhood. FDRs equal zero are set to  $FDR = 1/T * N$  for computational reasons, as  $\log_{10}(FDR)$  is plotted by `sigint.plot`. Correspondingly, the significance of observing negative deviations of  $\bar{M}$  can be determined. If the neighbourhood window extends over the limits of the intensity scale, the significance is set to NA.

**Value**

A list of vector containing the false discovery rates for positive (FDRp) and negative (FDRn) deviations of  $\bar{M}$  (of the spot's neighbourhood) is produced.

**Note**

The same functionality but with our input and output formats is offered by [fdr.int](#)

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[fdr.int2](#), [p.int](#), [fdr.spatial](#), [sigint.plot](#)

**Examples**

```
# To run these examples, delete the comment signs (#) in front of the commands.
#
# LOADING DATA NOT-NORMALISED
# data(sw)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this example, N was chosen rather small. For "real" analysis, it should be larger.
# FDR <- fdr.int(maA(sw)[,1],maM(sw)[,1],delta=50,N=10,av="median")
# VISUALISATION OF RESULTS
# sigint.plot(maA(sw)[,1],maM(sw)[,1],FDR$FDRp,FDR$FDRn,c(-5,-5))

# LOADING NORMALISED DATA
# data(sw.olin)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# FDR <- fdr.int(maA(sw.olin)[,1],maM(sw.olin)[,1],delta=50,N=10,av="median")
# VISUALISATION OF RESULTS
# sigint.plot(maA(sw.olin)[,1],maM(sw.olin)[,1],FDR$FDRp,FDR$FDRn,c(-5,-5))
```

---

fdr.int2

*Assessment of the significance of intensity-dependent bias*

---

**Description**

This function assesses the significance of intensity-dependent bias by an one-sided random permutation test. The observed average values of logged fold-changes within an intensity neighbourhood are compared to an empirical distribution generated by random permutation. The significance is given by the false discovery rate.

**Usage**

```
fdr.int2(object,delta=50,N=100,av="median")
```

**Arguments**

object	object of class <code>marrayRaw</code> or <code>marrayNorm</code>
delta	integer determining the size of the neighbourhood. The actual window size is $(2 * \text{delta} + 1)$ .
N	number of random permutations performed for generation of empirical distribution
av	averaging of M within neighbourhood by <i>mean</i> or <i>median</i> (default)

**Details**

This function `fdr.int2` is basically the same as `fdr.int` except for differences in their in- and output format. For the details about the functionality, see [fdr.int](#).

**Note**

This function will be merged with `fdr.int` in future versions.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[fdr.int](#), [p.int2](#), [sigint.plot2](#)

**Examples**

```
# To run these examples, delete the comment signs (#) in front of the commands.
#
# LOADING DATA NOT-NORMALISED
# data(sw)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this example, N was chosen rather small. For "real" analysis, it should be larger.
# FDR <- fdr.int2(sw,delta=50,N=10,av="median")
# VISUALISATION OF RESULTS
# sigint.plot2(sw[,1],FDR$FDRp[[1]],FDR$FDRn[[1]],c(-5,-5)) # array 1
# sigint.plot2(sw[,4],FDR$FDRp[[4]],FDR$FDRn[[4]],c(-5,-5)) # array 4
```

---

fdr.spatial

*Assessment of the significance of spatial bias*

---

**Description**

This function assesses the significance of spatial bias by a one-sided random permutation test. This is achieved by comparing the observed average values of logged fold-changes within a spot's spatial neighbourhood with an empirical distribution generated by random permutation. The significance of spatial bias is given by the false discovery rate.

**Usage**

```
fdr.spatial(X,delta=2,N=100,av="median",edgeNA=FALSE)
```

**Arguments**

X	matrix of logged fold changes. For alternative input format, see <a href="#">fdr.spatial2</a> .
delta	integer determining the size of spot neighbourhoods $((2*\text{delta}+1)\times(2*\text{delta}+1))$ .
N	number of random permutations performed for generation of empirical background distribution
av	averaging of M within neighbourhood by <i>mean</i> or <i>median</i> (default)
edgeNA	treatment of edges of array: For edgeNA=TRUE, the significance of a neighbourhood (defined by a sliding window) is set to NA, if the neighbourhood extends over the edges of the matrix.

**Details**

The function `fdr.spatial` assesses the significance of spatial bias using a one-sided random permutation test. The null hypothesis states random spotting i.e. the independence of log ratio M and spot location. First, a neighbourhood of a spot is defined by a two dimensional square window of chosen size  $((2*\text{delta}+1)\times(2*\text{delta}+1))$ . Next, a test statistic is defined by calculating the *median* or *mean* of M within a symmetrical spot's neighbourhood. An empirical distribution of  $\bar{M}$  is generated based N random permutations of the spot locations on the array. The randomisation and calculation of  $\bar{M}$  is repeated N times. Comparing this empirical distribution of  $\bar{M}$  with the observed distribution of  $\bar{M}$ , the independence of M and spot location can be assessed. If M is independent of spot's location, the empirical distribution can be expected to be distributed around its mean value. To assess the significance of observing positive deviations of  $\bar{M}$ , the false discovery rate (*FDR*) is used. It indicates the expected proportion of false discoveries among rejected null hypotheses. It is defined as  $FDR = q * T / s$ , where  $q$  is the fraction of  $\bar{M}$  larger than chosen threshold  $c$  for the empirical distribution,  $s$  is the number of neighbourhoods with  $\bar{M} > c$  for the distribution derived from the original data and  $T$  is the total number of neighbourhoods on the array. FDRs equal zero are set to  $FDR = 1/T * N$ . Varying threshold  $c$  determines the FDR for each spot neighbourhood. Correspondingly, the significance of observing negative deviations of  $\bar{M}$  can be determined.

**Value**

A list of matrices containing the false discovery rates for positive (FDRp) and negative (FDRn) deviations of  $\bar{M}$  of the spot's neighbourhood is produced.

**Note**

The same functionality but with our input and output formats is offered by [fdr.spatial](#)

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[p.spatial](#), [fdr.int](#), [sigxy.plot](#), [fdr.spatial2](#)



## Examples

```
# To run these examples, delete the comment signs before the commands.
#
# LOADING DATA
# data(sw)
# M <- v2m(maM(sw)[,1],Ngc=maNgc(sw),Ngr=maNgr(sw),
#         Nsc=maNsc(sw),Nsr=maNsr(sw),main="MXY plot of SW-array 1")
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this illustration, N was chosen rather small. For "real" analysis, it should be larger.
# FDR <- fdr.spatial(M,delta=2,N=10,av="median",edgeNA=TRUE)
# sigxy.plot(FDR$FDRp,FDR$FDRn,color.lim=c(-5,5),main="FDR")
#
# LOADING NORMALISED DATA
# data(sw.olin)
# M<- v2m(maM(sw.olin)[,1],Ngc=maNgc(sw.olin),Ngr=maNgr(sw.olin),
#         Nsc=maNsc(sw.olin),Nsr=maNsr(sw.olin),main="MXY plot of SW-array 1")
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# FDR <- fdr.spatial(M,delta=2,N=10,av="median",edgeNA=TRUE)
# VISUALISATION OF RESULTS
# sigxy.plot(FDR$FDRp,FDR$FDRn,color.lim=c(-5,5),main="FDR")
```

---

fdr.spatial2

*Assessment of the significance of spatial bias*


---

## Description

This function assesses the significance of spatial bias by a one-sided random permutation test. This is achieved by comparing the observed average values of logged fold-changes within a spot's spatial neighbourhood with an empirical distribution generated by random permutation. The significance of spatial bias is given by the false discovery rate.

## Usage

```
fdr.spatial2(object,delta=2,N=100,av="median",edgeNA=FALSE)
```

## Arguments

object	object of class <code>marrayRaw</code> or <code>marrayNorm</code>
delta	integer determining the size of spot neighbourhoods $((2*\delta+1)\times(2*\delta+1))$ .
N	number of random permutations performed for generation of empirical background distribution
av	averaging of M within neighbourhood by <i>mean</i> or <i>median</i> (default)
edgeNA	treatment of edges of array: For <code>edgeNA=TRUE</code> , the significance of a neighbourhood (defined by a sliding window) is set to NA, if the neighbourhood extends over the edges of the matrix.

**Details**

The function `fdr.spatial2.Rd` is basically the same as `fdr.spatial`, but differs in its input and output formats. Details about the functionality can be found at [fdr.spatial](#).

**Value**

Two list of vectors containing the false discovery rates for positive (FDRp) and negative (FDRn) deviations of  $\bar{M}$  of the spot's neighbourhood is produced. Each vector contains the false discovery values for one array.

**Note**

This function will be fused with `fdr.spatial` in future versions using S4-style methods.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[p.spatial](#), [fdr.int](#), [sigxy.plot](#),

**Examples**

```
# To run these examples, delete the comment signs before the commands.
#
# LOADING DATA
# data(sw)
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this illustration, N was chosen rather small. For "real" analysis, it should be larger.
# FDR <- fdr.spatial2(sw,delta=2,N=10,av="median",edgeNA=TRUE)
#
# SIGNIFICANCE PLOTS OF ARRAY 1
# sigxy.plot2(sw[,1],FDR$FDRp[[1]],FDR$FDRn[[1]],color.lim=c(-5,5),main="FDR")
# SIGNIFICANCE PLOTS OF ARRAY 3
# sigxy.plot2(sw[,3],FDR$FDRp[[3]],FDR$FDRn[[3]],color.lim=c(-5,5),main="FDR")
#
```

---

fgbg.visu

*Visualisation of foreground and background fluorescence spot intensities in both channels*

---

**Description**

This function generates 2D-plots of the foreground, background and background corrected fluorescence intensities of channel 1 and of channel 2, respectively.

**Usage**

```
fgbg.visu(obj, label)
```

**Arguments**

obj	object of class “marrayRaw”
label	character string for labelling. It will be added to the title of the first sub-plot.

**Details**

The function `fgbg.visu` produces 2D-representations of the foreground and background intensities for both fluorescence channels (as stored in `obj`). Additionally, a plot of the difference between fore- and background intensities is generated (*background-corrected intensities*). All intensities are log<sub>2</sub>-transformed. The colour range for plotting is defined by 0 and the maximum of the logged intensity for each sub-graph separately.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[marrayRaw](#)

**Examples**

```
# LOADING RAW DATA
data(sw)
# PLOTTING
fgbg.visu(sw[,1])
```

---

ino

*Intensity-dependent normalisation of two-colour microarrays*

---

**Description**

This functions performs intensity-dependent normalisation based on local regression by `locfit`.

**Usage**

```
ino(object, alpha=0.3, weights=NA, bg.corr="subtract", ...)
```

**Arguments**

object	object of class “marrayRaw” or “marrayNorm”
alpha	smoothing parameter
weights	matrix of weights for local regression. Rows correspond to the spotted probe sequences, columns to arrays in the batch. These may be derived from the matrix of spot quality weights as defined for “maRaw” objects.
bg.corr	backcorrection method (for “marrayRaw” objects): “none” or “subtract”(default).
...	Further arguments for locfit function.

**Details**

The function `ino` regresses the average logged fold changes ( $M$ ) with respect to the average logged spot intensity ( $A$ ). The residuals of this fit are the normalised logged fold changes. The parameter `alpha` specifies the fraction of points that are included in the neighbourhood and thus has a value between 0 and 1. Larger `alpha` values lead to smoother fits.

**Value**

Object of class “marrayNorm” with normalised logged ratios

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[maNorm](#), [locfit.raw](#), [olin](#), [oin](#), [lin](#)

**Examples**

```
# LOADING DATA
data(sw)

# INTENSITY-DEPENDENT NORMALISATION
norm.ino <- ino(sw)

# MA-PLOT OF NORMALISATION RESULTS OF FIRST ARRAY
plot(maA(norm.ino)[,1],maM(norm.ino)[,1],main="INO")

# CORRESPONDING MXY-PLOT
mxy.plot(maM(norm.ino)[,1],Ngc=maNgc(norm.ino),Ngr=maNgr(norm.ino),
          Nsc=maNsc(norm.ino),Nsr=maNsr(norm.ino),main="INO")
```

---

lin *Local intensity-dependent normalisation of two-colour microarrays*

---

### Description

This function performs local intensity-dependent normalisation (LIN)

### Usage

```
lin(object, X=NA, Y=NA, alpha=0.3, iter=2, scale=TRUE, weights=NA, bg.corr="subtract", ...)
```

### Arguments

object	object of class “marrayRaw”
X	matrix with x-coordinates of spots. If X=NA, columns on array are used as proxies for the location in x-direction
Y	matrix with y-coordinates of spots. If Y=NA, rows on array are used as proxies for the location in y-direction
alpha	smoothing parameter for local regression
iter	number of iterations in the LIN procedure
scale	scale parameter for smoothing in Y-direction of the array in respect to smoothing in X-direction. If scale=TRUE, standard deviations are used.
weights	matrix of weights for local regression. Rows correspond to the spotted probe sequences, columns to arrays in the batch. These may be derived from the matrix of spot quality weights as defined for “maRaw” objects.
bg.corr	backcorrection method (for “marrayRaw” objects): “none” or “subtract”(default).
...	Further arguments for <code>locfit</code> function.

### Details

LIN is based on the same normalisation scheme as OLIN, but does not incorporate optimisation of model parameters. The function `lin` can serve for comparison. Alternatively, it can be used to enforce a conservative model fit.

The smoothing parameter `alpha` controls the neighbourhood size  $h$  of local fitting. It specifies the fraction of points that are included in the neighbourhood and, thus, has a value between 0 and 1. Larger `alpha` values lead to smoother fits.

If the normalisation should be based on set of genes assumed to be not differentially expressed (*house-keeping genes*), weights can be used for local regression. In this case, all weights should be set to zero except for the house-keeping genes for which weights are set to one. In order to achieve a reliable regression, it is important, however, that there is a sufficient number of house-keeping genes that cover the whole expression range and are spotted across the whole array.

### Value

Object of class “marrayNorm” with normalised logged ratios

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**References**

1. M.Futschik and T.Crompton (2004) *Model selection and efficiency testing for normalization of cDNA microarray data*, **Genome Biology**, 5:R60

**See Also**

[maNorm](#), [locfit](#), [olin](#), [oin](#)

**Examples**

```
# LOADING DATA
data(sw)
data(sw.xy)

# LOCAL INTENSITY-DEPENDENT NORMALISATION
norm.lin <- lin(sw,X=sw.xy$X,Y=sw.xy$Y)

# MA-PLOT OF NORMALISATION RESULTS OF FIRST ARRAY
plot(maA(norm.lin)[,1],maM(norm.lin)[,1],main="LIN")

# CORRESPONDING MXY-PLOT
mxy.plot(maM(norm.lin)[,1],Ngc=maNgc(norm.lin),Ngr=maNgr(norm.lin),
         Nsc=maNsc(norm.lin),Nsr=maNsr(norm.lin),main="LIN")
```

---

m2v

*Converts matrix to vector based on spot layout*


---

**Description**

This function converts a matrix based on the spatial layout of spots to a vector. Optionally, a 2D-plot is produced.

**Usage**

```
m2v(M,Ngc,Ngr,Nsc,Nsr,visu=FALSE,color.lim=c(-1,1),xlab="Columns",ylab="Rows",...)
```

## Arguments

M	Matrix of real values
Ngc	number of columns for the grid matrix
Ngr	number of rows for the grid matrix
Nsc	number of columns for the spot matrix
Nsr	number of rows for the spot matrix
visu	If TRUE, MXY plot is generated.
color.lim	limits of colour range for 2D-plot
xlab	label of x -axis of 2D-plot
ylab	label of y-axis of 2D-plot
...	Further optional parameters for the image function generating the MXY plot

## Details

The function `m2v` rearranges the values of a matrix `M` corresponding to the intensity values on the array to a vector `V`. The matrix `M` may have been generated by e.g. `v2m`. The order of values in `V` follows the convention of *marrayRaw* objects. In fact, the transformation of `m2v` is the reverse of `v2m` (assuming the arguments are kept the same.) Note that these functions assume a specific mapping between the data points and the location of spot (i.e. the same mapping rule that is used for *marrayRaw*/*marrayNorm* objects.) The validity of the mappings should be carefully checked (see also the documentation of the *marray* package.) The option for spatial visualisation is rather restricted to logged fold-changes as the corresponding colour range is centred around zero and follows the conventional colouring (green for negative, red for positive fold-changes). The MXY plot produced by `m2v` does not include a colour-bar. To have a colour included, `mxy.plot` can be used.

## Value

A vector `V` of length  $((Ngc*Nsc)*(Ngr*Nsr))$  is produced. The values of `V` represents the spatial distribution of the values of vector `V` given the print-layout. Optionally, a 2D-plot of `M` is generated.

## Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

## See Also

[v2m](#), [mxy.plot](#)

## Examples

```
# LOADING DATA NOT-NORMALISED
data(sw)
# CONVERSION FROM VECTOR TO MATRIX
M <- v2m(maM(sw)[,1],Ngc=maNgc(sw),Ngr=maNgr(sw),Nsc=maNsc(sw),Nsr=maNsr(sw),visu=TRUE)

# BACK-CONVERSION FROM MATRIX TO VECTOR
V <- m2v(M,Ngc=maNgc(sw),Ngr=maNgr(sw),Nsc=maNsc(sw),Nsr=maNsr(sw),visu=TRUE)
```

---

ma.matrix

*Calculation of moving average for a matrix*


---

### Description

Using a sliding square window this function produces the moving average for a matrix.

### Usage

```
ma.matrix(X,av="median",delta= 2,edgeNA=FALSE )
```

### Arguments

X	matrix
av	averaging by <i>mean</i> or <i>median</i> (default)
delta	integer determining the size of the sliding square window $(2*\text{delta}+1)\times(2*\text{delta}+1)$ .
edgeNA	treatment of edges of array: For edgeNA=TRUE, averaged values of sliding windows are set to NA if the corresponding windows extend over the edges of the matrix.

### Details

A square window with size  $(2*\text{delta}+1)\times(2*\text{delta}+1)$  is moved over the entire matrix and a new matrix is created with each value equals the average value in the corresponding window. This procedure defines a local regression of zeroth order.

### Value

Matrix with average values of matrix X.

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### See Also

ma.vector

### Examples

```
### LOADING DATA
data(sw)

### GENERATION OF MATRIX
Morig <- v2m(maM(sw)[,1],Ngc=maNgc(sw),Ngr=maNgr(sw),Nsc=maNsc(sw),Nsr=maNsr(sw),visu=TRUE)

### AVERAGING BY MA.MATRIX
Mav <- ma.matrix(Morig,av="median",delta= 2,edgeNA=FALSE )
```



```
### VISUALISATION
m2v(Mav,Ngc=maNgc(sw),Ngr=maNgr(sw),Nsc=maNsc(sw),Nsr=maNsr(sw),visu=TRUE)
```

---

ma.vector

*Calculation of moving average for a vector*

---

## Description

This functions calculates the moving average for a vector.

## Usage

```
ma.vector(A,M,av="median",delta=50)
```

## Arguments

A	vector of predictor to be used for sorting
M	vector of variable to be averaged
av	averaging by <i>mean</i> or <i>median</i> (default)
delta	even integer determining the size of the sliding window ( $2*\delta+1$ .)

## Details

The function `ma.vector` first sorts `M` according to the corresponding values of `A`. Subsequently, a moving average is calculated with window size  $(2*\delta+1)$ . The values for the moving average are set to zero if the corresponding window extends over the boarder of the vector `M`.

## Value

Vector with moving average values of `M`

## Author(s)

Matthias E. Futschik,<http://itb.biologie.hu-berlin.de/~futschik>

## See Also

[ma.matrix](#)

**Examples**

```

### LOADING DATA
data(sw)
A <- maA(sw[,1])
M <- maM(sw[,1])

# MA-PLOT
plot(A,M)

# MOVING AVERAGE
Mav <- ma.vector(A,M,av="median",delta=100)
points(A,Mav,col="red")

```

---

mxy.abs.plot

*Generation of MXY plots of absolute values*


---

**Description**

This function produce a MXY plot of absolute values of M including a colour bar.

**Usage**

```
mxy.abs.plot(V,Ngc,Ngr,Nsc,Nsr,color.lim,color="red",xlab="Columns",ylab="Rows",...)
```

**Arguments**

V	vector of positive values
Ngc	number of columns for the grid matrix
Ngr	number of rows for the grid matrix
Nsc	number of columns for the spot matrix
Nsr	number of rows for the spot matrix
color.lim	limits of color range for MXY plot
color	color to be used for plot: "red" (default) or "green"
xlab	label of x -axis of MXY plot
ylab	label of y-axis of MXY plot
...	Further optional graphical parameter for the image function generating the MXY plot

**Details**

The function `mxy.abs.plot` is similar to function `mxy.plot`. Details can therefore be found at [mxy.plot](#). Two differences, however, exist: First, `mxy.abs.plot` plots the absolute value of V and second, "red" (default) or "green" can be chosen as colour of plotting. Hence, `mxy.abs.plot` facilitates the inspection of spatial artifacts in single fluorescence channels.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[v2m](#), [m2v](#), [colorbar.mxy.abs](#), [fgbg.visu](#), [image](#)

**Examples**

```
# LOADING DATA
data(sw)
# PLOTTING OF ABSOLUTE LOGGED FOLD-CHANGES
mxy.plot(abs(maM(sw)[,1]),Ngc=maNgc(sw),Ngr=maNgr(sw),Nsc=maNsc(sw),Nsr=maNsr(sw))

# PLOTTING SPATIAL DISTRIBUTION OF SINGLE-CHANNEL INTENSITIES
mxy.abs.plot(maRf(sw)[,1],color.lim=c(0,10000),Ngc=maNgc(sw),Ngr=maNgr(sw),
             Nsc=maNsc(sw),Nsr=maNsr(sw))
mxy.abs.plot(maGf(sw)[,1],color.lim=c(0,10000),color="green",Ngc=maNgc(sw),Ngr=maNgr(sw),
             Nsc=maNsc(sw),Nsr=maNsr(sw))
```

---

mxy.plot

*Generation of MXY plots*

---

**Description**

This function produce a MXY plot including a colour bar.

**Usage**

```
mxy.plot(V,Ngc,Ngr,Nsc,Nsr,color.lim=c(-1,1),xlab="Columns",ylab="Rows",...)
```

**Arguments**

V	vector of real values typically logged ratios <i>M</i> . Alternatively, V can be an object of class <code>marrayRaw</code> or <code>marrayNorm</code> . In this case, the layout of the array does not need to be given.
Ngc	number of columns for the grid matrix
Ngr	number of rows for the grid matrix
Nsc	number of columns for the spot matrix
Nsr	number of rows for the spot matrix
color.lim	limits of color range for MXY plot
xlab	label of x-axis of MXY plot
ylab	label of y-axis of MXY plot
...	Further optional graphical parameter for the image function generating the MXY plot

## Details

Spotted microarrays have generally a grid layout of form with  $N_{gc}$  columns and  $N_{gr}$  rows. Each block (or spot matrix) of the grid corresponds to a specific pin used for spotting. The blocks have generally  $N_{sc}$  columns and  $N_{sr}$  rows. The function `mxy.plot` generates a 2D-plot (MXY-plot) of the values of  $M$  across the array.  $M$  is given in form of the vector  $V$ . Note that this function assumes a specific mapping between the data points and the location of spot (i.e. the same mapping rule that is used for `marrayRaw`/`marrayNorm` objects (see the documentation of packet `marray`) The colour range of the MXY plot is centred around zero and follows the conventional colouring (green for negative, red for positive fold-changes). For a separate visualisation of the two channels, see function `fgbg.visu`.

## Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

## See Also

[v2m](#), [m2v](#), [fgbg.visu](#), [image](#), [marrayRaw](#)

## Examples

```
# LOADING DATA
data(sw)

# PLOTTING
mxy.plot(maM(sw)[,1],Ngc=maNgc(sw),Ngr=maNgr(sw), Nsc=maNsc(sw),Nsr=maNsr(sw))

# ALTERNATIVE
mxy.plot(sw[,1])
```

---

`mxy2.plot`

*Generation of MXY plots based on spot coordinates*

---

## Description

This function produce a MXY plot with a colour bar. In contrast to `mxy.plot`, the plot is based on spot coordinates (instead on column and row index as proxies for spot location).

## Usage

```
mxy2.plot(V,X,Y,Ngc,Ngr,Nsc,Nsr,color.lim=c(-1,1),xlab="X",ylab="Y",...)
```

**Arguments**

V	vector of real values typically logged ratios $M$ .
X	vector of x coordinates of spot locations
Y	vector of y coordinates of spot locations
Ngc	number of columns for the grid matrix
Ngr	number of rows for the grid matrix
Nsc	number of columns for the spot matrix
Nsr	number of rows for the spot matrix
color.lim	limits of color range for MXY plot
xlab	label of x-axis of MXY plot
ylab	label of y-axis of MXY plot
...	Further optional graphical parameter for the image function generating the MXY plot

**Details**

The function `mxy2.plot` can be used to plot the distribution of  $V$  across the array. As `mxy.plot`, it mainly aims for the plotting of the distribution of logged fold changes. It differs from `mxy.plot` in the representation of spot location. The function `mxy.plot` uses the index of columns and rows as proxies for the spot location. The gaps between the grid matrices (spotted by different pins) are, therefore, not reproduced in the plot. A more accurate spatial plot is produced by `mxy2.plot`, which is based on the coordinates of the first column and first row of the array. Assuming a regular rectangular print layout, gaps and the edges of the array are shown.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[mxy.plot](#), [v2m](#), [m2v](#), [fgbg.visu](#), [image](#)

**Examples**

```
# LOADING DATA
data(sw)
data(sw.xy)
# PLOTTING
mxy2.plot(maM(sw)[,1], X=sw.xy$X[,1], Y=sw.xy$Y[,1], Ngc=maNgc(sw), Ngr=maNgr(sw),
          Nsc=maNsc(sw), Nsr=maNsr(sw))
```

---

oin	<i>Optimised intensity-dependent normalisation of two-colour microarrays</i>
-----	--

---

### Description

This function performs optimised intensity-dependent normalisation (OLIN).

### Usage

```
oin(object, alpha=seq(0.1, 1, 0.1), weights=NA, bg.corr="subtract", ...)
```

### Arguments

object	object of class “marrayRaw” or “marrayNorm”
alpha	vector of alpha parameters that are tested in the GCV procedure
weights	matrix of weights for local regression. Rows correspond to the spotted probe sequences, columns to arrays in the batch. These may be derived from the matrix of spot quality weights as defined for “marrayRaw” objects.
bg.corr	backcorrection method (for “marrayRaw” objects): “none” or “subtract”(default).
...	Further arguments for <code>locfit</code> function.

### Details

The function `oin` is based on iterative local regression of logged fold changes in respect to average logged spot intensities. It incorporates optimisation of the smoothing parameter  $\alpha$  that controls the neighbourhood size  $h$  of local fitting. The parameter  $\alpha$  specifies the fraction of points that are included in the neighbourhood and thus has a value between 0 and 1. Larger  $\alpha$  values lead to smoother fits.

If the normalisation should be based on set of genes assumed to be not differentially expressed (*house-keeping genes*), weights can be used for local regression. In this case, all weights should be set to zero except for the house-keeping genes for which weights are set to one. In order to achieve a reliable regression, it is important, however, that there is a sufficient number of house-keeping genes that are distributed over the whole expression range and spotted across the whole array.

In contrast to OLIN and OSLIN, the OIN scheme does not correct for spatial dye bias. It can, therefore, be used if the assumption of random spotting does not hold.

### Value

Object of class “marrayNorm” with normalised logged ratios

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[maNorm](#), [locfit](#), [gcv](#), [olin](#), [lin](#), [ino](#)

**Examples**

```
# LOADING DATA
data(sw)

# OPTIMISED INTENSITY-DEPENDENT NORMALISATION
norm.oin <- oin(sw)

# MA-PLOT OF NORMALISATION RESULTS OF FIRST ARRAY
plot(maA(norm.oin)[,1],maM(norm.oin)[,1],main="OIN")

# CORRESPONDING MXY-PLOT
mxy.plot(maM(norm.oin)[,1],Ngc=maNgc(norm.oin),Ngr=maNgr(norm.oin),
          Nsc=maNsc(norm.oin),Nsr=maNsr(norm.oin),main="OIN")

#
```

---

olin	<i>Optimised local intensity-dependent normalisation of two-colour microarrays</i>
------	--

---

**Description**

This functions performs optimised local intensity-dependent normalisation (OLIN) and optimised scaled intensity-dependent normalisation (OSLIN).

**Usage**

```
olin(object,X=NA,Y=NA,alpha=seq(0.1,1,0.1),iter=3,
      scale=c(0.05,0.1,0.5,1,2,10,20),OSLIN=FALSE,weights=NA,
      genepix=FALSE,bg.corr="subtract",...)
```

**Arguments**

object	object of class “marrayRaw” or “marrayNorm” corresponding to a single array or a batch of arrays.
X	matrix with x-coordinates of spots of the arrays in object. Each column includes the x-coordinates for the spots of one array. If X=NA, columns on array are used as proxies for the location in x-direction
Y	matrix with y-coordinates of spots. Each column includes the y-coordinates for the spots of one array.If Y=NA, rows on array are used as proxies for the location in y-direction
alpha	vector of alpha parameters that are tested in the GCV procedure

<code>iter</code>	number of iterations in the OLIN procedure
<code>scale</code>	vector of scale parameters that are tested in a GCV procedure for spatial regression. This define the amount of smoothing in X-direction with respect to smoothing in Y-direction.
<code>OSLIN</code>	If <code>OSLIN=TRUE</code> , subsequent scaling of the range of M across the array is performed.
<code>weights</code>	matrix of (non-negative) weights for local regression (see <code>locfit</code> ). Rows correspond to the spotted probe sequences, columns to arrays in the batch. If the weight of the corresponding spot equals zero, the spot is not used in the normalisation procedures (except the <code>genepix</code> argument is set to <code>TRUE</code> .) If the weight matrix include negative values, these will be set to zero. These weight matrix may be derived from the matrix of spot quality weights as defined for “maRaw” objects ( <code>weights=maW(object)</code> ). Weights can be also used if the normalisation should be based on a set of selected genes that are assumed to be not differentially expressed.
<code>genepix</code>	If <code>genepix</code> is set to <code>TRUE</code> , spot weights equal zero or larger are set to one for the local regression whereas negative spot with negative weights are not used for the regression. The argument <code>genepix</code> should be set to <code>TRUE</code> , if <code>weights=maW(object)</code> is set and spot quality weights derived by <code>GenePix</code> are stored in <code>maW(object)</code> .
<code>bg.corr</code>	background correction method (for “marrayRaw” objects) : “none”, “subtract”, “half”, “minimum”, “movingmin”, “edwards” or “normexp”.
<code>...</code>	Further arguments for <code>locfit</code> function.

## Details

OLIN and OSLIN are based on iterative local regression and incorporate optimisation of model parameters. Local regression is performed using LOCFIT, which requires the user to choose a specific smoothing parameter  $\alpha$  that controls the neighbourhood size  $h$  of local fitting. The parameter  $\alpha$  specifies the fraction of points that are included in the neighbourhood and thus has a value between 0 and 1. Larger  $\alpha$  values lead to smoother fits. Additionally, the setting of scale parameters controls for distinct amount of smoothing in Y-direction compared to smoothing in X-direction. The parameter `scale` can be of arbitrary value. The choice of model parameters  $\alpha$  and `scale` for local regression is crucial for the efficiency and quality of normalization. To optimize the model parameters, a general cross-validation procedure (GCV) is applied. The arguments `alpha` and `scale` define the parameters values which are tested in the GCV. OSLIN comprises the OLIN procedure with a subsequent optimized scaling of the range of logged intensity ratios across the spatial dimensions of the array. Details concerning the background correction methods can be found in the help page for [backgroundCorrect2](#).

Detailed information about OLIN and OSLIN can be found in the package documentation and in the reference stated below. The `weights` argument specifies the influence of the single spots on the local regression. To exclude spots being used for the local regression (such as control spots), set their corresponding weight to zero. Note that OLIN and OSLIN are based on the assumptions that most genes are not differentially expressed (or up- and down-regulation is balanced) and that genes are randomly spotted across the array. If these assumptions are not valid, local regression can lead to an underestimation of differential expression. OSLIN is especially sensitive to violations of these assumptions. However, this sensitivity can be decreased if the minimal  $\alpha$ -value is increased. Minimal  $\alpha$  defines the smallest scale used for local regression. Increasing  $\alpha$  can



reduce the influence of localised artifacts as a larger fraction of data points is included. Alternative normalisation functions such as `oin`, `lin` and `ino` might also be used for a more conservative fit.

If the normalisation should be based on set of genes assumed to be not differentially expressed (*house-keeping genes*), weights can be used for local regression. In this case, all weights are set to zero except for the house-keeping genes for which weights are set to one. In order to achieve a reliable regression, it is important, however, that there is a sufficient number of house-keeping genes that are distributed over the whole expression range and spotted across the whole array.

It is also important to note that OLIN/OSLIN is fairly efficient in removing intensity- and spatial-dependent dye bias, so that normalised data will look quite “good” after normalisation independently of the true underlying data quality. Normalisation by local regression assumes smoothness of bias. Therefore, localised artifacts such as scratches, edge effects or bubbles should be avoided. Spots of these areas should be flagged (before normalisation is applied) to ensure data integrity. To stringently detect artifacts, the OLIN functions `fdr.int`, `fdr.spatial`, `p.int` and `p.spatial` can be used.

### Value

Object of class “marrayNorm” with normalised logged ratios

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### References

1. M.Futschik and T.Crompton (2004) *Model selection and efficiency testing for normalization of cDNA microarray data*, **Genome Biology**, 5:R60
2. M.Futschik and T.Crompton (2005) *OLIN: Optimized normalization, visualization and quality testing for two-channel microarray data*, **Bioinformatics**, 21(8):1724-6
3. OLIN web-page: <http://itb.biologie.hu-berlin.de/~futschik/software/R/OLIN>

### See Also

`maNorm`, `locfit`, `gcv`, `oin`, `lin`

### Examples

```
# LOADING DATA
data(sw)
data(sw.xy)

# OPTIMISED LOCAL INTENSITY-DEPENDENT NORMALISATION OF FIRST ARRAY
norm.olin <- olin(sw[,1],X=sw.xy$X[,1],Y=sw.xy$Y[,1])

# MA-PLOT OF NORMALISATION RESULTS OF FIRST ARRAY
plot(maA(norm.olin),maM(norm.olin),main="OLIN")

# CORRESPONDING MXY-PLOT
```

```

mxy.plot(maM(norm.olin)[,1],Ngc=maNgc(norm.olin),Ngr=maNgr(norm.olin),
         Nsc=maNsc(norm.olin),Nsr=maNsr(norm.olin),main="OLIN")

# OPTIMISED SCALED LOCAL INTENSITY-DEPENDENT NORMALISATION
norm.oslin <- olin(sw[,1],X=sw.xy$X[,1],Y=sw.xy$Y[,1],OSLIN=TRUE)
# MA-PLOT
plot(maA(norm.oslin),maM(norm.oslin),main="OSLIN")
# MXY-PLOT
mxy.plot(maM(norm.oslin)[,1],Ngc=maNgc(norm.oslin),Ngr=maNgr(norm.oslin),
         Nsc=maNsc(norm.oslin),Nsr=maNsr(norm.oslin),main="OSLIN")

```

---

p.int

*Calculates significance of intensity-dependent bias*


---

### Description

This function assesses the significance of intensity-dependent bias. This is achieved by comparing the observed average values of logged fold-changes within an intensity neighbourhood with an empirical distribution generated by permutation tests. The significance is given by (adjusted) p-values.

### Usage

```
p.int(A,M,delta=50,N=-1,av="median",p.adjust.method="none")
```

### Arguments

A	vector of average logged spot intensity
M	vector of logged fold changes
delta	integer determining the size of the neighbourhood ( $2 * \text{delta} + 1$ ).
N	number of random samples (of size $2 * \text{delta} + 1$ ) used for the generation of empirical distribution. If N is negative, the number of samples 100 times the length of A.
av	averaging of M within neighbourhood by <i>mean</i> or <i>median</i> (default)
p.adjust.method	method for adjusting p-values due to multiple testing regime. The available methods are “none”, “bonferroni”, “holm”, “hochberg”, “hommel” and “fdr”. See also <a href="#">p.adjust</a>

### Details

The function `p.int` assesses the significance of intensity-dependent bias using a permutation test. The null hypothesis states the independence of A and M. To test if M depends on A, spots are ordered with respect to A. This defines a neighbourhood of spots with similar A for each spot. Next, the test statistic is the *median* or *mean* of M within a spot's intensity neighbourhood of chosen size ( $2 * \text{delta} + 1$ ). The empirical distribution of the this statistic is then generated based on N random

samples (with replacement). (Note that sampling without replacement is used for `fdr.int`. Also note, that different meaning of argument `N` in `p.int` and `fdr.int`. The argument `N` in `p.int` is the number of independent samples (of size  $2 * \text{delta} + 1$ ) derived from the original distribution. The argument `N` in `fdr.int` states how many times the original distribution is randomised and the permuted distribution is used for generating the empirical distribution.) Comparing this empirical distribution of  $\bar{M}$  with the observed distribution of  $\bar{M}$ , the independence of `M` and `A` is assessed. If `M` is independent of `A`, the empirical distribution of  $\bar{M}$  can be expected to be symmetrically distributed around its mean value. To assess the significance of observing positive deviations of the p-values are used. It indicates the expected proportion of neighbourhoods with larger  $\bar{M}$  than the actual one based on the empirical distribution of  $\bar{M}$ . The minimal p-value is set to  $1/N$ . Correspondingly, the significance of observing negative deviations of  $\bar{M}$  can be determined. Since this assessment of significance involves multiple testing, an adjustment of the p-values might be advisable.

### Value

A list of vector containing the p-values for positive (`Pp`) and negative (`Pn`) deviations of  $\bar{M}$  of the spot's neighbourhood is produced. Values corresponding to spots within an interval of `delta` at the lower or upper end of the `A`-scale are set to `NA`.

### Note

The same functionality but with our input and output formats is offered by [p.int2](#)

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### See Also

[p.int2](#), [fdr.int](#), [sigint.plot](#), [p.adjust](#)

### Examples

```
# To run these examples, "un-comment" them!
#
# LOADING DATA NOT-NORMALISED
# data(sw)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this illustration, N was chosen rather small. For "real" analysis, it should be larger.
# P <- p.int(maA(sw)[,1],maM(sw)[,1],delta=50,N=10000,av="median",p.adjust.method="none")
# VISUALISATION OF RESULTS
# sigint.plot(maA(sw)[,1],maM(sw)[,1],Sp=P$Pp,Sn=P$Pn,c(-5,-5))

# LOADING NORMALISED DATA
# data(sw.olin)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# P <- p.int(maA(sw.olin)[,1],maM(sw.olin)[,1],delta=50,N=10000,av="median",p.adjust.method="none")
# VISUALISATION OF RESULTS
# sigint.plot(maA(sw.olin)[,1],maM(sw.olin)[,1],Sp=P$Pp,Sn=P$Pn,c(-5,-5))
```

---

p.int2

*Calculates significance of intensity-dependent bias*

---

### Description

This function assesses the significance of intensity-dependent bias. This is achieved by comparing the observed average values of logged fold-changes within an intensity neighbourhood with an empirical distribution generated by permutation tests. The significance is given by (adjusted) p-values.

### Usage

```
p.int2(object,delta=50,N=-1,av="median",p.adjust.method="none")
```

### Arguments

object	object of class <code>marrayRaw</code> or <code>marrayNorm</code>
delta	integer determining the size of the neighbourhood ( $2 * \text{delta} + 1$ ).
N	number of random samples (of size $2 * \text{delta} + 1$ ) used for the generation of empirical distribution. If N is negative, the number of samples 100 times the length of A.
av	averaging of M within neighbourhood by <i>mean</i> or <i>median</i> (default)
p.adjust.method	method for adjusting p-values due to multiple testing regime. The available methods are "none", "bonferroni", "holm", "hochberg", "hommel" and "fdr". See also <a href="#">p.adjust</a>

### Details

This function `p.int2` is basically the same as `p.int` except for differences in their in- and output format. For the details about the functionality, see [p.int](#).

### Note

This function will be merged with `p.int` in future versions.

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### See Also

[p.int](#), [fdr.int2](#), [sigint.plot2](#), [p.adjust](#)

## Examples

```
# To run these examples, "un-comment" them!
#
# LOADING DATA NOT-NORMALISED
# data(sw)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this illustration, N was chosen rather small. For "real" analysis, it should be larger.
# P <- p.int2(sw,delta=50,N=10000,av="median",p.adjust.method="none")
# VISUALISATION OF RESULTS
# sigint.plot2(sw[,1],Sp=P$Pp[[1]],Sn=P$Pn[[1]],c(-5,-5)) # array 1
# sigint.plot2(sw[,3],Sp=P$Pp[[3]],Sn=P$Pn[[3]],c(-5,-5)) # array 3
```

---

p.spatial

*Assessment of the significance of spatial bias based on p-values*


---

## Description

This function assesses the significance of spatial bias. This is achieved by comparing the observed average values of logged fold-changes within a spot's spatial neighbourhood with an empirical distribution generated by permutation tests. The significance is given by (adjusted) p-values derived in one-sided permutation test.

## Usage

```
p.spatial(X,delta=2,N=-1,av="median",p.adjust.method="none")
```

## Arguments

X	matrix of logged fold changes
delta	integer determining the size of spot neighbourhoods $((2 \times \text{delta} + 1) \times (2 \times \text{delta} + 1))$ .
N	number of samples for generation of empirical background distribution
av	averaging of M within neighbourhood by <i>mean</i> or <i>median</i> (default)
p.adjust.method	method for adjusting p-values due to multiple testing regime. The available methods are "none", "bonferroni", "holm", "hochberg", "hommel" and "fdr". See also <a href="#">p.adjust</a> .

## Details

The function `p.spatial` assesses the significance of spatial bias using an one-sided random permutation test. The null hypothesis states random spotting i.e. the independence of log ratio M and spot location. First, a neighbourhood of a spot is defined by a two dimensional square window of chosen size  $((2 \times \text{delta} + 1) \times (2 \times \text{delta} + 1))$ . Next, a test statistic is defined by calculating the *median* or *mean* of M for N random samples of size  $((2 \times \text{delta} + 1) \times (2 \times \text{delta} + 1))$ . Note that this scheme defines a sampling with replacement procedure whereas sampling without replacement is used for

`fdr.spatial`. Comparing the empirical distribution of  $\bar{M}$  with the observed distribution of  $\bar{M}$ , the independence of  $M$  and spot location can be assessed. If  $M$  is independent of spot's location, the empirical distribution can be expected to be distributed around its mean value. To assess the significance of observing positive deviations of  $\bar{M}$ , p-values are calculated using Fisher's method. The p-value equals the fraction of values in the empirical distribution which are larger than the observed value. The minimal p-value is set to  $1/N$ . Correspondingly, the significance of observing negative deviations of  $\bar{M}$  can be determined.

### Value

A list of vectors containing the p-values for positive (Pp) and negative (Pn) deviations of  $\bar{M}$  of the spot's neighbourhood is produced.

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### See Also

[fdr.int](#), [sigxy.plot](#), [p.adjust](#)

### Examples

```
# To run these examples, "un-comment" them!
#
# LOADING DATA
# data(sw)
# M <- v2m(maM(sw)[,1],Ngc=maNgc(sw),Ngr=maNgr(sw),
#         Nsc=maNsc(sw),Nsr=maNsr(sw),main="MXY plot of SW-array 1")
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this illustration, N was chosen rather small. For "real" analysis, it should be larger.
# P <- p.spatial(M,delta=2,N=10000,av="median")
# sigxy.plot(P$Pp,P$Pn,color.lim=c(-5,5),main="FDR")

# LOADING NORMALISED DATA
# data(sw.olin)
# M <- v2m(maM(sw.olin)[,1],Ngc=maNgc(sw.olin),Ngr=maNgr(sw.olin),
#         Nsc=maNsc(sw.olin),Nsr=maNsr(sw.olin),main="MXY plot of SW-array 1")

# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# P <- p.spatial(M,delta=2,N=10000,av="median")
# VISUALISATION OF RESULTS
# sigxy.plot(P$Pp,P$Pn,color.lim=c(-5,5),main="FDR")
```

p.spatial2

Assessment of the significance of spatial bias based on p-values

**Description**

This function assesses the significance of spatial bias. This is achieved by comparing the observed average values of logged fold-changes within a spot's spatial neighbourhood with an empirical distribution generated by permutation tests. The significance is given by (adjusted) p-values derived in one-sided permutation test.

**Usage**

```
p.spatial2(object,delta=2,N=-1,av="median",p.adjust.method="none")
```

**Arguments**

object	object of class <code>marrayRaw</code> or <code>marrayNorm</code>
delta	integer determining the size of spot neighbourhoods $((2 \times \text{delta} + 1) \times (2 \times \text{delta} + 1))$ .
N	number of samples for generation of empirical background distribution
av	averaging of M within neighbourhood by <i>mean</i> or <i>median</i> (default)
p.adjust.method	method for adjusting p-values due to multiple testing regime. The available methods are "none", "bonferroni", "holm", "hochberg", "hommel" and "fdr". See also <a href="#">p.adjust</a> .

**Details**

The function `p.spatial2.Rd` is basically the same as `p.spatial`, but differs in its input and output formats. Details about the functionality can be found at [p.spatial](#).

**Value**

A list of a two lists of vectors is produced containing the p-values for positive (Pp) and negative (Pn) deviations of  $\bar{M}$  of the spot's neighbourhood is produced (see example below).

**Note**

This function will be fused with `p.spatial` in future versions using S4-style methods.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[fdr.int](#), [sigxy.plot](#), [p.adjust](#), [p.spatial](#)

## Examples

```
# To run these examples, "un-comment" them!
#
# LOADING DATA
# data(sw)
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this illustration, N was chosen rather small. For "real" analysis, it should be larger.
# P <- p.spatial2(sw,delta=2,N=10000,av="median")
# SIGNIFICANCE PLOTS OF ARRAY 1
# sigxy.plot2(sw[,1],P$Pp[[1]],P$Pn[[1]],color.lim=c(-5,5),main="P-value")
# SIGNIFICANCE PLOTS OF ARRAY 3
# sigxy.plot2(sw[,3],P$Pp[[3]],P$Pn[[3]],color.lim=c(-5,5),main="P-value")
```

---

sig.mask

*Masking of data based on significance testing*

---

## Description

This function sets data to NA if the corresponding spots have significantly biased neighbourhoods on the intensity scale or on the spatial dimensions of the array.

## Usage

```
sig.mask(object, Sp, Sn, thrp, thrn)
```

## Arguments

object	object of class marrayRaw or marrayNorm
Sp	list of vectors of false discovery rate or p-values for positive deviation of $\bar{M}$ as produced by <code>fdr.int2</code> , <code>p.int2</code> , <code>fdr.spatial2</code> or <code>p.spatial2</code> .
Sn	list vector of false discovery rate or p-values for negative deviation of $\bar{M}$ as produced by <code>fdr.int2</code> , <code>p.int2</code> , <code>fdr.spatial2</code> or <code>p.spatial2</code> .
thrp	vector of thresholds for significance of positive deviation (Sp)
thrn	vector of thresholds for significance of negative deviation (Sn)

## Details

This function can be used for the masking of data that has been decided to be unreliable after the application of significance test for intensity- and location dependent dye bias (e.g. `p.int2`, `fdr.int2`, `p.spatial2`, `fdr.spatial2`).

## Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)



**See Also**

[sigint.plot](#), [fdr.int](#), [p.int](#), [sigxy.plot](#), [fdr.spatial](#), [p.spatial](#)

**Examples**

```
# To run these commands, delete comment sign (#) !
#
# LOADING DATA
# data(sw)
#
# MASKING REGIONS WITH SPATIAL DYE BIAS
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this example, N was chosen rather small. For "real" analysis, it should be larger.
# FDR <- fdr.spatial2(sw,delta=2,N=10,av="median",edgeNA=FALSE)
#
# VISUALISATION
# sigxy.plot2(sw[,1],FDR$FDRp[[1]],FDR$FDRn[[1]],color.lim=c(-5,5),main="FDR")
#
# MASKING SIGNIFICANT NEIGHBOURHOODS
# thresp <- c(0.01,0.01,0.01,0.01)
# thresn <- c(0.01,0.01,0.01,0.01)
# sw.masked <- sig.mask(sw,Sp=FDR$FDRp,Sn=FDR$FDRn,thrp=thresp,thrn=thresn)
# mxy.plot(sw.masked[,4]) # plot masked data for array 4
```

---

sigint.plot

*Visualisation of significance of intensity-dependent bias*

---

**Description**

This function visualises the significance of intensity-dependent bias.

**Usage**

```
sigint.plot(A,M,Sp,Sn,ylim=c(-3,-3),...)
```

**Arguments**

A	vector of average logged spot intensity
M	vector of logged fold changes
Sp	vector of false discovery rate or p-values for positive deviation of $\bar{M}$ as produced by <code>fdr.int</code> or <code>p.int</code>

Sn	vector of false discovery rate or p-values for negative deviation of $\bar{M}$ as produced by <code>fdr.int</code> or <code>p.int</code>
ylim	vector of minimal $\log_{10}(\text{fdr})$ or $\log_{10}(\text{p-value})$ to be visualised corresponding to $S_p$ and $S_n$ . FDR or p-values smaller than these values will be set equal to these threshold values for visualisation.
...	Further optional graphical parameter for the <code>plot</code> function generating the MA plot

### Details

The function `sigint.plot` produces a MA-plot of the significance ( $S_p, S_n$ ) generated by `fdr.int` or `p.int`. The abscissa (x-axis) shows by the average logged spot intensity  $A = 0.5 * (\log(Cy3) + \log(Cy5))$ ; the ordinate axis (y-axis) shows the  $\log_{10}(\text{FDR})$  or  $\log_{10}(\text{p})$  given by  $\text{FDR}_p$  or  $\text{P}_n$  and  $\text{FDR}_n$  or  $\text{P}_n$ . The significance for positive  $\bar{M}$  of spot intensity neighbourhoods are presented by red colour; the significance for negative  $\bar{M}$  of spot intensity neighbourhoods are presented by green colour. The ordinate axis (y-axis) give the  $\log_{10}$ -transformed FDR or p-values.

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### See Also

[sigxy.plot](#), [fdr.int](#), [p.int](#)

### Examples

```
# To run these examples, "un-comment" them!
#
# LOADING DATA NOT-NORMALISED
# data(sw)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# This can take a while! For testing, you may choose a smaller N.
# FDR <- fdr.int(maA(sw)[,1],maM(sw)[,1],delta=50,N=100,av="median")
# VISUALISATION OF RESULTS
# sigint.plot(maA(sw)[,1],maM(sw)[,1],FDR$FDRp,FDR$FDRn,c(-5,-5))

# data(sw.olin)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# F <- fdr.int(maA(sw.olin)[,1],maM(sw.olin)[,1],delta=50,N=100,av="median")
# VISUALISATION OF RESULTS
# sigint.plot(maA(sw.olin)[,1],maM(sw.olin)[,1],FDR$FDRp,FDR$FDRn,c(-5,-5))
```

---

`sigint.plot2`*Visualisation of significance of intensity-dependent bias*

---

### Description

This function produce visualises the significance of intensity-dependent bias.

### Usage

```
sigint.plot2(object, Sp, Sn, ylim=c(-3, -3), ...)
```

### Arguments

<code>object</code>	object of class <code>marrayRaw</code> or <code>marrayNorm</code>
<code>Sp</code>	vector of false discovery rate or p-values for positive deviation of $\bar{M}$ as produced by <code>fdr.int2</code> or <code>p.int2</code>
<code>Sn</code>	vector of false discovery rate or p-values for negative deviation of $\bar{M}$ as produced by <code>fdr.int2</code> or <code>p.int2</code>
<code>ylim</code>	vector of minimal $\log_{10}(\text{fdr})$ or $\log_{10}(\text{p-value})$ to be visualised corresponding to <code>Sp</code> and <code>Sn</code> . FDR or p-values smaller than these values will be set equal to these threshold values for visualisation.
<code>...</code>	Further optional graphical parameter for the <code>plot</code> function generating the MA plot

### Details

The function `sigint.plot2` only differs from `sigint.plot` in its input arguments. The functionality is the same. For details, see [sigint.plot](#).

### Note

This function will be merged with `sigint.plot` in future versions.

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### See Also

[sigxy.plot](#), [fdr.int2](#), [p.int2](#)

**Examples**

```
# To run these examples, delete the comment signs (#) in front of the commands.
#
# LOADING DATA NOT-NORMALISED
# data(sw)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this example, N was chosen rather small. For "real" analysis, it should be larger.
# FDR <- fdr.int2(sw,delta=50,N=10,av="median")
# VISUALISATION OF RESULTS
# sigint.plot2(sw[,1],FDR$FDRp[[1]],FDR$FDRn[[1]],c(-5,-5)) # array 1
# sigint.plot2(sw[,4],FDR$FDRp[[4]],FDR$FDRn[[4]],c(-5,-5)) # array 4
```

sigxy.plot

*Visualisation of significance tests for spatial bias***Description**

This function produces a 2D-plot visualizing the significance of spatial bias.

**Usage**

```
sigxy.plot(Sp,Sn,color.lim=c(-3,3),...)
```

**Arguments**

Sp	matrix of false discovery rates or p-values for positive deviation of $\bar{M}$ as produced by <code>fdrspatial</code> or <code>p.spatial</code>
Sn	matrix of false discovery rate or p-values for negative deviation of $\bar{M}$ as produced by <code>fdrspatial</code> or <code>p.spatial</code>
color.lim	limits of color range for plotting vector corresponding to $\log_{10}(pS)$ and $\log_{10}(nS)$
...	Further optional graphical parameter for the image function generating the MXY plot

**Details**

The function `sigxy.plot` produces a 2d-plot presenting the significance ( $pS,nS$ ) generated by `fdrint` or `p.spatial`. The significance  $Sp$  for positive  $\bar{M}$  of spatial spot neighbourhoods are presented by red colour; the significance( $Sn$ ) for negative  $\bar{M}$  of spatial spot neighbourhoods are presented by green colour.

**Author(s)**

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

**See Also**

[colorbar.sig](#), [fdr.spatial](#), [p.spatial](#), [image](#), [p.spatial](#)

**Examples**

```

# To run these examples, "un-comment" them!
#
# LOADING DATA
# data(sw)
#
# M <- v2m(maM(sw)[,1],Ngc=maNgc(sw),Ngr=maNgr(sw),
#          Nsc=maNsc(sw),Nsr=maNsr(sw),main="MXY plot of SW-array 1")
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# This can take a while! For testing, you may choose a smaller N.
# FDR <- fdr.spatial(M,delta=2,N=100,av="median",edgeNA=TRUE)
# sigxy.plot(FDR$FDRp,FDR$FDRn,color.lim=c(-5,5),main="FDR")
#
# LOADING NORMALISED DATA
# data(sw.olin)
# M <- v2m(maM(sw.olin)[,1],Ngc=maNgc(sw.olin),Ngr=maNgr(sw.olin),
#          Nsc=maNsc(sw.olin),Nsr=maNsr(sw.olin),main="MXY plot of SW-array 1")
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# FDR <- fdr.spatial(M,delta=2,N=100,av="median",edgeNA=TRUE)
# VISUALISATION OF RESULTS
# sigxy.plot(FDR$FDRp,FDR$FDRn,color.lim=c(-5,5),main="FDR")
#
#
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# P <- p.spatial(M,delta=2,N=-1,av="median",p.adjust.method="holm")
# VISUALISATION OF RESULTS
# sigxy.plot(P$Pp,P$Pn,color.lim=c(-5,5),main="FDR")

```

sigxy.plot2

*Visualisation of significance tests for spatial bias***Description**

This function produces a 2D-plot visualizing the significance of spatial bias.

**Usage**

```
sigxy.plot2(object,Sp,Sn,color.lim=c(-3,3),...)
```

**Arguments**

object	object of class <code>marrayRaw</code> or <code>marrayNorm</code>
Sp	vector of false discovery rates or p-values for positive deviation of $\bar{M}$ as produced by <code>fdrspatial</code> or <code>p.spatial</code>
Sn	vector of false discovery rate or p-values for negative deviation of $\bar{M}$ as produced by <code>fdrspatial</code> or <code>p.spatial</code>

`color.lim` limits of color range for plotting vector corresponding to  $\log_{10}(pS)$  and  $\log_{10}(nS)$   
`...` Further optional graphical parameter for the image function generating the MXY plot

### Details

The function `sigxy.plot2` differs from `sigxy.plot` in its input arguments. The functionality is the same. For details, see [sigxy.plot](#).

### Note

This function will be merged with `sigxy.plot` in future versions.

### Author(s)

Matthias E. Futschik (<http://itb.biologie.hu-berlin.de/~futschik>)

### See Also

[colorbar.sig](#), [sigxy.plot](#), [sigxy.plot](#), [fdr.spatial2](#), [p.spatial2](#), [image](#)

### Examples

```
# To run these examples, "un-comment" them!
#
# LOADING DATA
# data(sw)
# CALCULATION OF SIGNIFICANCE OF SPOT NEIGHBOURHOODS
# For this illustration, N was chosen rather small. For "real" analysis, it should be larger.
# FDR <- fdr.spatial2(sw,delta=2,N=10,av="median",edgeNA=TRUE)
#
# SIGNIFICANCE PLOTS OF ARRAY 1
# sigxy.plot2(sw[,1],FDR$FDRp[[1]],FDR$FDRn[[1]],color.lim=c(-5,5),main="FDR")
# SIGNIFICANCE PLOTS OF ARRAY 3
# sigxy.plot2(sw[,3],FDR$FDRp[[3]],FDR$FDRn[[3]],color.lim=c(-5,5),main="FDR")
#
```

---

sw

*cDNA microarray data of SW480/SW620 experiment*

---

### Description

Gene expression in two cancer cell lines, SW480 and SW620, is compared. The SW480 cell line was derived from a colon tumour of a 50-year old male patient. The second cell line (SW620) originated from a lymph node metastasis of the same patient. Sharing the same genetic background, these cell lines serve as an model of cancer progression.

Target cDNA from SW480 was labelled with Cy3 whereas cDNA from SW620 was labelled with Cy5 using the amino-allyl labelling method. Both cDNA pools were co-hybridised on glass slides

with 8448 spots. The spots consisted of 3986 distinct sequence-verified human cDNA clones (Research Genetics, release GF211) printed in duplicates, 84 spots from non-human cDNA clones and a further 154 control spots. Spots were printed by 4x4 pins. The experiment consisted of four replicated arrays derived from separate labelling reactions. Local background spot intensities were extracted by QuantArray software (version2.1). Analysis showed that replicated spots were highly correlated (average Pearson correlation: 0.94). Since this may interfere with the efficiency testing performed (and to reduce the size of the data set for illustration purpose), the replicated spots were not included here. Experimental details and further analysis can be found in Futschik et al. (2002).

**Usage**

data(sw)

**Format**

An object of class “marrayRaw”

**Source**

The data was produced and provided by Sharon Pattison of the Cancer Genetics lab and Aaron Jeffs of the Otago Genomics facility of the University of Otago, Dunedin, New Zealand.

**References**

M. Futschik, A.Jeffs, S.Pattison, N.Kasabov, M.Sullivan, A.Merrie and A.Reeve, Gene expression profiling of metastatic and non-metastatic colorectal cancer cell-lines, *Genome Letters*, vol.1, No.1, pp. 26-34, 2002

**See Also**

[sw.olin](#)

---

sw.olin

*Normalised cDNA microarray data of SW480/SW620 experiment*

---

**Description**

The data set sw.olin is derived from data set sw by optimised local intensity-dependent normalisation (OLIN).

**Usage**

data(sw.olin)

**Format**

An object of class “marrayNorm”

**Source**

The original data (sw) was produced and provided by S.Pattison of the Cancer Genetics lab and A. Jeffs of the Otago Genomics facility of the University of Otago, Dunedin, New Zealand.

**References**

M. Futschik, A.Jeffs, S.Pattison, N.Kasabov, M.Sullivan, A.Merrie and A.Reeve, Gene expression profiling of metastatic and non-metastatic colorectal cancer cell-lines, *Genome Letters*, vol.1, No.1, pp. 26-34, 2002

**See Also**

[sw](#), [olin](#)

---

sw . xy

*Spatial coordinates of spot locations of SW480/SW620 experiment*

---

**Description**

The data set sw . xy contains the x- and y-coordinates of the spots in the data set sw

**Usage**

```
data(sw . xy)
```

**Format**

A list of two matrices

**Source**

The original data (sw) was produced and provided by S.Pattison of the Cancer Genetics lab and A.Jeffs of the Otago Genomics facility of the University of Otago, Dunedin, New Zealand.

**References**

M. Futschik, A.Jeffs, S.Pattison, N.Kasabov, M.Sullivan, A.Merrie and A.Reeve, Gene expression profiling of metastatic and non-metastatic colorectal cancer cell-lines, *Genome Letters*, vol.1, No.1, pp. 26-34, 2002

**See Also**

[sw](#)



---

v2m *Converts vector to matrix based on spot layout*

---

### Description

This functions converts a vector to a matrix based on a given spot layout. Optionally, it produces a 2D-plot.

### Usage

```
v2m(V, Ngc, Ngr, Nsc, Nsr, visu=FALSE, color.lim=c(-1, 1), xlab="Columns", ylab="Rows", ...)
```

### Arguments

V	vector of real values
Ngc	number of columns for the grid matrix
Ngr	number of rows for the grid matrix
Nsc	number of columns for the spot matrix
Nsr	number of rows for the spot matrix
visu	If FALSE, MXY plot is generated.
color.lim	Limits of color range for MXY plot
xlab	label of x -axis of MXY plot
ylab	label of y-axis of MXY plot
...	Further optional parameters for the image function generating the MXY plot

### Details

The function v2m converts a vector V (as e.g. derived by `maM(object)[, index]`) to a matrix representing the spatial distribution of the values of V across the array. Note that this function assumes a specific mapping between the data points and the location of spot (i.e. the same mapping rule that is used for `marrayRaw/marrayNorm` objects.) The validity of this mapping should be carefully checked (see also the documentation of packet *marray*.) The option for spatial visualisation is rather restricted to logged fold-changes as the corresponding colour range is centred around zero and follows the conventional colouring (green for negative, red for positive fold-changes). The MXY plot produced by v2n does not include a colour bar. To have a colour included, you can use `mxy.plot`.

### Value

A 2D-matrix with  $(Ngc \times Nsc)$  columns and  $(Ngr \times Nsr)$  is produced. This matrix represents the spatial distribution of the values of vector V given the print-layout.

### Author(s)

Matthias E. Futschik, <http://itb.biologie.hu-berlin.de/~futschik>

**See Also**

[mxy.plot](#), [m2v](#), [marrayRaw](#)

**Examples**

```
# LOADING DATA NOT-NORMALISED
data(sw.olin)
# CONVERSION FROM VECTOR TO MATRIX
M <- v2m(maM(sw.olin)[,1],Ngc=maNgc(sw.olin),Ngr=maNgr(sw.olin),
        Nsc=maNsc(sw.olin),Nsr=maNsr(sw.olin),visu=TRUE)

# BACK-CONVERSION FROM MATRIX TO VECTOR
V <- m2v(M,Ngc=maNgc(sw.olin),Ngr=maNgr(sw.olin),
        Nsc=maNsc(sw.olin),Nsr=maNsr(sw.olin),visu=TRUE)
```

# Index

- \* **datasets**
    - sw, [46](#)
    - sw.olin, [47](#)
    - sw.xy, [48](#)
  - \* **hplot**
    - colorbar.mxy, [10](#)
    - colorbar.mxy.abs, [11](#)
    - colorbar.sig, [12](#)
    - fgbg.visu, [18](#)
    - mxy.abs.plot, [26](#)
    - mxy.plot, [27](#)
    - mxy2.plot, [28](#)
    - sig.mask, [40](#)
    - sigint.plot, [41](#)
    - sigint.plot2, [43](#)
    - sigxy.plot, [44](#)
    - sigxy.plot2, [45](#)
  - \* **htest**
    - fdr.int, [13](#)
    - fdr.int2, [14](#)
    - fdr.spatial, [15](#)
    - fdr.spatial2, [17](#)
    - p.int, [34](#)
    - p.int2, [36](#)
    - p.spatial, [37](#)
    - p.spatial2, [39](#)
  - \* **manip**
    - m2v, [22](#)
    - v2m, [49](#)
  - \* **models**
    - anovaint, [2](#)
    - anovapin, [4](#)
    - anovaplate, [5](#)
    - anovaspatial, [6](#)
  - \* **nonparametric**
    - fdr.int, [13](#)
    - fdr.int2, [14](#)
    - fdr.spatial, [15](#)
    - fdr.spatial2, [17](#)
    - p.int, [34](#)
    - p.int2, [36](#)
    - p.spatial, [37](#)
    - p.spatial2, [39](#)
  - \* **regression**
    - anovaint, [2](#)
    - anovapin, [4](#)
    - anovaplate, [5](#)
    - anovaspatial, [6](#)
    - ino, [19](#)
    - lin, [21](#)
    - ma.matrix, [24](#)
    - ma.vector, [25](#)
    - oin, [30](#)
    - olin, [31](#)
  - \* **univar**
    - fdr.int, [13](#)
    - fdr.int2, [14](#)
    - fdr.spatial, [15](#)
    - fdr.spatial2, [17](#)
    - p.int, [34](#)
    - p.int2, [36](#)
    - p.spatial, [37](#)
    - p.spatial2, [39](#)
  - \* **utilities**
    - backgroundCorrect2, [7](#)
    - bas, [9](#)
    - ino, [19](#)
    - lin, [21](#)
    - ma.matrix, [24](#)
    - ma.vector, [25](#)
    - oin, [30](#)
    - olin, [31](#)
- anova, [3–5](#), [7](#)  
anovaint, [2](#), [7](#)  
anovapin, [4](#)  
anovaplate, [5](#)  
anovaspatial, [3](#), [6](#)

backgroundCorrect, 7, 8  
backgroundCorrect2, 7, 32  
bas, 9

colorbar.mxy, 10, 12, 13  
colorbar.mxy.abs, 11, 27  
colorbar.sig, 11, 12, 12, 44, 46

fdr.int, 13, 14–16, 18, 35, 38, 39, 41, 42  
fdr.int2, 14, 14, 36, 43  
fdr.spatial, 14, 15, 16, 18, 41, 44  
fdr.spatial2, 16, 17, 46  
fgbg.visu, 18, 27–29

gcv, 31, 33

image, 27–29, 44, 46  
ino, 19, 31, 33

kooperberg, 8

lin, 20, 21, 31, 33  
locfit, 22, 31–33  
locfit.raw, 20

m2v, 22, 27–29, 50  
ma.matrix, 24, 25  
ma.vector, 25  
mad, 9, 10  
maNorm, 20, 22, 31, 33  
marrayNorm, 3, 7, 10  
marrayRaw, 3, 7, 19, 28, 50  
mxy.abs.plot, 12, 26  
mxy.plot, 11, 23, 26, 27, 29, 50  
mxy2.plot, 28

oin, 20, 22, 30, 33  
olin, 20, 22, 31, 31, 48

p.adjust, 34–39  
p.int, 14, 34, 36, 41, 42  
p.int2, 15, 35, 36, 43  
p.spatial, 16, 18, 37, 39, 41, 44  
p.spatial2, 39, 46

sig.mask, 40  
sigint.plot, 14, 35, 41, 41, 43  
sigint.plot2, 15, 36, 43  
sigxy.plot, 13, 16, 18, 38, 39, 41–43, 44, 46  
sigxy.plot2, 45

summary.lm, 3–5, 7  
sw, 46, 48  
sw.olin, 47, 47  
sw.xy, 48

v2m, 23, 27–29, 49  
var, 9, 10