Package 'snm'

November 12, 2026

Type Package

Title Supervised Normalization of Microarrays

Version 1.60.0

Author Brig Mecham and John D. Storey <jstorey@princeton.edu>

Description SNM is a modeling strategy especially designed for normalizing high-throughput genomic data. The underlying premise of our approach is that your data is a function of what we refer to as study-specific variables. These variables are either biological variables that represent the target of the statistical analysis, or adjustment variables that represent factors arising from the experimental or biological setting the data is drawn from. The SNM approach aims to simultaneously model all study-specific variables in order to more accurately characterize the biological or clinical variables of interest.

Maintainer John D. Storey <jstorey@princeton.edu>

Depends R (>= 2.12.0)

Imports corpcor, lme4 (>= 1.0), splines

License LGPL

biocViews Microarray, OneChannel, TwoChannel, MultiChannel, DifferentialExpression, ExonArray, GeneExpression, Transcription, MultipleComparison, Preprocessing, QualityControl

git_url https://git.bioconductor.org/packages/snm

git_branch RELEASE_3_23

git_last_commit 37667cd

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

buildBasisFunction 2
sim.doubleChannel 2
sim.preProcessed 4
sim.refDesign 5
sim.singleChannel 6
snm 8
snm.fitted 10
snm.plot 12
snm.summary 13

Index **15**

`buildBasisFunction` *Internal snm functions.*

Description

Internal functions for the snm package not called by the user.

Author(s)

All internal functions written by Brig Mecham <brig.mecham@sagebase.org>, except `edge.fit`, `edge.glr`, and `edge.qvalue` written by John D. Storey <jstorey@princeton.edu>

`sim.doubleChannel` *Simulated data for a double channel microarray experiment.*

Description

Simulates data used in snm function examples.

Usage

```
sim.doubleChannel(seed)
```

Arguments

`seed`	Numeric value used to seed random number generator.

Details

Simulated data set influenced by a probe-specific biological variable, a probe-specific adjustment variables, and intensity-dependent array and dye effects. Data were simulated for a total of 25,000 probes and 20 arrays. The biological variable is a dichotomous variable specifying two groups (Group 1 and Group 2), with 10 channels per group. The continuous probe-specific adjustment variable is sampled from a Normal(1,0.1) and mimics an age effect. The baseline probe intensities were sampled from a chi(1,2) distribution. Any baseline intensities greater than 15 were set to a random value from the interval [15,16]. The random variation terms were sampled from a Normal(0,0.25) and the array and dye functions were defined by randomly sampling coefficients for a two-dimensional B-spline basis functions from a Normal(0,1).

Randomly selected subsets of 30% and 20% of the probes were defined as influenced by the biological groups and age variables, respectfully. The magnitude of the biological effects were sampled from a Normal(1,0.3) distribution, and the probe-specific age effects from a Normal(1,0.1). An instance of this simulated data can be generated using the code in the examples section below.

Value

`raw.data`	a 25,000 by 50 matrix of simulated data generated according to the description above.
`true.nulls`	a vector of indices corresponding to the rows in raw.data of the probes unaffected by the biological variable of interest
`bio.var`	a model matrix of the biological variable of interest.
`adj.var`	a model matrix of the adjustment variables
`int.var`	a data frame of the intensity-dependent adjustment variables

Author(s)

Brig Mecham

See Also

`snm`, `sim.singleChannel`, `sim.preProcessed`, `sim.refDesign`

Examples

```
## Not run:
doubleChannel <- sim.doubleChannel(12346)
snm.obj <- snm(doubleChannel$raw.data, doubleChannel$bio.var,
   doubleChannel$adj.var, doubleChannel$int.var)
ks.test(snm.obj$pval[doubleChannel$true.nulls],"punif")

## End(Not run)
```

`sim.preProcessed` *Simulate data from a microarray experiment without any intensity-dependent effects.*

Description

Simulated data set influenced by a single probe-specific biological and two probe-specific adjustment variables. This parameters for this data are identical to single channel simulated data available as sim.singleChannel(seed) with the difference that this example does not include the intensity-dependent effects. Consult the corresponding help file for details on this simulation.

Usage

```
sim.preProcessed(seed)
```

Arguments

`seed`	Numeric value used to seed random number generator.

Value

`raw.data`	a 25,000 by 50 matrix of simulated data generated according to the description above.
`true.nulls`	a vector of indices corresponding to the rows in raw.data of the probes unaffected by the biological variable of interest
`bio.var`	a model matrix of the biological variable of interest.
`adj.var`	a model matrix of the adjustment variables
`int.var`	set to NULL

Author(s)

Brig Mecham

See Also

`snm`, `sim.doubleChannel`, `sim.singleChannel`, `sim.refDesign`

Examples

```
preProcessed <- sim.preProcessed(12347)
snm.obj <- snm(preProcessed$raw.data,
                      preProcessed$bio.var,
                      preProcessed$adj.var, rm.adj=TRUE)
ks.test(snm.obj$pval[preProcessed$true.nulls],"punif")
```

`sim.refDesign` *Simulates data from a two-color microarray experiment using a reference design.*

Description

Simulates a two channel experiment with a reference design. Used as an example for snm function call.

Usage

```
sim.refDesign(seed)
```

Arguments

`seed`	Numeric value used to seed random number generator.

Details

Simulated data set influenced by a probe-specific biological variable, and intensity-dependent array and dye effects. Data were simulated assuming a uniform reference design for a total of 25,000 probes and 20 arrays, each consisting of two channels. The reference channel consists of a single reference RNA population. The experimental channel measures a dichotomous biological variable specifying two groups (Group 1 and Group 2), with 10 samples per group. The baseline probe intensities were sampled from a chi(1,2) distribution. Any baseline intensities greater than 15 were set to a random value from the interval [15,16]. The random variation terms were sampled from a Normal(0,0.25) and the array and dye functions were defined by randomly sampling coefficients for a two-dimensional B-spline basis functions from a Normal(0,1).

A randomly selected subset of 30% of the probes was defined as influenced by the biological group variable. The magnitude of the biological effects were sampled from a Normal(1,0.3) distribution. An instance of this simulated data can be generated using the code in the examples section below.

Value

`raw.data`	a 25,000 by 50 matrix of simulated data generated according to the description above.
`true.nulls`	a vector of indices corresponding to the rows in raw.data of the probes unaffected by the biological variable of interest
`bio.var`	a model matrix of the biological variable of interest.
`adj.var`	a model matrix of the probe-specific adjustment variables. In this case set to NULL.
`int.var`	a data frame of the intensity-dependent adjustment variables

Author(s)

Brig Mecham and John D. Storey <jstorey@princeton.edu>

See Also

snm, sim.singleChannel, sim.doubleChannel, sim.preProcessed

Examples

```
# reference design on channel level data
# reference channel is included in bio.var
refChannel <- sim.refDesign(12347)
snm.obj <- snm(refChannel$raw.data,refChannel$bio.var, refChannel$adj.var, refChannel$int.var)
ks.test(snm.obj$pval[refChannel$true.nulls],"punif")

# this is a different model formulation
# where reference channel is not included
# in bio.var
bio.var2 = matrix(c(rep(-1,10), rep(1, 10), rep(0,20)), ncol=1, dimnames=list(NULL, "treatment"))
adj.var2 = matrix(c(rep(1,40), rep(1,20), rep(0,20)), ncol=2, byrow=FALSE, dimnames=list(NULL, c("intercept", "ta
# compare bio.var2 to refChannel$bio.var and adj.var2 to refChannel$adj.var
snm.obj <- snm(refChannel$raw.data, bio.var2, adj.var2, refChannel$int.var)
ks.test(snm.obj$pval[refChannel$true.nulls],"punif")

# reference design on log ratio data
# that is, log(target/ref) data
refChannel <- sim.refDesign(12347)
refChannel$raw.data = refChannel$raw.data[,1:20]-refChannel$raw.data[,21:40]
# removing reference channel variables
refChannel$bio.var = refChannel$bio.var[1:20,-3]
refChannel$adj.var = matrix(refChannel$adj.var[1:20,], ncol=1)
refChannel$int.var = data.frame(refChannel$int.var[1:20,1])
snm.obj <- snm(refChannel$raw.data,refChannel$bio.var, refChannel$adj.var, refChannel$int.var)
ks.test(snm.obj$pval[refChannel$true.nulls],"punif")
```

---

sim.singleChannel *Simulate data from a single channel microarray experiment*

Description

Simulates single channel data used as an example for snm function call.

Usage

```
sim.singleChannel(seed)
```

Arguments

seed	Numeric value used to seed random number generator.

Details

Simulated data set influenced by a single probe-specific biological, two probe-specific adjustment variables, and intensity-dependent array effects. Data were simulated for a total of 25,000 probes and 50 arrays. The biological variable is a dichotomous variable specifying two groups (Group 1 and Group 2), with 25 arrays sampled from each group. The dichtomous probe-specific adjustment variables has 5 different levels and mimics a batch effect. The 5 batches each contain 10 samples, and are balanced with respect to the biological grouping factor. The continuous probe-specific adjustment variable is sampled from a Normal(1,0.1) and mimics an age effect. The baseline probe intensities were sampled from a chi(1,2) distribution. Any baseline intensities greater than 15 were set to a random value from the interval [15,16]. The random variation terms were sampled from a Normal(0,0.25) and the array functions were defined by randomly sampling coefficients for a two-dimensional B-spline basis functions from a Normal(0,1).

Randomly selected subsets of 30%, 10%, and 20% of the probes were defined as influenced by the biological groups, batch, and age variables, respectfully. The magnitude of the biological effects were sampled from a Normal(1,0.3) distribution, the probe-specific batch effects from a Normal(0,0.3) and the probe-specific age effects from a Normal(1,0.1). An instance of this simulated data can be generated using the code in the examples section below.

Value

`raw.data`	a 25,000 by 50 matrix of simulated data generated according to the description above.
`true.nulls`	a vector of indices corresponding to the rows in raw.data of the probes unaffected by the biological variable of interest
`bio.var`	a model matrix of the biological variable of interest.
`adj.var`	a model matrix of the adjustment variables
`int.var`	a data frame of the intensity-dependent adjustment variables

Author(s)

Brig Mecham

See Also

`snm`, `sim.doubleChannel`, `sim.preProcessed`, `sim.refDesign`

Examples

```
## Not run:
singleChannel <- sim.singleChannel(12345)
snm.obj <- snm(singleChannel$raw.data,
        singleChannel$bio.var,
        singleChannel$adj.var,
        singleChannel$int.var)
ks.test(snm.obj$pval[singleChannel$true.nulls],"punif")

## End(Not run)
```

snm *Perform a supervised normalization of microarray data*

Description

Implement Supervised Normalization of Microarrays on a gene expression matrix. Requires a set of biological covariates of interest and at least one probe-specific or intensity-dependent adjustment variable.

Usage

```
snm(raw.dat, bio.var=NULL, adj.var=NULL, int.var=NULL,
    weights=NULL, spline.dim = 4, num.iter = 10, lmer.max.iter=1000,
    nbins=20, rm.adj=FALSE, verbose=TRUE, diagnose=TRUE)
```

Arguments

raw.dat	An m probes by n arrays matrix of expression data. If the user wishes to remove intensity-dependent effects, then we request the matrix corresponds to the raw, log transformed data.
bio.var	A model matrix (see model.matrix) or data frame with n rows of the biological variables. If NULL, then all probes are treated as "null" in the algorithm.
adj.var	A model matrix (see model.matrix) or data frame with n rows of the probe-specific adjustment variables. If NULL, a model with an intercept term is used.
int.var	A data frame with n rows of type factor with the unique levels of intensity-dependent effects. Each column parametrizes a unique source of intensity-dependent effect (e.g., array effects for column 1 and dye effects for column 2).
weights	A vector of length m. Values unchanged by algorithm, used to control the influence of each probe on the intensity-dependent array effects.
spline.dim	Dimension of basis spline used for array effects.
num.iter	Number of snm model fit iterations to run.
lmer.max.iter	Number of lmer iterations that are permitted. Set lmer.max.iter=NULL if no maximum is desired.
nbins	Number of bins used by binning strategy. Array effects are calculated from a $nbins$ x n data matrix, where the (i, j) value is equal to that bin i's average intensity on array j.
rm.adj	If set to FALSE, then only the intensity dependent effects have been removed from the normalized data, implying the effects from the adjustment variables are still present. If TRUE, then the adjustment variables effects and the intensity dependent effects are both removed from the returned normalized data.
verbose	A flag telling the software whether or not to display a report after each iteration. TRUE produces the output.
diagnose	A flag telling the software whether or not to produce diagnostic output in the form of consecutive plots. TRUE produces the plot.

Details

This function implements the supervised normalization of microarrays algorithm described in Mecham, Nelson, and Storey (2010).

Value

`norm.dat`	The matrix of normalized data. The default setting is rm.adj=FALSE, which means that only the intensity-dependent effects have been subtracted from the data. If the user wants the adjustment variable effects removed as well, then set rm.adj=TRUE when calling the `snm` function.
`pvalues`	A vector of p-values testing the association of the biological variables with each probe. These p-values are obtained from an ANOVA comparing models where the full model contains both the probe-specific biological and adjustment variables versus a reduced model that just contains the probe-specific adjustment variables. The data used for this comparison has the intensity-dependent variables removed. These returned p-values are calculated after the final iteration of the algorithm.
`pi0`	The estimated proportion of true null probes pi_0, calculated after the final iteration of the algorithm.
`iter.pi0s`	A vector of length equal to num.iter containing the estimated pi_0 values at each iteration of the `snm` algorithm. These values should converge and any non-convergence suggests a problem with the data, the assumed model, or both
`nulls`	A vector indexing the probes utilized in estimating the intensity-dependent effects on the final iteration.
`M`	A matrix containing the estimated probe intensities for each array utilized in estimating the intensity-dependent effects on the final iteration. For memory parsimony, only a subset of values spanning the range is returned, currently nbins*100 values.
`array.fx`	A matrix of the final estimated intensity-dependent array effects. For memory parsimony, only a subset of values spanning the range is returned, currently nbins*100 values.
`bio.var`	The processed version of the same input variable.
`adj.var`	The processed version of the same input variable.
`int.var`	The processed version of the same input variable.
`df0`	Degrees of freedom of the adjustment variables.
`df1`	Degrees of freedom of the full model matrix, which includes the biological variables and the adjustment variables.
`raw.dat`	The input data.
`rm.var`	Same as the input (useful for later analyses).
`call`	Function call.

Note

It is necessary for adj.var and adj.var+bio.var to be valid model matrices (e.g., the models cannot be over-determined).

We suggest that the probe level data be analyzed on the log-transformed scale, particularly if the user wishes to remove intensity-dependent effects. It is recommended that the normalized data (and resulting inference) be inspected for latent structure using Surrogate Variable Analysis (Leek and Storey 2007, PLoS Genetics).

Author(s)

Brig Mecham and John D. Storey <jstorey@princeton.edu>

References

Mecham BH, Nelson PS, Storey JD (2010) Supervised normalization of microarrays. Bioinformatics, 26: 1308-1315.

See Also

model.matrix, plot.snm, fitted.snm, summary.snm, sim.singleChannel, sim.doubleChannel, sim.preProcessed, sim.refDesign

Examples

```
singleChannel <- sim.singleChannel(12345)
snm.obj <- snm(singleChannel$raw.data,
        singleChannel$bio.var,
        singleChannel$adj.var,
        singleChannel$int.var)
ks.test(snm.obj$pval[singleChannel$true.nulls],"punif")
plot(snm.obj)
summary(snm.obj)
snm.fit = fitted(snm.obj)
```

---

snm.fitted *Extract fitted values from an snm object*

Description

Computes fitted values under models used in snm normalization.

Usage

```
snm.fitted(object, ...)
## S3 method for class 'snm'
fitted(object, ...)
```

Arguments

object	Output from the snm function.
...	Not used.

Details

Returns the fitted values under the "null model" (adjustment variables only) and the "full model" (adjustment variables + biological variables).

Value

fit0	Linear model fits when regressing each probe's normalized data on the null model, ~adj.var.
fit1	Linear model fits when regressing each probe's normalized data on the full model, ~adj.var+bio.var.

Note

These fits are useful for investigating the quality of the study-specific model used in the normalization. For example, the residuals can be obtained from the full model fit and examined for latent structure.

Author(s)

John D. Storey <jstorey@princeton.edu>

References

Mecham BH, Nelson PS, Storey JD (2010) Supervised normalization of microarrays. Bioinformatics, 26: 1308-1315.

See Also

snm, sim.singleChannel

Examples

```
## Not run:
singleChannel <- sim.singleChannel(12345)
snm.obj <- snm(singleChannel$raw.data,
        singleChannel$bio.var,
        singleChannel$adj.var[,-6],
        singleChannel$int.var, num.iter=10)
snm.fit = fitted(snm.obj)
res1 = snm.obj$norm.dat - snm.fit$fit1
snm.svd = fast.svd(res1)
cor(snm.svd$v[,1], singleChannel$adj.var[,6])
plot(snm.svd$v[,1], singleChannel$adj.var[,6])

## End(Not run)
```

snm.plot *Display plots for an snm object*

Description

Creates a diagnostic plot of the snm fit.

Usage

```
snm.plot(x, col.by=NULL, ...)
## S3 method for class 'snm'
plot(x, ...)
```

Arguments

x	Output from the snm function.
col.by	A factor vector of length equal to the number of arrays providing a grouping by which to color the intensity-dependent effects. Instead of a factor vector, the input may also be a model matrix composed only of 0's and 1's with the number of rows equal to the number of arrays and number of columns less than or equal to the number of arrays.
...	Arguments passed to the plot functions. Not recommended.

Details

A four panel plot composed of the following:

1. Convergence of π_0 estimates over model fitting iterations. The pi_0 estimates for each iteration are compared to the pi_0 estimate calculated during the final model fit.
2. A scree plot of the principal components analysis of the full model residual matrix.
3. A plot of the estimated intensity-dependent effects.
4. A histogram of the p-values testing each probe for an asssociation with the biological variables (bio.var). All probes to the right of the vertical red line are the least $\hat{\pi}_0$ significant probes (i.e., those used in estimating intensity-dependent effects). The dashed horizontal line is the pi_0 estimate from the final model fit.

Value

Nothing of interest.

Author(s)

John D. Storey <jstorey@princeton.edu>

References

Mecham BH, Nelson PS, Storey JD (2010) Supervised normalization of microarrays. Bioinformatics, 26: 1308-1315.

See Also

snm, sim.singleChannel

Examples

```
## Not run:
singleChannel <- sim.singleChannel(12345)
snm.obj <- snm(singleChannel$raw.data,
        singleChannel$bio.var,
        singleChannel$adj.var,
        singleChannel$int.var, num.iter=10)
plot(snm.obj, col.by=snm.obj$bio.var) #color by biological group
plot(snm.obj, col.by=snm.obj$adj.var[,-6]) #color by batch

## End(Not run)
```

---

snm.summary *Display summary information for an snm object*

Description

Provides a summary of the snm fit.

Usage

```
snm.summary(object, cuts=c(0.0001, 0.001, 0.01, 0.025, 0.05, 0.10, 1), ...)
## S3 method for class 'snm'
summary(object, ...)
```

Arguments

object	An object of class "snm", i.e., output from the snm function.
cuts	Cut points at which to calculate the number of significant probes.
...	Optional arguments to send to the qvalue function in calculating statistical significance.

Details

A summary of the snm fit.

Value

Nothing of interest.

Author(s)

John D. Storey <jstorey@princeton.edu>

References

Mecham BH, Nelson PS, Storey JD (2010) Supervised normalization of microarrays. Bioinformatics, 26: 1308-1315.

See Also

snm, sim.singleChannel

Examples

```
## Not run:
singleChannel <- sim.singleChannel(12345)
snm.obj <- snm(singleChannel$raw.data,
        singleChannel$bio.var,
        singleChannel$adj.var,
        singleChannel$int.var, num.iter=10)
summary(snm.obj)

## End(Not run)
```

Index

∗ **datagen**
  sim.doubleChannel, 2
  sim.preProcessed, 4
  sim.refDesign, 5
  sim.singleChannel, 6
∗ **htest**
  snm, 8
∗ **internal**
  buildBasisFunction, 2
∗ **misc**
  snm.fitted, 10
  snm.plot, 12
  snm.summary, 13
∗ **models**
  snm, 8

buildBasisFunction, 2
buildBasisSplineMatrix *(buildBasisFunction)*, 2

calcArrayEffects *(buildBasisFunction)*, 2
calculate.nulls *(buildBasisFunction)*, 2
checkArguments *(buildBasisFunction)*, 2

edge.fit *(buildBasisFunction)*, 2
edge.glr *(buildBasisFunction)*, 2
edge.qvalue *(buildBasisFunction)*, 2
err.msg *(buildBasisFunction)*, 2
estimateDifferentialExpression *(buildBasisFunction)*, 2

fit.fast.model *(buildBasisFunction)*, 2
fit.model *(buildBasisFunction)*, 2
fitted.snm, *10*
fitted.snm *(snm.fitted)*, 10

getSpanningSet *(buildBasisFunction)*, 2

lmer, *8*

make.ref.model.matrices *(buildBasisFunction)*, 2
make.snm.obj *(buildBasisFunction)*, 2
makeDataObject *(buildBasisFunction)*, 2
model.matrix, *8*, *10*

plot.snm, *10*
plot.snm *(snm.plot)*, 12

qvalue, *13*

remove.adjustment.vars *(buildBasisFunction)*, 2
rm.zero.cols *(buildBasisFunction)*, 2

sim.doubleChannel, 2, *4*, *6, 7*, *10*
sim.function.var *(buildBasisFunction)*, 2
sim.intensity.dep *(buildBasisFunction)*, 2
sim.preProcessed, *3*, 4, *6, 7*, *10*
sim.probe.specific *(buildBasisFunction)*, 2
sim.refDesign, *3, 4*, 5, *7*, *10*
sim.singleChannel, *3, 4*, *6*, 6, *10, 11*, *13, 14*
snm, *3, 4*, *6, 7*, 8, *11*, *13, 14*
snm.diagnostic.plot *(buildBasisFunction)*, 2
snm.fitted, 10
snm.plot, 12
snm.summary, 13
summary.snm, *10*
summary.snm *(snm.summary)*, 13