XDE

November 11, 2009

R topics documented:

burnin *Indicator for running a MCMC burnin*

Description

When TRUE, log files from MCMC chains are not written to file. When FALSE, log files are written for every parameter by default.

Usage

```
burnin(object)
```

Arguments

object	An object of class XdeParameter

Value

logical

Author(s)

R. Scharpf

See Also

XdeParameter-class

Examples

```
data(expressionSetList)
params <- new("XdeParameter", phenotypeLabel="adenoVsquamous",
              esetList=expressionSetList)

##the replacement method for burnin is called for its side effect of
##providing default values of storing MCMC chains
output(params)[2:22]
burnin(params) <- FALSE
output(params)[2:22]
burnin(params) <- TRUE
output(params)[2:22]
```

`calculatePosteriorAvg` *Calculate the posterior average for indicators of concordant and discordant differential expression*

Description

This function calculates the posterior average for indicators of concordant and discordant differential expression from the saved log files. See details.

Usage

```
calculatePosteriorAvg(object, NCONC=2, NDIFF=1)
```

Arguments

`object`	Object of class `XdeMcmc`
`NCONC`	Integer: number of studies for which the gene must be differentially expressed (in the same direction) to be classified as concordant differential expression
`NDIFF`	Integer: number of studies for which a gene must be up- or down-regulated to be classified as differentially expressed. It is the union of concordant and discordant differential expression.

Details

For each iteration,

1. calculate the sign of delta * Delta

2. For each gene, compute the number of positive signs (P) and the number of negative signs (N) (a G x 2 matrix, where G is the number of genes in common across all studies). P + N is <= S, where S is the number of studies.

3. for a given gene, the discordant indicator is simply when P * N is nonzero.

4. The concordant indicator requires P * N = 0 AND P + N >= NCONC, where NCONC is specified by the user.

5. differential expression is simply | P | + | N | >= NDIFF. By default, NDIFF is 1 but can be user-specified.

The posterior average is then computed from the mean over all MCMC iterations.

Value

A G x 3 matrix.

Author(s)

RS

See Also

`posteriorAvg`

`empiricalStart` *Empirical starting values for the MCMC*

Description

Empirical starting values for the MCMC are based on data in objects of class `ExpressionSetList`

Usage

```
empiricalStart(object, zeroNu = FALSE, phenotypeLabel, one.delta=FALSE, T_THRESH
```

Arguments

`object`	An object of class `ExpressionSetList`
`zeroNu`	Logical: if TRUE, the nu in the Bayesian model are not modeled – set to zero and not updated in the MCMC. Setting zeroNu to TRUE should be regarded as experimental
`phenotypeLabel`	character: binary phenotype. phenotypeLabel must be in the varLabels of each ExpressionSet object
`one.delta`	delta in the Bayesian model is a gene-specific indicator for differential expression. If one.delta is FALSE, we assume that a gene can be differentially expressed in a subset of studies. When TRUE, we assume that a gene is differentially expressed in all studies or in none.
`T_THRESH`	A threshold of t-statistics (calculated row-wise for each study) for determining starting values of the differential expression indicator, delta.

Value

A list containing starting values for the MCMC that are derived from empirical estimates of the data.

Author(s)

R. Scharpf

See Also

`zeroNu`, `XdeParameter-class`, `ExpressionSetList-class`

Examples

```
data(expressionSetList)
eList <- studyCenter(expressionSetList)
empirical <- empiricalStart(eList, phenotypeLabel="adenoVsquamous", T_THRESH=3)
##By default, initial values for the MCMC are sampled from the prior
##when initializing an object of class XdeParamater
params <- new("XdeParameter", esetList=eList,
              phenotypeLabel="adenoVsquamous", one.delta=FALSE, burnin=TRUE)
##The initial values can be replaced by empirical values as follows:
firstMcmc(params) <- empirical
```

ExpressionSetList-class
A class for containing a list of ExpressionSets

Description

Each element in the list must be a valid ExpressionSet. The featureNames must be identical for each ExpressionSet.

Objects from the Class

Objects can be created by calls of the form new("ExpressionSetList", ...).

Slots

.Data: Object of class "list"

Extends

Class "list", from data part. Class "vector", by class "list", distance 2. Class class.AssayData, by class "list", distance 2.

Methods

.integrativeCorrelationFilter signature(object = "ExpressionSetList") Experimental function for filtering an arbitrary list of ExpressionSets by integrative correlation. Genes are excluded that do not exceed the fdr threshold in at least 1 of the studies.

"[" signature(x = "ExpressionSetList") Subsets each ExpressionSet element in the list.

coerce signature(from = "list", to = "ExpressionSetList") Coerces a list of ExpressionSet objects to an object of class ExpressionSetList. The validityMethod for the ExpressionSetList class will return an error if the featureNames for each ExpressionSet are not identical.

dim signature(x="ExpressionSetList") applies dim to each element of the list.

featureNames signature(object = "ExpressionSetList") Accessor for the featureNames

geneCenter signature(object = "ExpressionSetList") See geneCenter

lapply signature(object="ExpresssionSetList") Coerces instance of ExpressionSetList to a list and does lapply on the list. Returns an object of class ExpressionSetList

nSamples signature(x = "ExpressionSetList") Numerical vector giving the number of samples in each ExpressionSet

nrow signature(x = "ExpressionSetList") Numerical: number of features or genes

pData signature(object = "ExpressionSetList") returns a list of data.frames. The elements of the list correspond to the studies in the ExpressionSetList object.

.pca signature(object = "ExpressionSetList") Runs principal components to generate cross-study summary scores of differential expression in multiple studies. Not meant to be called directly. See xsScores.

standardizeSamples signature(object = "ExpressionSetList") See standardizeSamples

studyCenter signature(object = "ExpressionSetList") See studyCenter

zeroNu signature(object = "ExpressionSetList") See zeroNu.

Author(s)

R. Scharpf

See Also

XdeMcmc-class, XdeParameter-class

Examples

```
showClass("ExpressionSetList")
data(expressionSetList)
```

---

expressionSetList *Example of ExpressionSetList*

Description

Object of class ExpressionSetList containing three studies. Each element in the list is an ExpressionSet

Usage

```
data(expressionSetList)
```

Details

Parmigiani et al. (2004) performed a cross-study analysis of three lung cancer studies. The studies used in this analysis were merged by UniGene identifiers to obtain a set of 3,171 gene. The R experiment data package lungExpression that was developed to facilitate the reproducibility of this analysis contains the three studies as ExpressionSets. Here, we take a random sample of 500 features from one study (the "stanford" study), and split this study into three artificial studies that each contain 4 squamous carcinomas and 3 adenocarcinomas. The three artificial studies are then used to create an instance of the ExpressionSetList class.

See Garber et al. (2001) for the raw data and description of the stanford study.

Source

The experiment data package lungExpression (www.bioconductor.org)

References

Parmigiani et al. (2004) A cross-study comparison of gene expression studies for the molecular classification of lung cancer, Clin Cancer Res, 10(9): 2922-2927

Garber et al. (2001) Diversity of gene expression in adenocarcinoma of the lung, PNAS, 98:13784-13789

Examples

```
data(expressionSetList)
```

firstMcmc *Values for the first MCMC iteration*

Description

Accessor method for the values of the first MCMC iteration

Usage

```
firstMcmc(object)
```

Arguments

object	An object of class XdeParameter

Value

Returns a list of the values to be used in the first iteration of the MCMC.

Author(s)

R. Scharpf

See Also

XdeParameter-class, lastMcmc

Examples

```
data(expressionSetList)
params <- new("XdeParameter", phenotypeLabel="adenoVsquamous",
              esetList=expressionSetList)
str(firstMcmc(params))
```

geneCenter *Center the expression values for each gene in a study to zero*

Description

Mean centers the genes for each study in a list

Usage

```
geneCenter(object)
```

Arguments

object	Object of class ExpressionSetList

Value

Object of class `ExpressionSetList`

Author(s)

R. Scharpf

See Also

`studyCenter`, `ExpressionSetList-class`

Examples

```
data(expressionSetList)
centered <- geneCenter(expressionSetList)
```

---

hyperparameters *Accessor for hyperparameters of the Bayesian model*

Description

Accessor and replacement methods for hyperparameters of the Bayesian model are provided

Usage

```
hyperparameters(object)
```

Arguments

`object`	An object of class `XdeParameter`

Details

See the XdeParameterClass vignette for a more detailed discussion. The default values provided when initializing an object of class `XdeParameter` works well in most instances.

Value

A numerical vector

Author(s)

R. Scharpf

References

R. Scharpf et al., A Bayesian Model for Cross-Study Differential Gene Expression, Technical Report 158, Johns Hopkins University, Department of Biostatistics, 2007

Examples

```
data(expressionSetList)
xlist <- new("XdeParameter", esetList=expressionSetList, phenotypeLabel="adenoVsquamous")
hyperparameters(xlist)
```

---

iterations *Number of MCMC iterations*

Description

Number of MCMC iterations

Usage

```
iterations(object)
```

Arguments

object	An object of class XdeParameter or XdeMcmc.

Details

For an object of class XdeParameter, iterations specifies the total number of MCMC iterations. Note that by setting the thin parameter to a value greater than 1, the number of MCMC iterations will be greater than the number of saved MCMC iterations (saved iterations = iterations / thin).

For an object of class XdeMcmc (a class that stores output from the MCMC), iterations specifies the number of iterations that were saved.

The replacement method is only defined for the XdeParameter class. The class XdeMcmc is meant to reflect the information in an already run chain, whereas XdeParameter is a class for parameterizing the Bayesian model that has not yet been fit.

Value

An integer

Author(s)

R. Scharpf

See Also

XdeParameter-class, XdeMcmc-class

lastMcmc *MCMC values for the last iteration*

Description

MCMC values for the last iteration. Useful if more iterations are needed.

Usage

```
lastMcmc(object)
```

Arguments

object	Object of class XdeMcmc

Value

An environment.

Author(s)

R. Scharpf

See Also

firstMcmc

Examples

```
## Not run:
  data(expressionSetList)
  xparam <- new("XdeParameter", phenotypeLabel="adenoVsquamous",
                esetList=expressionSetList)
  iterations(xparam) <- 10
  fit <- xde(xparam, esetList=expressionSetList)
  ##Do more iterations and use a different seed
  firstMcmc(xparam) <- lastMcmc(fit)
  seed(xparam) <- 97814
  fit2 <- xde(xparam, esetList=expressionSetList)

  ##Or
  fit2 <- xde(xparam, esetList=expressionSetList, outputMcmc=fit)
## End(Not run)
```

output *Options for storing results of the MCMC chains*

Description

A numeric vector indicating which chains to write to file and, for those parameters that are written to file, how often the chains should be written to file.

Usage

```
output(object)
```

Arguments

object	An object of class XdeParameter or XdeMcmc

Details

Replacement methods are only available for objects of class XdeParameter. Accessor methods are available for objects of class XdeParameter and XdeMcmc.

Value

A named numerical vector. The first element (thin) specifies how often to write chains to file. For instance, if output[1]=2 the chains will be written to file every other iteration. Elements 2 - 22 of the vector are indicators for whether to write the write the chains of the Bayesian parameters to file.

Note

Parameters indexed by gene and study (Delta, Phi, Nu, and sigma2) grow very large quickly.

Author(s)

R. Scharpf

See Also

burnin, XdeParameter-class, XdeMcmc-class

Examples

```
data(xmcmc)
output(xmcmc)
```

pairs-methods *pairs function for high-throughput data*

Description

A convenient wrapper for pairs that uses smoothScatter to plot the density of the points and displays the spearman correlation coefficient of the pairwise scatterplots.

Methods

x = "matrix" Typically a matrix of effect size estimates obtained in each study. Rows are genes, columns are studies.

x = "data.frame" Typically a `data.frame` of effect size estimates obtained in each study. Rows are genes, columns are studies.

posteriorAvg *Accessor and replacement methods for posterior averages of differential expression*

Description

Accessor and replacement methods for objects of class `XdeMcmc` for posterior averages of differential expression

Usage

```
posteriorAvg(object)
posteriorAvg(object) <- value
```

Arguments

`object`	Object of class `XdeMcmc`
`value`	A matrix of dimension G x 3, where G is the number of genes and 3 are different ways of quantifying differential expression in the context of multiple studies (concordant, discordant, or the union).

Value

A matrix of dimension G x 3, where G is the number of genes and 3 are different ways of quantifying differential expression in the context of multiple studies (concordant, discordant, or the union).

Author(s)

RS

See Also

`calculatePosteriorAvg`

seed *Seed for the MCMC*

Description

Setting a seed is useful for reproducing MCMC chains

Usage

```
seed(object)
seed(object) <- value
```

Arguments

object	An object of XdeParameter or XdeMcmc
value	Numeric or integer

Details

The seed stored in the slot of an object of class XdeParameter and an object of class XdeMcmc are useful in different ways. For the XdeParameter class, the seed indicates what seed was used to initialize an MCMC chain. By contrast, an object of class XdeMcmc contains a seed that would be useful for running additional iterations – the seed here is guaranteed to be different from the seed that was used to initiate the MCMC.

Value

An integer

Author(s)

R. Scharpf

ssStatistic *Calculate single study estimates of effect size*

Description

Calculate single study estimates of effect size for lists of ExpressionSets

Usage

```
ssStatistic(statistic = c("t", "sam", "z")[1], phenotypeLabel, esetList, ...)
```

Arguments

`statistic`	Character string indicating Welch t-statistic (t), SAM (sam), or a z-statistic (z)
`phenotypeLabel`	Character string indicating the name of the binary covariate
`esetList`	An object of class `ExpressionSetList`
`...`	Not implemented. Potentially additional arguments to the above methods that are implemented in other packages

Details

This function is a wrapper that provides an estimate of effect size for each study (element) in an `ExpressionSetList` object.

For Welch t-statistic, this function is a wrapper for mt.teststat in the multtest package.

For SAM, this function is a wrapper for the sam function in the siggenes package.

The "z" statistic is a standardized unbiased estimate of effect size (Hedges and Olkin, 1985) – implementation is in the zScores function in the R package GeneMeta.

See the complete references below.

Value

A matrix: rows are genes and columns are studies

Author(s)

R. Scharpf

References

J.K. Choi, U. Yu, S. Kim, and O.J. Yoo (2003), Combining multiple microarray studies and modeling interstudy variation, Bioinformatics, 19(1) I84-I90.

Y. Ge, S. Dudoit & T. P. Speed (2003), Resampling-based multiple testing for microarray data hypothesis Test 12(1) : 1-44 (with discussions on 44-77).

L. Lusa R. Gentleman, and M. Ruschhaupt, GeneMeta: MetaAnalysis for High Throughput Experiments

L.V. Hedges and I. Olkin, Statistical Methods for Meta-analysis (1985), Academic Press

Tusher, Tibshirani and Chu (2001), Significance analysis of microarrays applied to the ionizing radiation response, PNAS 2001 98: 5116-5121, (Apr 24).

Examples

```
data(expressionSetList)
if(require(siggenes)){
  sam <- ssStatistic("sam", esetList=expressionSetList, phenotypeLabel="adenoVsquamous")
}
```

standardizeSamples *Centers the genes at zero and standardizes the samples to have variance 1*

Description

For each study (element) in an `ExpressionSetList` object, this function centers the genes to have mean zero (rows) and scales the variance of the samples to 1.

Usage

```
standardizeSamples(object, ...)
```

Arguments

`object`	Object of class `ExpressionSetList`
`...`	Additional arguments not implemented

Value

An object of class `ExpressionSetList`

Note

Requires genefilter package

Author(s)

R. Scharpf

studyCenter *Center the expression values in a study to zero*

Description

Centers each study in a list so that the average expression value of each stuy is zero

Usage

```
studyCenter(object)
```

Arguments

`object`	An object of class `ExpressionSetList`

Value

An object of class `ExpressionSetList`

Author(s)

R. Scharpf

See Also

geneCenter, ExpressionSetList-class

Examples

```
data(expressionSetList)
centered <- studyCenter(expressionSetList)
lapply(centered, function(object) round(mean(exprs(object)), 4))
```

---

symbolsInteresting *Useful for changing the look of pairs plots to emphasize concordant or discordant genes*

Description

This function can be used to order genes in a matrix by the rank of a statistic and provide different plotting symbols and colors for genes that exceed a certain threshold of the ranking statistic.

Usage

```
symbolsInteresting(rankingStatistic, percentile = 0.9, colors = c("grey50", "roy
```

Arguments

rankingStatistic	Any numerical vector
percentile	A percentile of the rankingStatistic – above which a gene would be classified as 'interesting'
colors	character string of length 2: a color for genes not exceeding the percentile and a color for genes exceeding the thresold
symbols	two plotting symbols (numeric or character): symbol for genes not exceeding percentile and symbol for genes exceeding percentile
size	numeric vector of length 2: size of plotting symbol for genes not exceeding percentile and size of plotting symbol for genes exceeding percentile
background	character vector of length 2: background color of plotting symbols for gene not exceeding percentile and for genes exceeding the percentile

Value

order	the order of the rankingStatistic
pch	plotting symbols (same length as rankingStatistic)
col	color of plotting symbols (same length as rankingStatistic)
bg	background color of plotting symbols (same length as rankingStatistic)
cex	size of plotting symbols (same length as rankingStatistic)

Author(s)

R. Scharpf

Examples

```
data(expressionSetList)
data(xmcmc)
pathToLogFiles <- system.file("logFiles", package="XDE")
load(file.path(pathToLogFiles, "BES.rda"))
load(file.path(pathToLogFiles, "postAvg.rda"))
op.conc <- symbolsInteresting(rankingStatistic=postAvg[, "concordant"])
graphics:::pairs(BES[op.conc$order, ], pch=op.conc$pch, col=op.conc$col,
                 bg=op.conc$bg, upper.panel=NULL, cex=op.conc$cex)
```

---

thin *How often to write MCMC iterations to file*

Description

A value greater than one means that not every MCMC iteration is written to file.

Usage

```
thin(x, ...)
```

Arguments

x	An object of class XdeParameter
...	not implemented

Details

thin is an accessor for the first element in the vector returned by the method output.

The replacement method replaces the first element in the output vector.

Value

An integer.

Author(s)

R. Scharpf

See Also

output

tuning *Tuning parameters for Metropolis-Hastings proposals*

Description

Accessor and replacement methods for tuning the Metropolis-Hastings proposal parameters.

Usage

```
tuning(object)
```

Arguments

object	Object of class XdeParameter

Details

See the XdeParameterClass vignette

Value

A numerical vector

Author(s)

R. Scharpf

updates *Frequency of updating a parameter per MCMC iteration*

Description

Accessor and replacement methods for the class XdeParameter are available. Specifying an update of integer N for a Metropolis-Hastings parameter means that N values are proposed for that parameter for each MCMC iteration.

Usage

```
updates(object)
```

Arguments

object	An object of class XdeParameter

Details

See the XdeParameterClass vignette

Value

A numerical vector

Author(s)

R. Scharpf

`XdeMcmc-class` *Class for storing output from the Bayesian model*

Description

Stores output, including the last iteration of the MCMC.

Objects from the Class

Objects can be created by calls of the form `new("XdeMcmc", studyNames, featureNames, iterations, seed, output, directory, lastMcmc, posteriorAvg, bayesianEffectSize)`

Slots

studyNames: Object of class `"character"`
featureNames: Object of class `"character"`
iterations: Object of class `"numeric"`
directory: Object of class `"character"`
seed: Object of class `"integer"`
output: Object of class `"numeric"`
lastMcmc: Object of class `"environment"`
posteriorAvg: Object of class `"NULLorMatrix"`
bayesianEffectSize: Object of class `"NULLorMatrix"`

Methods

$ `signature(x = "XdeMcmc")`
.standardizedDelta `signature(object = "XdeMcmc")`
bayesianEffectSize `signature(object = "XdeMcmc")`
bayesianEffectSize<- `signature(object = "XdeMcmc", value = "matrix")`
calculatePosteriorAvg `signature(object = "XdeMcmc")`: See `calculatePosteriorAvg`
directory `signature(object = "XdeMcmc")`
featureNames `signature(object = "XdeMcmc")`
initialize `signature(.Object = "XdeMcmc")`
iterations `signature(object = "XdeMcmc")`
lastMcmc `signature(object = "XdeMcmc")`
nrow `signature(x = "XdeMcmc")`
output `signature(object = "XdeMcmc")`
plot `signature(x = "XdeMcmc")`
posteriorAvg `signature(object = "XdeMcmc")`
seed `signature(object = "XdeMcmc")`
show `signature(object = "XdeMcmc")`
studyNames `signature(object = "XdeMcmc")`

Author(s)

R. Scharpf

See Also

The class for storing the data: `ExpressionSetList-class` and the class that contains default options for fitting the Bayesian model: `XdeParameter-class`

Examples

```
##See XDE vignette:
## Not run:
openVignette(package="XDE")
## End(Not run)
```

XdeParameter-class *Container class for storing options of the Bayesian hierarchical model*

Description

This class contains initial values for the first iteration of the MCMC, options for saving MCMC chains, options for changing the tuning parameters of the Metropolis-Hastings algorithm, options for changing hyperparameters from their defaults, etc.

Objects from the Class

Objects can be created by calls of the form `new("XdeParameter", esetList, updates, tuning, hyperparameters, output, iterations, burnin, seed, randomSeed, genes, studies, firstMcmc, specifiedInitialValues, directory, phenotypeLabel, showIterations, verbose, studyNames, one.delta)`.

Slots

`updates`: Object of class `numeric`. The frequency of updates for each iteration of the chain.

`tuning`: Object of class `numeric`. Tuning parameters for the Metropolis-Hastings proposals

`hyperparameters`: Object of class `numeric`. Hyperparameters for the Bayesian hierarchical model

`output`: Object of class `numeric`. Indicator for whether to save the MCMC chain to file. If the value is zero, the chain is not saved.

`iterations`: Object of class `numeric`. The total number of MCMC iterations.

`burnin`: Object of class `logical`. If set to FALSE, by default none of the chains will be saved (called for its side-effect of setting the output to zero for each parameter).

`notes`: Object of class `character`.

`firstMcmc`: Object of class `environment`. Values for the first iteration of the MCMC

`showIterations`: Object of class `logical`. Whether to show the MCMC iteration when fitting the model

`specifiedInitialValues`: Object of class `logical`. If TRUE (the default), the values stored in `firstMcmc` will be used for the first iteration of the MCMC.

directory: Object of class `character`. Specifies where to write the log files

phenotypeLabel: Object of class `character`. The name of the binary covariate used for differential expression

verbose: Object of class `logical`

studyNames: Object of class `character`. Names of the datasets

Methods

burnin `signature(object = "XdeParameter")` logical. See `burnin`

burnin<- `signature(object = "XdeParameter", value = "logical")` logical. See `burnin`

directory `signature(object = "XdeParameter")` character string giving the path or relative path to store log files from the MCMC chain

directory<- `signature(object = "XdeParameter")` Path to store log files.

firstMcmc `signature(object = "XdeParameter")` See `firstMcmc`

firstMcmc<- `signature(object = "XdeParameter", value = "environment")`

firstMcmc<- `signature(object = "XdeParameter", value = "list")`

hyperparameters `signature(object = "XdeParameter")` See the XdeParameterClass vignette

hyperparameters<- `signature(object = "XdeParameter")` See the XdeParameterClass vignette

initialize `signature(.Object = "XdeParameter")` Method for initializing an instance of the class. The default values provided work well in most cases.

iterations `signature(object = "XdeParameter")` Accessor for the total number of MCMC iterations to run

iterations<- `signature(object = "XdeParameter", value = numeric)` The replacement method is useful for setting a different number of iterations.

iterations<- `signature(object = "XdeParameter", value = "integer")`

output `signature(object = "XdeParameter")` See also `output`. This method is also defined for class `XdeMcmc`

output<- `signature(object = "XdeParameter")` See also `output`

phenotypeLabel `signature(object = "XdeParameter")` The name of a binary covariate present in each study

phenotypeLabel<- `signature(object = "XdeParameter", value = "character")`

savedIterations `signature(object = "XdeParameter")` The number of MCMC iterations written to file. It is the value of the total number of iterations divided by the thinning parameter. See also `output`

seed `signature(object = "XdeParameter")` See `seed`

seed<- `signature(object = "XdeParameter", value="integer")` Replacement method. See also `seed`.

show `signature(object = "XdeParameter")` Produces a short summary of objects that are instances of the `XdeParameter` class

showIterations `signature(object = "XdeParameter")` logical

showIterations<- `signature(object = "XdeParameter")`

studyNames `signature(object = "XdeParameter")` Names of the high-throughput gene expression studies

studyNames<- `signature(object = "XdeParameter")`

thin `signature(x = "XdeParameter")` See `output` and `thin`

thin<- `signature(x = "XdeParameter", value = numeric)` See `thin`

tuning `signature(object = "XdeParameter")` See also `tuning`

tuning<- `signature(object = "XdeParameter")`

updates `signature(object = "XdeParameter")` See also `updates`

updates<- `signature(object = "XdeParameter")`

Author(s)

R. Scharpf

References

R. Scharpf

See Also

`ExpressionSetList-class`

Examples

```
showClass("XdeParameter")
##See the XdeParameterClass vignette
```

`xde` *Fit the Bayesian hierarchical model for cross-study differential gene expression*

Description

Fits the Bayesian hierarchical model for cross-study differential gene expression.

Usage

```
xde(paramsMcmc, esetList, outputMcmc, batchSize=NULL, NCONC=2)
```

Arguments

`paramsMcmc`	Object of class `XdeParameter`
`esetList`	Object of class `ExpressionSetList`
`outputMcmc`	Object of class `XdeMcmc` (optional)
`batchSize`	Integer or NULL. The number of iterations written to log files before summarizing the chain and then removing. Experimental.
`NCONC`	The number of studies for which a gene must be differentially expressed in the same direction to be considered as concordantly differentially expressed.

Details

Details for fitting the Bayesian model are discussed elsewhere (see citation below and XdeParameterClass vignette)

If an integer is specified for the batchSize, summary statistics for the log-files are calculated for every batchSize iterations. The log files are then removed and the next iteration will start a new log file. This allows one to do many iterations without creating enormous log files. This is only reasonable to do if one has already assessed convergence.

Value

Object of class `XdeMcmc`

Note

See the vignettes for XdeParameterClass and XDE.

Author(s)

R. Scharpf

References

R. Scharpf et al., A Bayesian Model for Cross-Study Differential Gene Expression, Technical Report 158, Johns Hopkins University, Department of Biostatistics, 2007

See Also

`XdeMcmc-class`, `XdeParameter-class`, `ExpressionSetList-class`

Examples

```
## Not run:
  data(expressionSetList)
  xparam <- new("XdeParameter", phenotypeLabel="adenoVsquamous", esetList=expressionSetLi
  iterations(xparam) <- 10
  fit <- xde(xparam, esetList=expressionSetList)
## End(Not run)
```

---

xmcmc *Object of class XdeMcmc*

Description

An object of class XdeMcmc is created by fitting the Bayesian hierarchical model to the `expressionSetList` example data.

Usage

```
data(xmcmc)
```

Details

The xmcmc data example was obtained as described in the XDE vignette.

Examples

```
data(xmcmc)
xmcmc

##ordinarily, one should not need to change the directory in an object
##of class XdeMcmc -- therefore, a replacment method is not defined
pathToLogFiles <- system.file("logFiles", package="XDE")
xmcmc@directory <- pathToLogFiles

##The $ operator can be used to extract chains. For instance, here we
##extract the c2 chain
c2 <- xmcmc$c2
if(require(coda)){
  plot(as.mcmc(c2))
}
```

---

xsScores *Alternative cross-study scores of differential expression*

Description

Alternative cross-study scores of differential expression

Usage

```
xsScores(statistic, N)
```

Arguments

statistic	a matrix of study-specific estimates of effect size. Rows are genes and columns are studies.
N	numerical vector: the number of samples in each study (the length should be the number of columns in statistic)

Value

A matrix of cross-study scores for differential expression ("diffExpressed"), concordant differential expression, and discordant differential expression.

Author(s)

R. Scharpf

References

J.K. Choi, U. Yu, S. Kim, and O.J. Yoo (2003), Combining multiple microarray studies and modeling interstudy variation, Bioinformatics, 19(1) I84-I90.

E. Garrett-Mayer, G. Parmigiani, X. Zhong, L. Cope, and E. Gabrielson (2007), Cross-study validation and combined analysis of gene expression microarray data, Biostatistics, September

R. Scharpf et al., A Bayesian Model for Cross-Study Differential Gene Expression, Technical Report 158, Johns Hopkins University, Department of Biostatistics, 2007

See Also

the GeneMeta package, `ssStatistic`

Examples

```
data(expressionSetList)
t <- ssStatistic(statistic="t", phenotypeLabel="adenoVsquamous", esetList=expressionSet
tScores <- xsScores(t, N=nSamples(expressionSetList))
```

---

zeroNu *Option for not modeling Nu*

Description

Nu is the average expression value in each study.

Usage

```
zeroNu(object, ...)
```

Arguments

`object`	object of class `ExpressionSetList`
`...`	Not implemented

Details

This function should be regarded as experimental.

The nu parameter models the average expression value in each study. Modeling nu allows one to estimate differential expression across studies that may differ in location and scale (as often occurs when multiple platforms are used). The price to pay for modeling nu are additional assumptions (the nuś are assumed Gaussian) and a more heavily parameterized model.

The method zeroNu allows one to fit the Bayesian model without estimating nu:

- each gene is centered at zero

- initial values for the first MCMC are chosen on the basis of empirical starting values

- the initial values for a and rho are set to zero.

- the nu, a, gamma2, and rho parameters are not updated during MCMC

Value

object of class `XdeParameter`

Author(s)

R. Scharpf

References

R. Scharpf et al. (2007), A Bayesian Model for Cross-Study Differential Gene Expression, Technical Report 158, Johns Hopkins University, Department of Biostatistics

Index