

# XDE

February 8, 2012

---

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="ExpressionSetList") 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.

**phenotype** signature(object="ExpressionSetList", varLabel="character") Accessor for the clinical variable. Assumes that the clinical variable has the same name in each study.

**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-methods

*Methods for ExpressionSetList*

---

**Description**

Methods for objects of class ExpressionSetList.

**Usage**

```
phenotype(object, varLabel)
```

**Arguments**

object           A ExpressionSetList.

varLabel         character. Name of the clinical variable.

**Value**

phenotype returns a matrix of the clinical variable where each column is a study. We require that the clinical variable have the same name in each study (each element of the ExpressionSetList object) and that the clinical variable is binary with values 1 or 0.

---

Parameters-class     *Container for XDE parameters*

---

### Description

Container for XDE parameters

### Objects from the Class

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

### Slots

seed: Object of class "integer" ~~  
data: Object of class "numeric" ~~  
phenodata: Object of class "integer" ~~  
G: Object of class "integer" ~~  
Q: Object of class "integer" ~~  
S: Object of class "integer" ~~  
alphaA: Object of class "numeric" ~~  
alphaB: Object of class "numeric" ~~  
betaA: Object of class "numeric" ~~  
betaB: Object of class "numeric" ~~  
pA0: Object of class "numeric" ~~  
pA1: Object of class "numeric" ~~  
pB0: Object of class "numeric" ~~  
pB1: Object of class "numeric" ~~  
nuR: Object of class "numeric" ~~  
nuRho: Object of class "numeric" ~~  
alphaXi: Object of class "numeric" ~~  
betaXi: Object of class "numeric" ~~  
c2Max: Object of class "numeric" ~~  
alphaEta: Object of class "numeric" ~~  
betaEta: Object of class "numeric" ~~  
pOmega0: Object of class "numeric" ~~  
lambdaOmega: Object of class "numeric" ~~  
lambdaKappa: Object of class "numeric" ~~  
gamma2: Object of class "numeric" ~~  
c2: Object of class "numeric" ~~  
tau2Rho: Object of class "numeric" ~~  
tau2R: Object of class "numeric" ~~  
a: Object of class "numeric" ~~

```

b: Object of class "numeric" ~~
l: Object of class "numeric" ~~
t: Object of class "numeric" ~~
lambda: Object of class "numeric" ~~
theta: Object of class "numeric" ~~
phi: Object of class "numeric" ~~
sigma2: Object of class "numeric" ~~
r: Object of class "numeric" ~~
rho: Object of class "numeric" ~~
nu: Object of class "numeric" ~~
delta: Object of class "numeric" ~~
Delta: Object of class "numeric" ~~
xi: Object of class "numeric" ~~

```

### Methods

```

"[[<-" signature(x = "Parameters"):...
"[" signature(x = "Parameters"):...
"$<-" signature(x = "Parameters"):...
$ signature(x = "Parameters"):...
coerce signature(from = "XdeParameter", to = "Parameters"):...
show signature(object = "Parameters"):...

```

### Examples

```
showClass("Parameters")
```

---

RUpdates

*R interface to MCMC updates in C*


---

### Description

R interface to MCMC updates in C

### Usage

```

rupdateANu(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateBDDelta(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateBDDelta(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateTau2RhoNu(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateTau2RDDelta(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateNu(object, nAccept = 0L, dryrun = FALSE)
rupdateDDelta(object, nAccept = 0L, dryrun = FALSE)
rupdateC2(object, nTry=10L, nAccept = 0L, dryrun = FALSE)
rupdateC2DDelta(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateGamma2(object, nAccept=0L, dryrun=FALSE)

```

```

rupdateGamma2Nu(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateRDDelta(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateRC2(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateRhoNu(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateRhoGamma2(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateSigma2(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdatePhi(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateTheta(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateLambda(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateT(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateL(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateXi(object, nAccept = 0L, dryrun = FALSE, one.delta=FALSE)
rupdateDeltaDDelta(object, nTry = 10L, nAccept = 0L, dryrun = FALSE, one.delta=F
rupdateLSigma2(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateTSigma2(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateLambdaPhi(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateThetaPhi(object, nTry = 10L, nAccept = 0L, epsilon = 0.2, dryrun = FALSE)
rupdateDeltaDDelta_MRF(object, nTry = 10L, nAccept = 0L, dryrun = FALSE, one.del
rupdateAlpha_MRF(object, nTry = 10L, nAccept = 0L, epsilon=0.2, dryrun = FALSE,
rupdateBeta_MRF(object, nTry = 10L, nAccept = 0L, epsilon=0.2, dryrun = FALSE, c
rupdateBetag_MRF(object, nTry = 10L, nAccept = 0L, epsilon=0.2, dryrun = FALSE)

```

### Arguments

<code>object</code>	Object of class <code>Parameters</code>
<code>nTry</code>	Number of MCMC samples
<code>nAccept</code>	Number of samples accepted. Only relevant for Metropolis-Hastings updates
<code>epsilon</code>	Tuning parameter for the proposal. Only relevant for Metropolis-Hastings updates
<code>dryrun</code>	Return a list of the arguments that are to be passed to the low-level C function, without calling the C function
<code>one.delta</code>	Logical. If true, we assume that a gene is differentially expressed in all studies or in none of the studies

### Details

The parameters that are updated are indicated by the function name. For example, `rupdateSigma2` updates the parameter `sigma2`.

### Value

A object of class `Parameters`. The returned object contains any updates from the MCMC, as well as a new seed.

### Author(s)

RS

### See Also

[Parameters](#)

---

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, seed, 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

**seed:** Object of class `integer`. Seed used for simulating random numbers.

**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

**one.delta:** Logical. If TRUE, a gene is assumed to be differentially in all studies or none of the studies.

## 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
```

---

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**

```
## Not run:
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]

## End(Not run)
```

---

```
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, burnin=0)
```

**Arguments**

<code>object</code>	Object of class <code>XdeMcmc</code>
<code>NCONC</code>	Integer: number of studies for which the gene must be differentially expressed (in the same direction) to be classified as concordant differential expression
<code>NDIFF</code>	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.
<code>burnin</code>	Integer: number of MCMC iterations for the burnin. Posterior means are computed from the MCMC samples following burnin.

**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  $G \times 2$  matrix, where G is the number of genes in common across all studies).  $P + N \leq S$ , 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  $P + N \geq NCONC$ , where NCONC is specified by the user.
5. differential expression is simply  $|P| + |N| \geq NDIFF$ . 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  $G \times 3$  matrix.

**Author(s)**

RS

**See Also**

[posteriorAvg](#)

---

<code>empiricalStart</code>	<i>Empirical starting values for the MCMC</i>
-----------------------------	---

---

**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**

<code>object</code>	An object of class <code>ExpressionSetList</code>
<code>zeroNu</code>	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
<code>phenotypeLabel</code>	character: binary phenotype. phenotypeLabel must be in the varLabels of each ExpressionSet object
<code>one.delta</code>	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.
<code>T_THRESH</code>	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**

```
library(XDE)
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 XdeParameter
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 *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](http://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	<i>Values for the first MCMC iteration</i>
-----------	--

---

## 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="adenoVsSquamous")
hyperparameters(xlist)
```

---

<code>iterations</code>	<i>Number of MCMC iterations</i>
-------------------------	----------------------------------

---

**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 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	<i>pairs function for high-throughput data</i>
---------------	--

---

### 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	<i>Accessor and replacement methods for posterior averages of differential expression</i>
--------------	---

---

### 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 <code>XdeMcmc</code>
value	A matrix of dimension $G \times 3$ , 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  $G \times 3$ , 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	<i>Seed for the MCMC</i>
------	--------------------------

---

**Description**

Setting a seed is useful for reproducing MCMC chains

**Usage**

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

**Arguments**

object	An object of <code>XdeParameter</code> or <code>XdeMcmc</code>
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	<i>Calculate single study estimates of effect size</i>
-------------	--

---

**Description**

Calculate single study estimates of effect size for lists of `ExpressionSets`

**Usage**

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

**Arguments**

<code>statistic</code>	Character string indicating Welch t-statistic (t), SAM (sam), or a z-statistic (z)
<code>phenotypeLabel</code>	Character string indicating the name of the binary covariate
<code>esetList</code>	An object of class <code>ExpressionSetList</code>
<code>...</code>	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 study 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", "royalblue4"))
```

**Arguments**

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

**Value**

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

**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**

<code>x</code>	An object of class <code>XdeParameter</code>
<code>...</code>	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

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,
center=TRUE, ...)
```

**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.
center	Logical. If TRUE, each study is centered to have mean zero.
...	Additional arguments passed to xdeFit.

**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, JASA 2009, p1295–1310.

## 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**

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

**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="adenoVsSquamous", esetList=expressionSetList)
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**

<code>object</code>	object of class <code>ExpressionSetList</code>
<code>...</code>	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 nu's 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

## \*Topic **classes**

- ExpressionSetList-class, 1
- Parameters-class, 3
- XdeMcmc-class, 6
- XdeParameter-class, 7

## \*Topic **datasets**

- expressionSetList, 12
- xmcmc, 26

## \*Topic **dplot**

- symbolsInteresting, 22

## \*Topic **hplot**

- pairs-methods, 18

## \*Topic **htest**

- xsScores, 27

## \*Topic **interface**

- RUpdates, 4

## \*Topic **iteration**

- RUpdates, 4

## \*Topic **manip**

- calculatePosteriorAvg, 10
- ExpressionSetList-methods, 2

## \*Topic **methods**

- burnin, 9
- empiricalStart, 11
- ExpressionSetList-methods, 2
- firstMcmc, 13
- geneCenter, 14
- hyperparameters, 14
- iterations, 15
- lastMcmc, 16
- output, 17
- pairs-methods, 18
- posteriorAvg, 18
- seed, 19
- ssStatistic, 19
- standardizeSamples, 21
- studyCenter, 21
- thin, 23
- tuning, 24
- updates, 24
- zeroNu, 28

## \*Topic **models**

- RUpdates, 4

- xde, 25

- xsScores, 27

## \*Topic **multivariate**

- xde, 25

- .integrativeCorrelationFilter, ExpressionSetList-method (ExpressionSetList-class), 1

- .standardizedDelta, XdeMcmc-method (XdeMcmc-class), 6

- [, ExpressionSetList-method (ExpressionSetList-class), 1

- [[, Parameters-method (Parameters-class), 3

- [[<-, Parameters-method (Parameters-class), 3

- \$(XdeMcmc-class), 6

- \$, Parameters-method (Parameters-class), 3

- \$, XdeMcmc-method (XdeMcmc-class), 6

- \$<-, Parameters-method (Parameters-class), 3

- bayesianEffectSize (XdeMcmc-class), 6

- bayesianEffectSize, XdeMcmc-method (XdeMcmc-class), 6

- bayesianEffectSize<- (XdeMcmc-class), 6

- bayesianEffectSize<- , XdeMcmc, matrix-method (XdeMcmc-class), 6

- burnin, 8, 9, 17

- burnin, XdeParameter-method (XdeParameter-class), 7

- burnin<- (burnin), 9

- burnin<- , XdeParameter, logical-method (XdeParameter-class), 7

- calculateBayesianEffectSize (XdeMcmc-class), 6

- calculateBayesianEffectSize, XdeMcmc-method (XdeMcmc-class), 6

- calculatePosteriorAvg, 6, 10, 18

- calculatePosteriorAvg, XdeMcmc-method  
     (XdeMcmc-class), 6
- class.AssayData, 1
- coerce, list, ExpressionSetList-method  
     (ExpressionSetList-class),  
     1
- coerce, XdeParameter, Parameters-method  
     (Parameters-class), 3
- coerce, XdeParameter, Params-method  
     (XdeParameter-class), 7
- dim, ExpressionSetList-method  
     (ExpressionSetList-class),  
     1
- directory (XdeMcmc-class), 6
- directory, XdeMcmc-method  
     (XdeMcmc-class), 6
- directory, XdeParameter-method  
     (XdeParameter-class), 7
- directory<- (XdeParameter-class),  
     7
- directory<- , XdeParameter-method  
     (XdeParameter-class), 7
- empiricalStart, 11
- expressionSetList, 12
- ExpressionSetList-class, 7, 9, 12,  
     14, 22, 26
- ExpressionSetList-class, 1
- ExpressionSetList-methods, 2
- featureNames, ExpressionSetList-method  
     (ExpressionSetList-class),  
     1
- featureNames, XdeMcmc-method  
     (XdeMcmc-class), 6
- firstMcmc, 8, 13, 16
- firstMcmc, XdeParameter-method  
     (XdeParameter-class), 7
- firstMcmc<- (firstMcmc), 13
- firstMcmc<- , XdeParameter, environment-method  
     (XdeParameter-class), 7
- firstMcmc<- , XdeParameter, list-method  
     (XdeParameter-class), 7
- geneCenter, 1, 14, 22
- geneCenter, ExpressionSetList-method  
     (ExpressionSetList-class),  
     1
- hyperparameters, 14
- hyperparameters, XdeParameter-method  
     (XdeParameter-class), 7
- hyperparameters<-  
     (hyperparameters), 14
- hyperparameters<- , XdeParameter-method  
     (XdeParameter-class), 7
- initialize, XdeMcmc-method  
     (XdeMcmc-class), 6
- initialize, XdeParameter-method  
     (XdeParameter-class), 7
- iterations, 15
- iterations, XdeMcmc-method  
     (XdeMcmc-class), 6
- iterations, XdeParameter-method  
     (XdeParameter-class), 7
- iterations<- (iterations), 15
- iterations<- , XdeParameter, integer-method  
     (XdeParameter-class), 7
- iterations<- , XdeParameter, numeric-method  
     (XdeParameter-class), 7
- lapply, ExpressionSetList-method  
     (ExpressionSetList-class),  
     1
- lastMcmc, 13, 16
- lastMcmc, XdeMcmc-method  
     (XdeMcmc-class), 6
- lastMcmc<- (lastMcmc), 16
- list, 1
- nrow, ExpressionSetList-method  
     (ExpressionSetList-class),  
     1
- nrow, XdeMcmc-method  
     (XdeMcmc-class), 6
- nSamples  
     (ExpressionSetList-class),  
     1
- nSamples, ExpressionSetList-method  
     (ExpressionSetList-class),  
     1
- output, 8, 9, 17, 23
- output, XdeMcmc-method  
     (XdeMcmc-class), 6
- output, XdeParameter-method  
     (XdeParameter-class), 7
- output<- (output), 17
- output<- , XdeParameter-method  
     (XdeParameter-class), 7
- pairs, data.frame-method  
     (pairs-methods), 18
- pairs, matrix-method  
     (pairs-methods), 18

- pairs-methods, 18
- Parameters, 5
- Parameters-class, 3
- pca, ExpressionSetList-method
  - (ExpressionSetList-class), 1
- pData, ExpressionSetList-method
  - (ExpressionSetList-class), 1
- phenotype
  - (ExpressionSetList-methods), 2
- phenotype, ExpressionSetList, character-method
  - (ExpressionSetList-class), 1
- phenotypeLabel
  - (XdeParameter-class), 7
- phenotypeLabel, XdeParameter-method
  - (XdeParameter-class), 7
- phenotypeLabel<-
  - (XdeParameter-class), 7
- phenotypeLabel<-, XdeParameter, character-method
  - (XdeParameter-class), 7
- plot, XdeMcmc, ANY-method
  - (XdeMcmc-class), 6
- plot, XdeMcmc-method
  - (XdeMcmc-class), 6
- posteriorAvg, 11, 18
- posteriorAvg, XdeMcmc-method
  - (XdeMcmc-class), 6
- posteriorAvg<- (posteriorAvg), 18
- posteriorAvg<-, XdeMcmc, matrix-method
  - (XdeMcmc-class), 6
  
- rupdateAlpha\_MRF (RUpdates), 4
- rupdateANu (RUpdates), 4
- rupdateBDDelta (RUpdates), 4
- rupdateBeta\_MRF (RUpdates), 4
- rupdateBetag\_MRF (RUpdates), 4
- rupdateC2 (RUpdates), 4
- rupdateC2DDelta (RUpdates), 4
- rupdateDDelta (RUpdates), 4
- rupdateDeltaDDelta (RUpdates), 4
- rupdateDeltaDDelta\_MRF
  - (RUpdates), 4
- rupdateGamma2 (RUpdates), 4
- rupdateGamma2Nu (RUpdates), 4
- rupdateL (RUpdates), 4
- rupdateLambda (RUpdates), 4
- rupdateLambdaPhi (RUpdates), 4
- rupdateLSigma2 (RUpdates), 4
- rupdateNu (RUpdates), 4
- rupdatePhi (RUpdates), 4
- rupdateRC2 (RUpdates), 4
- rupdateRDDelta (RUpdates), 4
- rupdateRhoGamma2 (RUpdates), 4
- rupdateRhoNu (RUpdates), 4
- RUpdates, 4
- rupdateSigma2 (RUpdates), 4
- rupdateT (RUpdates), 4
- rupdateTau2RDDelta (RUpdates), 4
- rupdateTau2RhoNu (RUpdates), 4
- rupdateTheta (RUpdates), 4
- rupdateThetaPhi (RUpdates), 4
- rupdateTSigma2 (RUpdates), 4
- rupdateXi (RUpdates), 4
  
- savedIterations
  - (XdeParameter-class), 7
- savedIterations, XdeParameter-method
  - (XdeParameter-class), 7
- seed, 8, 19
- seed, XdeMcmc-method
  - (XdeMcmc-class), 6
- seed, XdeParameter-method
  - (XdeParameter-class), 7
- seed<- (seed), 19
- seed<-, XdeParameter, integer-method
  - (XdeParameter-class), 7
- seed<-, XdeParameter, numeric-method
  - (XdeParameter-class), 7
- show, Parameters-method
  - (Parameters-class), 3
- show, XdeMcmc-method
  - (XdeMcmc-class), 6
- show, XdeParameter-method
  - (XdeParameter-class), 7
- showIterations
  - (XdeParameter-class), 7
- showIterations, XdeParameter-method
  - (XdeParameter-class), 7
- showIterations<-
  - (XdeParameter-class), 7
- showIterations<-, XdeParameter-method
  - (XdeParameter-class), 7
- ssStatistic, 19, 27
- standardizeSamples, 2, 21
- standardizeSamples, ExpressionSetList-method
  - (ExpressionSetList-class), 1
- studyCenter, 2, 14, 21
- studyCenter, ExpressionSetList-method
  - (ExpressionSetList-class), 1
- studyNames (XdeParameter-class), 7

studyNames, XdeMcmc-method  
(*XdeMcmc-class*), 6

studyNames, XdeParameter-method  
(*XdeParameter-class*), 7

studyNames<-  
(*XdeParameter-class*), 7

studyNames<- , XdeParameter-method  
(*XdeParameter-class*), 7

symbolsInteresting, 22

thin, 9, 23

thin, XdeParameter-method  
(*XdeParameter-class*), 7

thin<- (*thin*), 23

thin<- , XdeParameter, numeric-method  
(*XdeParameter-class*), 7

tuning, 9, 24

tuning, XdeParameter-method  
(*XdeParameter-class*), 7

tuning<- (*tuning*), 24

tuning<- , XdeParameter-method  
(*XdeParameter-class*), 7

updates, 9, 24

updates, XdeParameter-method  
(*XdeParameter-class*), 7

updates<- (*updates*), 24

updates<- , XdeParameter-method  
(*XdeParameter-class*), 7

vector, 1

xde, 25

XdeMcmc-class, 2, 16, 17, 26

XdeMcmc-class, 6

XdeParameter-class, 2, 7, 10, 12, 13,  
16, 17, 26

XdeParameter-class, 7

xmcmc, 26

xsScores, 27

zeroNu, 2, 12, 28

zeroNu, ExpressionSetList-method  
(*ExpressionSetList-class*),  
1