Package ‘dcmle’

March 12, 2016

Type   Package
Title   Hierarchical Models Made Easy with Data Cloning
Version 0.3-1
Date 2016-03-11
Author Peter Solymos
Maintainer Peter Solymos <solymos@ualberta.ca>

Description S4 classes around infrastructure provided by the 'coda' and 'dclone' packages to make package development easy as a breeze with data cloning for hierarchical models.

License GPL-2
Depends R (>= 2.15.0), dclone (>= 2.0-0)
Imports coda, methods, stats4, lattice
Suggests MASS, parallel, rjags
SystemRequirements JAGS (>= 3.0.0)

URL https://groups.google.com/forum/#!forum/dclone-users,
    http://datacloning.org

LazyLoad yes
LazyData true
NeedsCompilation no
Repository CRAN
Date/Publication 2016-03-12 08:51:18

R topics documented:

dcmle-package ......................................................... 2
chanames ............................................................. 4
codaMCMC-class ...................................................... 5
crosscorr.plot ......................................................... 7
custommodel-class .................................................... 8
Description

S4 classes around infrastructure provided by the dclone package to make package development with data cloning for hierarchical models easy as a breeze.

Details

The package defines S4 object classes for plain BUGS models ("gsFit", after BU*GS*/JA*GS*), and BUGS models made ready for data cloning ("dcFit"). It also defines virtual classes for S3 object classes defined in the dclone and coda packages.

The S4 class "dcmle" is a fitted model object containing MCMC results as returned by the dcmle function. These object classes are easily extensible to allow inclusion into functions fitting specific models to the data (see Examples).

Author(s)

Peter Solymos
Maintainer: Peter Solymos <solymos@ualberta.ca>
References

Forum: https://groups.google.com/forum/#!forum/dclone-users
Issues: https://github.com/datacloning/dcmle/issues
Data cloning website: http://datacloning.org

See Also

Fitting wrapper function: dcmle
Object classes: "dcmle", "codaMCMC", "dcCodaMCMC"
Creator functions makeGsFit and makeDcFit

Examples

```r
## Data and model taken from Ponciano et al. 2009
## Ecology 90, 356-362.

## Function to create template object for the Beverton-Holt model
## R CMD check will not choke on character representation of model
## the convenient makedcFit creator function is used here
bevholtFit <-
function(y) {
  makedcFit(
    data = list(ncl=1, n=length(y), Y=dcdim(data.matrix(y))),
    model = structure(
      c("model = ",
         " for (k in 1:ncl) {",
            " for(i in 2:(n+1)) {",
            " Y[(i-1), k] ~ dpois(exp(X[i, k]))",
            " X[i, k] ~ dnorm(mu[i, k], 1 / sigma^2)",
            " mu[i,k] <- X[(i-1),k]+log(lambda)-log(1+beta*exp(X[(i-1),k]))",
            " }",
            " X[1, k] ~ dnorm(mu0, 1 / sigma^2)",
            " }",
            " beta ~ dlnorm(-1, 1)",
            " sigma ~ dlnorm(0, 1)",
            " tmp ~ dlnorm(0, 1)",
            " lambda ~ tmp + 1",
            " mu0 ~ log(2) + log(lambda) - log(1 + beta * 2)",
          "})",
      class = "custommodel"),
    multiply = "ncl",
    unchanged = "n",
    params <- c("lambda","beta","sigma")
  )

## S4 class 'bevholtMle' extends the 'dcmle' class
## it can have additional slots
setClass("bevholtMle",
  representation(y="numeric", title="character"),
  contains = "dcmle")

## Function to fit the Beverton-Holt model to data
```
```r
bevholt <- function(y, n.clones, ...) {
  new("bevholtMle",
      dcmle(bevholtFit(y), n.clones=n.clones, ...),
      y = y,
      title = "Beverton-Holt Model")
}
```nn
## Show method with appropriate heading

```r
setMethod("show", "bevholtMle", function(object)
  show(summary(as(object, "dcmle"), object@title)))
```
nn
```r
paurelia <- c(17,29,39,63,185,258,267,392,510,
  570,650,560,575,650,550,480,520,500)
## Not run:
```r
  (m <- bevholt(paurelia, n.clones=2, n.iter=1000))
  vcov(m)
  m@y

## End(Not run)
```

---

**Description**

*cod*a package related generic functions.

**Usage**

```r
chanames(x, ...)
```

```r
varnames(x, ...)
```

**Arguments**

- **x** MCMC object.
- **...** Other arguments.

**Value**

See corresponding help pages.

**Author(s)**

Peter Solymos

**See Also**

- `chanames varnames`

Description

An S4 representation of an mcmc.lits object of the coda package.

Objects from the Class

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

Slots

values: Object of class "numeric", values from the posterior sample of length niter * nvar * nchains.

varnames: Object of class "character", variable names.

start: Object of class "integer", start of iterations.

end: Object of class "integer", end of iterations.

thin: Object of class "integer", thinning value.

nchains: Object of class "integer", number of chains.

niter: Object of class "integer", number of iterations.

nvar: Object of class "integer", number of variables

Methods

[ signature(x = "codaMCMC"): ...

[[ signature(x = "codaMCMC"): ...

acfplot signature(x = "codaMCMC"): ...

as.array signature(x = "codaMCMC"): ...

as.matrix signature(x = "codaMCMC"): ...

as.mcmc.list signature(x = "codaMCMC"): ...

autocorr.diag signature(mcmc.obj = "codaMCMC"): ...

chanames signature(x = "codaMCMC"): ...

chisq.diag signature(x = "codaMCMC"): ...

coef signature(object = "codaMCMC"): ...

coerce signature(from = "codaMCMC", to = "dcmle"): ...

coerce signature(from = "codaMCMC", to = "MCMCList"): ...

coerce signature(from = "dcmle", to = "codaMCMC"): ...

coerce signature(from = "MCMCList", to = "codaMCMC"): ...

confint signature(object = "codaMCMC"): ...

crosscorr.plot signature(x = "codaMCMC"): ...
codaMCMC-class

crosscorr signature(x = "codaMCMC"): ...
dcdiag signature(x = "codaMCMC"): ...
dcsd signature(object = "codaMCMC"): ...
detable signature(x = "codaMCMC"): ...
densityplot signature(x = "codaMCMC"): ...
densplot signature(x = "codaMCMC"): ...
end signature(x = "codaMCMC"): ...
frequency signature(x = "codaMCMC"): ...
gelman.diag signature(x = "codaMCMC"): ...
gelman.plot signature(x = "codaMCMC"): ...
geweke.diag signature(x = "codaMCMC"): ...
head signature(x = "codaMCMC"): ...
heidel.diag signature(x = "codaMCMC"): ...
lambdamax.diag signature(x = "codaMCMC"): ...
mepar signature(x = "codaMCMC"): ...
nchain signature(x = "codaMCMC"): ...
nclones signature(x = "codaMCMC"): ...
niter signature(x = "codaMCMC"): ...
nvar signature(x = "codaMCMC"): ...
pairs signature(x = "codaMCMC"): ...
plot signature(x = "codaMCMC", y = "missing"): ...
qqmath signature(x = "codaMCMC"): ...
quantile signature(x = "codaMCMC"): ...
raftery.diag signature(x = "codaMCMC"): ...
show signature(object = "codaMCMC"): ...
stack signature(x = "codaMCMC"): ...
start signature(x = "codaMCMC"): ...
summary signature(object = "codaMCMC"): ...
tail signature(x = "codaMCMC"): ...
thin signature(x = "codaMCMC"): ...
time signature(x = "codaMCMC"): ...
traceplot signature(x = "codaMCMC"): ...
varnames signature(x = "codaMCMC"): ...
vco signature(object = "codaMCMC"): ...
window signature(x = "codaMCMC"): ...
xyplot signature(x = "codaMCMC"): ...
crosscorr.plot

Author(s)

Peter Solymos

See Also

mcmc.list

Examples

showClass("codaMCMC")

crosscorr.plot  Generic after similar coda function

Description

Generic after similar coda function

Usage

crosscorr.plot(x, ...)

Arguments

x      MCMC object.
...
Other arguments.

Value

See corresponding help page

Author(s)

Peter Solymos

See Also

crosscorr.plot
**custommodel-class**  
*Class "custommodel"

**Description**

Stands for the 'custommodel' S3 class from *dclone* package.

**Objects from the Class**

A virtual Class: No objects may be created from it.

**Extends**

Class "dcModel", directly.

**Methods**

No methods defined with class "custommodel" in the signature.

**Author(s)**

Peter Solymos

**See Also**

custommodel

**Examples**

```
showClass("custommodel")
```

---

**dcArgs-class**  
*Class "dcArgs"

**Description**

A class union for NULL and "character".

**Objects from the Class**

A virtual Class: No objects may be created from it.

**Methods**

No methods defined with class "dcArgs" in the signature.
dcCodaMCMC-class

Author(s)
Peter Solymos

Examples
showClass("dcArgs")

dcCodaMCMC-class  Class "dcCodaMCMC"

Description
An S4 representation of an mcmc.lits object of the coda package, with data cloning attributes from dclone package (the mcmc.list.dc class).

Objects from the Class
Objects can be created by calls of the form new("dcCodaMCMC", ...).

Slots
dctable: Object of class "dcTable", data cloning based iterative posterior statistics based on dctable.
dcdiag: Object of class "dcDiag", data cloning convergence diagnostics based on dcdiag.
nclones: Object of class "nClones", number of clones.
values: Object of class "numeric", same as in "codaMCMC" class.
varnames: Object of class "character", same as in "codaMCMC" class.
start: Object of class "integer", same as in "codaMCMC" class.
end: Object of class "integer", same as in "codaMCMC" class.
thin: Object of class "integer", same as in "codaMCMC" class.
nchains: Object of class "integer", same as in "codaMCMC" class.
niter: Object of class "integer", same as in "codaMCMC" class.
nvar: Object of class "integer", same as in "codaMCMC" class.

Extends
Class "codaMCMC", directly.
Methods

[ signature(x = "dcCodaMCMC"): ...
[[ signature(x = "dcCodaMCMC"): ...
coerce signature(from = "dcCodaMCMC", to = "dcMle"): ...
coerce signature(from = "dcCodaMCMC", to = "MCMClist"): ...
coerce signature(from = "dcMle", to = "dcCodaMCMC"): ...
coerce signature(from = "MCMClist", to = "dcCodaMCMC"): ...
confint signature(object = "dcCodaMCMC"): ...
dcdiag signature(x = "dcCodaMCMC"): ...
dctable signature(x = "dcCodaMCMC"): ...
nclones signature(x = "dcCodaMCMC"): ...
str signature(object = "dcCodaMCMC"): ...
summary signature(object = "dcCodaMCMC"): ...

Author(s)

Peter Solymos

See Also

jagsNfit

Examples

showClass("dcCodaMCMC")

dcDiag-class Class "dcDiag"

Description

Virtual class for data cloning convergence diagnostics.

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

No methods defined with class "dcDiag" in the signature.

Author(s)

Peter Solymos
**dcdiag-class**

**See Also**
- `dcdiag`

**Examples**
```
showClass("dcDiag")
```

---

**Description**

Stands for the 'dcdiag' S3 class from `dclone` package.

**Objects from the Class**

A virtual Class: No objects may be created from it.

**Extends**

Class "dcDiag", directly.

**Methods**

No methods defined with class "dcdiag" in the signature.

**Author(s)**

Peter Solymos

**See Also**
- `dcdiag`

**Examples**
```
showClass("dcdiag")
```
dcFit-class

Class "dcFit"

Description

Compendium for data cloning

Objects from the Class

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

Slots

- multiply: Object of class "dcArgs", same as corresponding dc.fit argument.
- unchanged: Object of class "dcArgs", same as corresponding dc.fit argument.
- update: Object of class "dcArgs", same as corresponding dc.fit argument.
- updatefun: Object of class "dcFunction", same as corresponding dc.fit argument.
- initsfun: Object of class "dcFunction", same as corresponding dc.fit argument.
- flavour: Object of class "character", same as corresponding dc.fit argument, default is "jags". It can also be "winbugs", "openbugs", or "brugs" referring to the argument of bugs.fit. in which case flavour will be treated as "bugs".
- data: Object of class "list", same as corresponding dc.fit argument.
- model: Object of class "dcModel", same as corresponding dc.fit argument.
- params: Object of class "dcParams", same as corresponding dc.fit argument.
- inits: Object of class "dcInits", same as corresponding dc.fit argument.

Extends

Class "gsFit", directly.

Methods

show signature(object = "dcFit"): ...

Author(s)

Peter Solymos

See Also

dc.fit, makeDcFit

Examples

showClass("dcFit")
dcFunction-class

Description
Virtual class for BUGS/JAGS models defined as functions.

Objects from the Class
A virtual Class: No objects may be created from it.

Methods
No methods defined with class "dcFunction" in the signature.

Author(s)
Peter Solymos

Examples
showClass("dcFunction")

--

dcInits-class

Class "dcInits"

Description
Virtual class for initial values.

Objects from the Class
A virtual Class: No objects may be created from it.

Methods
No methods defined with class "dcInits" in the signature.

Author(s)
Peter Solymos

Examples
showClass("dcInits")
Description

This function is a wrapper to fit the model to the data and obtain MLE point estimates and asymptotic standard errors based on the estimate of the Fisher information matrix (theory given by Lele et al. 2007, 2010, software implementation is given in Solymos 2010).

Usage

dcmle(x, params, n.clones = 1, cl = NULL, nobs, ...)

Arguments

x an object of class "gsFit" or "dcFit".
params character, vector of model parameters to monitor.
n.clones integer, vector for the number of clones used in fitting.
cl cluster object (snow type cluster) or number of cores (multicore type forking), optional.
nobs number of observations, optional.
... other arguments passed to underlying functions (see Details).

Details

The function uses slots of the input object and passes them as arguments to underlying functions (jags.fit, jags.parfit, bugs.fit, dc.fit, dc.parfit).

Value

An object of class "dcmle".

Author(s)

Peter Solymos, <solymos@ualberta.ca>

References


See Also

For additional arguments: `jags.fit`, `jags.parfit`, `bugs.fit`, `dc.fit`, `dc.parfit`.

Object classes: "dcmler"

Creator functions `makeGsf` and `makeDcfit`

Examples

```r
## Data and model taken from Ponciano et al. 2009
## Ecology 90, 356-362.
pauropia <- c(17, 29, 39, 63, 185, 258, 267, 392, 510,
             570, 650, 560, 575, 650, 550, 680, 520, 500)
paramecium <- new("dcFit")
paramecium@data <- list(
  ncl=1,
  n=length(pauropia),
  Y=dcdim(data.matrix(pauropia)))
paramecium@model <- function() {
  for (k in 1:ncl) {
    for (i in 2:(n+1)) {
      Y[i, k] ~ dpois(exp(X[i, k])) # observations
      X[i, k] ~ dnorm(mu[i, k], 1 / sigma^2) # state
      mu[i, k] <- X[(i-1), k] + log(lambda) - log(1 + beta * exp(X[(i-1), k]))
    }
    X[1, k] ~ dnorm(mu0, 1 / sigma^2) # state at t0
  }
  beta ~ dnorm(-1, 1) # Priors on model parameters
  sigma ~ dnorm(0, 1)
  tmp ~ dlnorm(0, 1)
  lambda <- tmp + 1
  mu0 <- log(2) + log(lambda) - log(1 + beta * 2)
}
paramecium@multiply <- "ncl"
paramecium@unchanged <- "n"
paramecium@params <- c("lambda","beta","sigma")
## Not run:
(m1 <- dcmlle(paramecium, n.clones=1, n.iter=1000))
(m2 <- dcmlle(paramecium, n.clones=2, n.iter=1000))
(m3 <- dcmlle(paramecium, n.clones=1:3, n.iter=1000))
c1 <- makePSoCKcluster(3)
(m4 <- dcmlle(paramecium, n.clones=2, n.iter=1000, cl=c1))
(m5 <- dcmlle(paramecium, n.clones=1:3, n.iter=1000, cl=c1))
(m6 <- dcmlle(paramecium, n.clones=1:3, n.iter=1000, cl=c1,
partype="parchains"))
(m7 <- dcmlle(paramecium, n.clones=1:3, n.iter=1000, cl=c1,
partype="both"))
stopCluster(c1)
## End(Not run)
```
dcml-class  

Class "dcml"  

Description  
Fitted model object from \texttt{dcml}.

Objects from the Class  
Objects can be created by calls of the form \texttt{new("dcml", \ldots).}

Slots  
call: Object of class "language", the call.  
coef: Object of class "numeric", coefficients (posterior means).  
fullcoef: Object of class "numeric", full coefficients, possibly with fixed values.  
vcov: Object of class "matrix", variance covariance matrix.  
details: Object of class "dcCodaMCMC", the fitted model object.  
nobs: Object of class "integer", number of observations, optional.  
method: Object of class "character".

Methods  
[ signature(x = "dcml"): ...  
[ [ signature(x = "dcml"): ...  
\texttt{acfplot} signature(x = "dcml"): ...  
\texttt{as.array} signature(x = "dcml"): ...  
\texttt{as.matrix} signature(x = "dcml"): ...  
\texttt{as.mcmc.list} signature(x = "dcml"): ...  
\texttt{autocorr.diag} signature(mcmc.obj = "dcml"): ...  
\texttt{chanames} signature(x = "dcml"): ...  
\texttt{chisq.diag} signature(x = "dcml"): ...  
\texttt{coef} signature(object = "dcml"): ...  
\texttt{coerce} signature(from = "codaMCMC", to = "dcml"): ...  
\texttt{coerce} signature(from = "dcCodaMCMC", to = "dcml"): ...  
\texttt{coerce} signature(from = "dcml", to = "codaMCMC"): ...  
\texttt{coerce} signature(from = "dcml", to = "dcCodaMCMC"): ...  
\texttt{coerce} signature(from = "dcml", to = "MCMClist"): ...  
\texttt{coerce} signature(from = "MCMClist", to = "dcml"): ...  
\texttt{confint} signature(object = "dcml"): ...
crosscorr.plot signature(x = "dcmle"): ...
crosscorr signature(x = "dcmle"): ...
dcdiag signature(x = "dcmle"): ...
dcsd signature(object = "dcmle"): ...
detable signature(x = "dcmle"): ...
densityplot signature(x = "dcmle"): ...
densplot signature(x = "dcmle"): ...
end signature(x = "dcmle"): ...
frequency signature(x = "dcmle"): ...
gelman.diag signature(x = "dcmle"): ...
gelman.plot signature(x = "dcmle"): ...
geweke.diag signature(x = "dcmle"): ...
head signature(x = "dcmle"): ...
heidel.diag signature(x = "dcmle"): ...
lambdamax.diag signature(x = "dcmle"): ...
mcp signature(x = "dcmle"): ...
nchain signature(x = "dcmle"): ...
nclones signature(x = "dcmle"): ...
niter signature(x = "dcmle"): ...
nvar signature(x = "dcmle"): ...
pairs signature(x = "dcmle"): ...
plot signature(x = "dcmle", y = "missing"): ...
qqmath signature(x = "dcmle"): ...
quantile signature(x = "dcmle"): ...
raftery.diag signature(x = "dcmle"): ...
show signature(object = "dcmle"): ...
stack signature(x = "dcmle"): ...
start signature(x = "dcmle"): ...
str signature(object = "dcmle"): ...
summary signature(object = "dcmle"): ...
tail signature(x = "dcmle"): ...
thin signature(x = "dcmle"): ...
time signature(x = "dcmle"): ...
traceplot signature(x = "dcmle"): ...
update signature(object = "dcmle"): ...
vnames signature(x = "dcmle"): ...
vcov signature(object = "dcmle"): ...
window signature(x = "dcmle"): ...
xyplot signature(x = "dcmle"): ...
dcModel-class

Author(s)

Peter Solymos

See Also

dcmle

Examples

showClass("dcmle")

dcModel-class | Class "dcModel"

Description

Virtual class for BUGS/JAGS models.

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

No methods defined with class "dcModel" in the signature.

Author(s)

Peter Solymos

Examples

showClass("dcModel")
dcParams-class

Class "dcParams"

Description
Virtual class for model parameters to monitor.

Objects from the Class
A virtual Class: No objects may be created from it.

Methods
No methods defined with class "dcParams" in the signature.

Author(s)
Peter Solymos

Examples

dcTable-class

Class "dcTable"

Description
Posterior statistics from iterative fit, virtual class.

Objects from the Class
A virtual Class: No objects may be created from it.

Methods
No methods defined with class "dcTable" in the signature.

Author(s)
Peter Solymos

Examples

showClass("dcTable")
dctable-class        Class "dctable"

Description
Stands for the 'dctable' S3 class from dclone package.

Objects from the Class
A virtual Class: No objects may be created from it.

Extends
Class "dcTable", directly.

Methods
No methods defined with class "dctable" in the signature.

Author(s)
Peter Solymos

See Also
dctable

Examples
showClass("dctable")

diagnostics        Diagnostic functions set as generic

Description
Diagnostic functions set as generic.

Usage
gelman.diag(x, ...)
geweke.diag(x, ...)
heidel.diag(x, ...)
raftery.diag(x, ...)
gelman.plot(x, ...)
**Arguments**

- `x`: MCMC objects.
- `...`: Other arguments.

**Details**

Diagnostic functions from the `coda` package are defined as generics for extensibility.

**Value**

Diagnostics summaries, and plot.

**Author(s)**

Peter Solymos

**References**

See relevant help pages.

**See Also**

- `gelman.diag`
- `geweke.diag`
- `heidel.diag`
- `raftery.diag`
- `gelman.plot`

---

**Description**

BUGS/JAGS compendium

**Objects from the Class**

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

**Slots**

- `data`: Object of class "list", same as corresponding `jags.fit` or `bugs.fit` or argument.
- `model`: Object of class "dcModel", same as corresponding `jags.fit` or `bugs.fit` or argument.
- `params`: Object of class "dcParams", same as corresponding `jags.fit` or `bugs.fit` or argument.
- `inits`: Object of class "dcInits", same as corresponding `jags.fit` or `bugs.fit` or argument.
- `flavour`: Object of class "character", same as corresponding `dc.fit` argument, default is "jags".
  
  It can also be "winbugs", "openbugs", or "brugs" referring to the argument of `bugs.fit`, in which case `flavour` will be treated as "bugs".
makeDcFit

Methods

show signature(object = "gsFit"): ...

Author(s)

Peter Solymos

See Also

jags.fit, bugs.fit, makeGsFit

Examples

showClass("gsFit")

Description

Creator functions for data types used in the dcmle package.

Usage

makeGsFit(data, model, params = NULL, inits = NULL, flavour)

makeDcFit(data, model, params=NULL, inits = NULL,
  multiply = NULL, unchanged = NULL, update = NULL,
  updatefun = NULL, initsfun = NULL, flavour)

Arguments

data usually a named list with data.
model BUGS model (function, character vector or a custommodel object). The argument is coerced into a custommodel object.
params optional, character vector for model parameters to monitor.
inits initial values (NULL, list or function).
multiply optional, argument passed to dc.fit.
unchanged optional, argument passed to dc.fit.
update optional, argument passed to dc.fit.
updatefun optional, argument passed to dc.fit.
initsfun optional, argument passed to dc.fit.
flavour optional, argument passed to dc.fit.
Details

'gsFit' (after BU*GS*/JA*GS*) is a basic object class representing requirements for the Bayesian MCMC model fitting. The 'dcFit' object class extends 'gsFit' by additional slots that are used to fine tune how data cloning is done during fitting process. Both 'gsFit' and 'dcFit' represent prerequisites for model fitting, but do not containing any fitted parts. Creator functions makeGsFit and makeDcFit are available for these classes. See dcmle-package help page for usage of creator functions.

The default flavour is stored in getOption("dcmle.flavour") with value "jags". It can be changed as options("dcmle.flavour"="bugs") if required.

Value

makeGsFit returns a 'gsFit' object (gsFit-class).
makeDcFit returns a 'dcFit' object (dcFit-class).

Author(s)

Peter Solymos <solymos@ualberta.ca>

See Also

gsFit-class, dcFit-class, dcmle

Examples

showClass("gsFit")
new("gsFit")
showClass("dcFit")
new("dcFit")
Author(s)

Peter Solymos

See Also

mcmc

Examples

showClass("mcmc")

### mcmc.list-class

Class "mcmc.list"

**Description**

Stands for the 'mcmc.list' S3 class from coda package.

**Objects from the Class**

A virtual Class: No objects may be created from it.

**Extends**

Class "MCMClist", directly.

**Methods**

No methods defined with class "mcmc.list" in the signature.

Author(s)

Peter Solymos

See Also

mcmc

Examples

showClass("mcmc.list")
**mcmc.list.dc-class**

Description

Stands for the 'mcmc.list.dc' S3 class from dclone package.

Objects from the Class

A virtual Class: No objects may be created from it.

Extends

Class "MCMCList", directly.

Methods

No methods defined with class "mcmc.list.dc" in the signature.

Author(s)

Peter Solymos

See Also

mcmc.list.dc

Examples

showClass("mcmc.list.dc")

**MCMCList-class**

Description

Virtual class for S3 mcmc.list object from codata package.

Objects from the Class

A virtual Class: No objects may be created from it.
Methods

acfplot signature(x = "MCMClist"): ...
autocorr.diag signature(mcmc.obj = "MCMClist"): ...
chanames signature(x = "MCMClist"): ...
chisq.diag signature(x = "MCMClist"): ...
coerce signature(from = "codaMCMC", to = "MCMClist"): ...
coerce signature(from = "dcCodaMCMC", to = "MCMClist"): ...
coerce signature(from = "dcmle", to = "MCMClist"): ...
coerce signature(from = "MCMClist", to = "codaMCMC"): ...
coerce signature(from = "MCMClist", to = "dcCodaMCMC"): ...
coerce signature(from = "MCMClist", to = "dcmle"): ...
confint signature(object = "MCMClist"): ...
crosscorr.plot signature(x = "MCMClist"): ...
crosscorr signature(x = "MCMClist"): ...
densityplot signature(x = "MCMClist"): ...
densplot signature(x = "MCMClist"): ...
frequency signature(x = "MCMClist"): ...
gelman.diag signature(x = "MCMClist"): ...
gelman.plot signature(x = "MCMClist"): ...
geweke.diag signature(x = "MCMClist"): ...
heidel.diag signature(x = "MCMClist"): ...
lambdamax.diag signature(x = "MCMClist"): ...
mcpar signature(x = "MCMClist"): ...
nchain signature(x = "MCMClist"): ...
niter signature(x = "MCMClist"): ...
nvar signature(x = "MCMClist"): ...
pairs signature(x = "MCMClist"): ...
plot signature(x = "MCMClist", y = "missing"): ...
qqmath signature(x = "MCMClist"): ...
quantile signature(x = "MCMClist"): ...
raftery.diag signature(x = "MCMClist"): ...
thin signature(x = "MCMClist"): ...
traceplot signature(x = "MCMClist"): ...
varnames signature(x = "MCMClist"): ...
xyplot signature(x = "MCMClist"): ...

Author(s)

Peter Solymos
See Also

mcmc.list

Examples

showClass("MCMClisit")

summary.codaMCMC-class

Class "summary.codaMCMC"

Description

Summary object.

Objects from the Class

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

Slots

settings: Object of class "integer", MCMC settings.
coef: Object of class "matrix", posterior statistics.
Summary object.

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

Slots

- `settings`: Object of class "integer", MCMC settings.
- `coef`: Object of class "matrix", coefficients (posterior means).
- `convergence`: Object of class "dcdiag", data cloning convergence diagnostics.

Extends

Class "summary.codaMCMC", directly.

Methods

- `show` signature(object = "summary.codaMCMC"): ...

Author(s)

Peter Solymos

See Also

`mcmc.list`, `jags.fit, dcdiag`
**summary.dcmle-class**

**Examples**

```r
showClass("summary.dcCodaMCMC")
```

**summary.dcmle-class  Class "summary.dcmle"**

**Description**

Summary object.

**Objects from the Class**

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

**Slots**

- **title**: Object of class "character", title to print, optional.
- **call**: Object of class "language", the call.
- **settings**: Object of class "integer", MCMC settings.
- **coef**: Object of class "matrix", coefficients (posterior means).
- **convergence**: Object of class "dcDiag", data cloning convergence diagnostics.

**Extends**

Class "summary.dcCodaMCMC", directly. Class "summary.codaMCMC", by class "summary.dcCodaMCMC", distance 2.

**Methods**

- **show** signature(object = "summary.dcmle"): ...

**Author(s)**

Peter Solymos

**See Also**

- `jags.fit`, `dcdiag`, `dcmle`

**Examples**

```r
showClass("summary.dcmle")
```
Index

*Topic classes
  codaMCMC-class, 5
  custommodel-class, 8
dcArgs-class, 8
dcCodaMCMC-class, 9
dcDiag-class, 10
dcDiag-class, 11
dcFit-class, 12
dcFunction-class, 13
dcInits-class, 13
dcModel-class, 18
dcParams-class, 19
dcTable-class, 19
dctable-class, 20
gsFit-class, 21
makeDeFit, 22
mcmc-class, 23
mcmc.list-class, 24
mcmc.list.dc-class, 25
MCMClist-class, 25
nclones-class, 27
summary.codaMCMC-class, 27
summary.dcCodaMCMC-class, 28
summary.dcmle-class, 29

*Topic htest
dcmle, 14

*Topic models
dcmle, 14

*Topic package
dcmle-package, 2

*Topic utils
  chanames, 4
crosscorr.plot, 7
diagnostics, 20
makeDeFit, 22
[,codaMCMC, ANY, ANY, ANY-method
  (dcCodaMCMC-class), 9
[,dCodaMCMC-method (dcCodaMCMC-class), 9
[,dcmle, ANY, ANY, ANY-method
  (dCodaMCMC-class), 9
[,dcmle-method (dcml-class), 16
[[,codaMCMC-method (codaMCMC-class), 5
[[,dCodaMCMC-method
  (dCodaMCMC-class), 9
[[,dcmle-method (dcmle-class), 16
acfplot, codaMCMC-method
  (codaMCMC-class), 5
acfplot, dcmle-method (dcmle-class), 16
acfplot, MCMClist-method
  (MCMClist-class), 25
as.array, codaMCMC-method
  (codaMCMC-class), 5
as.array, dcmle-method (dcmle-class), 16
as.matrix, codaMCMC-method
  (codaMCMC-class), 5
as.matrix, dcmle-method (dcmle-class), 16
as.mcmc.list, codaMCMC-method
  (codaMCMC-class), 5
as.mcmc.list, dcmle-method
  (dcmle-class), 16
autocorr.diag, codaMCMC-method
  (codaMCMC-class), 5
autocorr.diag, dcmle-method
  (dcmle-class), 16
autocorr.diag, MCMClist-method
  (MCMClist-class), 25
bugs.fit, 12, 14, 15, 21, 22
chanames, 4, 4
chanames, codaMCMC-method
  (codaMCMC-class), 5
chanames, dcmle-method (dcmle-class), 16
INDEX

chanames, MCMClist-method (MCMClist-class), 25
chisq.diag, codaMCMC-method (codaMCMC-class), 5
chisq.diag, dcmle-method (dcmle-class), 16
chisq.diag, MCMClist-method (MCMClist-class), 25
codaMCMC, 3, 9
codaMCMC-class, 5
coef, codaMCMC-method (codaMCMC-class), 5
coef, dcmle-method (dcmle-class), 16
coe, codaMCMC, dcmle-method (dcmle-class), 5
coe, codaMCMC, MCMClist-method (MCMClist-class), 5
coe, dcCodaMCMC, dcmle-method (dcmle-class), 9
coe, dcCodaMCMC, MCMClist-method (dcmle-class), 9
coe, dcmle, codaMCMC-method (dcmle-class), 16
coe, dcmle, dcCodaMCMC-method (dcmle-class), 16
coe, dcmle, MCMClist-method (dcmle-class), 16
coe, MCMClist, codaMCMC-method (codaMCMC-class), 5
coe, MCMClist, dcCodaMCMC-method (dcmle-class), 9
coe, MCMClist, dcmle-method (dcmle-class), 16
confint, codaMCMC-method (codaMCMC-class), 5
confint, dcCodaMCMC-method (dcmle-class), 9
confint, dcmle-method (dcmle-class), 16
confint, MCMClist-method (MCMClist-class), 25
crosscorr, codaMCMC-method (codaMCMC-class), 5
crosscorr, dcmle-method (dcmle-class), 16
crosscorr, MCMClist-method (MCMClist-class), 25
crosscorr.plot, 7, 7
crosscorr.plot, codaMCMC-method (codaMCMC-class), 5
crosscorr.plot, dcmle-method (dcmle-class), 16
custommodel, 8, 22
custommodel-class, 8
dc.fit, 12, 14, 15, 21, 22
dc.parfit, 14, 15
dcArgs-class, 8
dcCodaMCMC, 3
dcCodaMCMC-class, 9
dcDiag, 11
dcDiag, 9, 11, 28, 29
dcDiag, dcmle-method (dcmle-class), 16
dcDiag-class, 10
dcDiag-class, 11
dcFit, 2, 14
dcFit-class, 12, 23
dcFunction-class, 13
dcInits-class, 13
dcmle, 2, 3, 14, 15, 16, 18, 23, 29
dcmle-class, 16
dcmle-package, 2
dcModel, 8
dcModel-class, 18
dcParams-class, 19
dcsd, codaMCMC-method (codaMCMC-class), 5
dcsd, dcmle-method (dcmle-class), 16
dcTable, 20
dcTable, 9, 20
dcTable, codaMCMC-method (codaMCMC-class), 5
dcTable, dcCodaMCMC-method (dcCodaMCMC-class), 9
dcTable, dcmle-method (dcmle-class), 16
dcTable-class, 19
dcTable-class, 20
densityplot, codaMCMC-method (codaMCMC-class), 5
densityplot, dcmle-method (dcmle-class), 16
densityplot, MCMClist-method (MCMClist-class), 25
densplot, codaMCMC-method (codaMCMC-class), 5
<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>densplot</td>
<td>dcmle-method</td>
<td>16</td>
</tr>
<tr>
<td>densplot</td>
<td>MCMClist-method</td>
<td>25</td>
</tr>
<tr>
<td>diagnostics</td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>end</td>
<td>codaMCMC-method</td>
<td>5</td>
</tr>
<tr>
<td>end</td>
<td>dcmle-method</td>
<td>16</td>
</tr>
<tr>
<td>frequency</td>
<td>codaMCMC-method</td>
<td>5</td>
</tr>
<tr>
<td>frequency</td>
<td>dcmle-method</td>
<td>16</td>
</tr>
<tr>
<td>frequency</td>
<td>MCMClist-method</td>
<td>25</td>
</tr>
<tr>
<td>gelman.diag</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>gelman.diag</td>
<td>diagnostics</td>
<td>20</td>
</tr>
<tr>
<td>gelman.diag</td>
<td>codaMCMC-method</td>
<td>5</td>
</tr>
<tr>
<td>gelman.diag</td>
<td>dcmle-method</td>
<td>16</td>
</tr>
<tr>
<td>gelman.diag</td>
<td>MCMClist-method</td>
<td>25</td>
</tr>
<tr>
<td>gelman.plot</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>gelman.plot</td>
<td>diagnostics</td>
<td>20</td>
</tr>
<tr>
<td>gelman.plot</td>
<td>codaMCMC-method</td>
<td>5</td>
</tr>
<tr>
<td>gelman.plot</td>
<td>dcmle-method</td>
<td>16</td>
</tr>
<tr>
<td>gelman.plot</td>
<td>MCMClist-method</td>
<td>25</td>
</tr>
<tr>
<td>geweke.diag</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>geweke.diag</td>
<td>diagnostics</td>
<td>20</td>
</tr>
<tr>
<td>geweke.diag</td>
<td>codaMCMC-method</td>
<td>5</td>
</tr>
<tr>
<td>geweke.diag</td>
<td>dcmle-method</td>
<td>16</td>
</tr>
<tr>
<td>geweke.diag</td>
<td>MCMClist-method</td>
<td>25</td>
</tr>
<tr>
<td>gsFit</td>
<td></td>
<td>2, 12, 14</td>
</tr>
<tr>
<td>gsFit-class</td>
<td></td>
<td>21, 23</td>
</tr>
<tr>
<td>head</td>
<td>codaMCMC-method</td>
<td>5</td>
</tr>
<tr>
<td>head</td>
<td>dcmle-method</td>
<td>16</td>
</tr>
<tr>
<td>heidel.diag</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>heidel.diag</td>
<td>diagnostics</td>
<td>20</td>
</tr>
<tr>
<td>heidel.diag</td>
<td>codaMCMC-method</td>
<td>5</td>
</tr>
<tr>
<td>heidel.diag</td>
<td>dcmle-method</td>
<td>16</td>
</tr>
<tr>
<td>heidel.diag, MCMClist-method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heidel.diag</td>
<td>(MCMClist-class)</td>
<td></td>
</tr>
<tr>
<td>jags.fit</td>
<td></td>
<td>10, 14, 15, 21, 22, 25, 28, 29</td>
</tr>
<tr>
<td>jags.parfit</td>
<td></td>
<td>14, 15</td>
</tr>
<tr>
<td>lambdamax.diag</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lambdamax.diag, codaMCMC-method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lambdamax.diag, dcmle-method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lambdamax.diag, MCMClist-method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>makeDcFit</td>
<td></td>
<td>3, 12, 15, 22</td>
</tr>
<tr>
<td>makeGsFit</td>
<td></td>
<td>3, 15, 22</td>
</tr>
<tr>
<td>makeGsFit</td>
<td>(makeDcFit)</td>
<td>22</td>
</tr>
<tr>
<td>mcmc</td>
<td></td>
<td>24</td>
</tr>
<tr>
<td>mcmc-class</td>
<td></td>
<td>23</td>
</tr>
<tr>
<td>mcmc.list</td>
<td></td>
<td>7, 24, 25, 27, 28</td>
</tr>
<tr>
<td>mcmc.list-class</td>
<td></td>
<td>24</td>
</tr>
<tr>
<td>mcmc.list, dc-class</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>MCMClist</td>
<td></td>
<td>23–25</td>
</tr>
<tr>
<td>MCMClist-class</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>mcpars, codaMCMC-method</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>mcpars, dcmle-method</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>mcpars, MCMClist-method</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>mcpars, MCMClist-method (MCMClist-class)</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>nchain, codaMCMC-method</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>nchain, dcmle-method</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>nchain, MCMClist-method</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>nclones, codaMCMC-method</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>nclones, dcDodaMCMC-method</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>nclones, dcmle-method</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>nclones-class</td>
<td></td>
<td>27</td>
</tr>
<tr>
<td>niter, codaMCMC-method</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>niter, dcmle-method</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>niter, MCMClist-method</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>nvar, codaMCMC-method</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>nvar, dcmle-method</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>nvar, MCMClist-method</td>
<td></td>
<td>25</td>
</tr>
</tbody>
</table>
pairs, codaMCMC-method (codaMCMC-class), 5
pairs, dcMle-method (dcMle-class), 16
pairs, MCMCList-method (MCMCList-class), 25
plot, codaMCMC, missing-method (codaMCMC-class), 5
plot, dcMle, missing-method (dcMle-class), 16
plot, MCMCList, missing-method (MCMCList-class), 25

qqmath, codaMCMC-method (codaMCMC-class), 5
qqmath, dcMle-method (dcMle-class), 16
qqmath, MCMCList-method (MCMCList-class), 25
quantile, codaMCMC-method (codaMCMC-class), 5
quantile, dcMle-method (dcMle-class), 16
quantile, MCMCList-method (MCMCList-class), 25

raftery.diag, 21
raftery.diag (diagnostics), 20
raftery.diag, codaMCMC-method (codaMCMC-class), 5
raftery.diag, dcMle-method (dcMle-class), 16
raftery.diag, MCMCList-method (MCMCList-class), 25

show, codaMCMC-method (codaMCMC-class), 5
show, dcFit-method (dcFit-class), 12
show, dcMle-method (dcMle-class), 16
show, gcFit-method (gcFit-class), 21
show, summary.codaMCMC-method (summary.codaMCMC-class), 27
show, summary.dcCodaMCMC-method (summary.dcCodaMCMC-class), 28
show, summary.dcMCMC-method (summary.dcMCMC-class), 29
stack, codaMCMC-method (codaMCMC-class), 5
stack, dcMle-method (dcMle-class), 16
start, codaMCMC-method (codaMCMC-class), 5
start, dcMle-method (dcMle-class), 16

str, dcCodaMCMC-method (dcCodaMCMC-class), 9
str, dcMle-method (dcMle-class), 16
summary, codaMCMC-method (codaMCMC-class), 5
summary, dcCodaMCMC-method (dcCodaMCMC-class), 9
summary, dcMle-method (dcMle-class), 16
summary, codaMCMC, 28, 29
summary, codaMCMC-class, 27
summary, dcCodaMCMC, 29
summary, dcMle-class, 29
tail, codaMCMC-method (codaMCMC-class), 5
tail, dcMle-method (dcMle-class), 16
thin, codaMCMC-method (codaMCMC-class), 5
thin, dcMle-method (dcMle-class), 16
thin, MCMCList-method (MCMCList-class), 25
time, codaMCMC-method (codaMCMC-class), 5
time, dcMle-method (dcMle-class), 16
traceplot, codaMCMC-method (codaMCMC-class), 5
traceplot, dcMle-method (dcMle-class), 16
traceplot, MCMCList-method (MCMCList-class), 25
update, dcMle-method (dcMle-class), 16

varnames, 4
varnames (chanames), 4
varnames, codaMCMC-method (codaMCMC-class), 5
varnames, dcMle-method (dcMle-class), 16
varnames, MCMCList-method (MCMCList-class), 25
vcov, codaMCMC-method (codaMCMC-class), 5
vcov, dcMle-method (dcMle-class), 16
window, codaMCMC-method (codaMCMC-class), 5
window, dcMle-method (dcMle-class), 16
xyplot, codaMCMC-method (codaMCMC-class), 5
xyplot, dcMle-method (dcMle-class), 16
xyplot, MCMCList-method (MCMCList-class), 25