applyConversions

**Description**

Create transformed parameters from original parameters in MCMC output

**Usage**

```r
applyConversions(samples, conversions)
```

**Arguments**

- `samples` One of: an `MCMCresult` object; a named list of `MCMCresult` objects (such as returned by `compareMCMCs`); a matrix of MCMC samples (such as the `samples` element of an `MCMCresult` object); or a named list of such matrices. In the first two cases, conversions will be done in place (as a "side effect" modifying the arguments) because `MCMCresult` objects are R6 objects and are thus passed by reference.

- `conversions` One of: a list of conversion specifications (see below); a named list of conversion specifications, with names matching those of a list provided for `samples`.

**Details**

A conversion specification is a named list. For each element:

- its name will be the name of a new column appended to a `samples` matrix.
- its value should be a character string that can be parsed as code to calculate elements of the new column. It can use existing column names in `samples`. Calculations will be done row-wise. Column names are often something like "\texttt{beta[2]}". To have this used as a name, enclose it in backticks, e.g. "\texttt{\`beta[2]\}". For example, an entry could be \texttt{log\_beta2 = log(`beta[2]`)}. A list value of `NULL` will remove the named column.
The conversion specification list will be processed in order. This allows creating new columns and removing old ones in a sensible order.

If both conversions and samples are named lists, they will be matched: the conversions element (itself a list of conversion specifications) used on a samples element will have the same name. If there is no conversions element for a given samples element, that samples element will be included in the returned list without any conversions.

Value

An object of the same type as samples after application of conversions.

---

**combineMetrics**

*Combine all metrics from a list of MCMCResult objects.*

**Description**

This is useful for seeing results from multiple MCMC engines compactly.

**Usage**

```r
combineMetrics(results, include_times = FALSE)
```

**Arguments**

- `results` a list of MCMCResult objects
- `include_times` if TRUE, attempt to include timing elements in the combination.

**Value**

A list with elements byParameter and byMCMC. Each element combines the corresponding elements for each MCMCResult object in the results argument.

if `include_times` is TRUE, an element times will also be in the returned list.

See Also

- `modifyMetrics`
compareMCMCs

run a set of MCMCs for performance comparison

Description

run one or more MCMC engines for one model specification, with timing and performance metrics calculated.

Usage

```r
compareMCMCs(
  modelInfo = list(),
  MCMCcontrol = list(niter = 10000, thin = 1, burnin = 2000),
  MCMCs = "nimble",
  monitors = character(),
  nimbleMCMCdefs = list(),
  externalMCMCinfo = list(),
  metrics = c("mean", "median", "sd", "CI95_low", "CI95_upp", "efficiency_coda"),
  metricOptions = list(),
  conversions = list(),
  seed = NULL,
  needRmodel,
  verbose = TRUE,
  sessionInfo = TRUE
)
```

Arguments

- **modelInfo**: A list of nimble model-specification information (which may be relevant for JAGS, WinBUGS and/or OpenBUGS as well) and/or a nimble model itself. To provide information for a different MCMC engine, see argument `externalMCMCinfo`. Named elements in `modelInfo` can include `code` (model code as returned from `nimbleCode`), `data` (a list with data), `constants` (a list with data and/or constants), `inits` (a list of initial values), and/or `model` (an object returned from `nimbleModel`). If `model` is not provided, and if an R model will be needed, then `nimbleModel` will be called to create one using `code`, `data`, and/or `inits`. See `nimbleModel` in package nimble for information on these arguments. For JAGS, WinBUGS and OpenBUGS, many models can be run from the same specification since they use nearly the same model language. If `model` is provided, the other elements will not be needed if only nimble MCMCs are used but will be needed if JAGS, WinBUGS or OpenBUGS will be used.

- **MCMCcontrol**: A list with fields `niter` (number of iterations), `thin` (thinning interval), and `burnin` (number of iterations to discard from the beginning of the MCMC sample).

- **MCMCs**: A character vector of MCMC cases to run. This can include "nimble" (default nimble samplers), "jags", "stan", one of several nimble special cases (see...
compareMCMCs
details below), custom nimble sampler configurations provided via argument
nimbleMCMCdefs, and external MCMC engines registered via registerMCMCengine.
See builtin_MCMCs for information on "jags" and "stan". Support for Open-
BUGS and WinBUGS is pending.

monitors A character vector of variable names to monitor (record in MCMC output). If
missing, this will be determined from the nimble model as all top-level parameter
names (e.g. hyper-parameters).
nimbleMCMCdefs A list of information for custom sampler configurations in nimble. See package
vignette for details.

externalMCMCinfo A list of arbitrary information for external MCMC engines, named by engine
names. If there is an external MCMC engine named "myMCMC", then a list element
myMCMC of externalMCMCinfo will be passed to the engine as its MCMCinfo
argument.

metrics Either a character vector of registered metric names to apply to each sample, or
a list of elements with either metric names or metric functions to apply to each
sample. See addMetrics for more information. A useful set of default metrics
is provided.

metricOptions Optional named list of individual metric options passed as the third argument
("options") of addMetrics when MCMC metrics are calculated.

conversions List of parameter conversion (transformation) specifications, useful when different
MCMCs use different parameterizations.

seed An (arbitrary) numeric value passed to set.seed to set the random-number gen-
erator seed before calling each MCMC engine. If NULL, no seed is set. To
obtain identical results from one call of compareMCMCs to the next, use identical
seed values.

needRmodel If TRUE, a nimble model object should definitely be created (if necessary, or
obtained from modelInfo$model if provided) and used, for example to deter-
mine variable names. If missing, needRmodel will be set TRUE if MCMCs includes
"nimble", "jags", "openbugs", or "winbugs".

verbose If TRUE, more verbose output may be generated.

sessionInfo If TRUE, record the results of sessionInfo(), run before calling each MCMC,
with each MCMC result.

Details

The special cases provided for the MCMCs argument include:

• "nimble_noConj": use adaptive random-walk Metropolis-Hastings (ARWMH) samplers in
place of Gibbs (conjugate) samplers.
• "nimble_RW": use all adaptive random-walk Metropolis-Hastings samplers.
• "nimble_slice": use all slice samplers.

See package vignette for more details and examples.
Description

Create html output with comparisons of MCMC results

Usage

make_MCMC_comparison_pages(
  comparisonResults,
  dir = tempdir(),
  pageComponents,
  modelName = "model",
  control,
  plot = TRUE
)

Arguments

- comparisonResults: An list of MCMCresult objects such as returned by `compareMCMCs`.
- dir: A directory in which to place the html file and any figure files used in it. This defaults to `tempdir()` (which will be erased when the R session is closed). Use `dir = getwd()` to use current working directory.
- pageComponents: A list whose names are registered page components and values are TRUE (to include a component) or FALSE (to omit a component). Components can also be omitted by leaving them out of the list.
- modelName: A name to be used for the model in generated output.
- control: A named list of control parameters.
- plot: TRUE to generate results, FALSE not to do so. Use of FALSE is useful if one wants to use the returned object (including plottable components) in one’s own way.

Details

See package vignette for information about page components, including about default page components and how to write and register new page components.

To see built-in page components and their options, use `as.list(getPageComponents())`. 
Value

A list of objects returned from each page component plugin. For figures, these contain a plottable object such as a ggplot object. For text, these contain information for text output such as an xtable object.

Description

These functions are normally called from compareMCMCs, which passes its arguments or elements extracted from its arguments to these functions.

Usage

MCMCdef_dummy(MCMCinfo, MCMCcontrol, monitorInfo, modelInfo)
MCMCdef_jags(MCMCinfo, MCMCcontrol, monitorInfo, modelInfo)
MCMCdef_stan(MCMCinfo, MCMCcontrol, monitorInfo, modelInfo)

Arguments

MCMCinfo The named element of externalMCMCinfo argument to compareMCMCs that matches a particular MCMC. ("External" refers to any MCMC that is not internal to nimble.)
MCMCcontrol The MCMCcontrol argument to compareMCMCs, with the seed argument added as a list element if it was provided.
monitorInfo A list with elements monitors and monitorVars, providing two formats of information on model parameters for which MCMC output should be recorded.
modelInfo The modelInfo argument to compareMCMCs

Details

These functions are called internally from compareMCMCs. Each one runs an MCMC engine. Functions to interface to other MCMC engines can be registered via registerMCMCengine.

MCMCs in nimble are run from runNIMBLE. This uses a different system because there may be multiple nimble MCMC configurations for one model.
MCMCdef_dummy does not run a real MCMC. It provides a quick way to generate MCMC-formatted output for testing other parts of this package.
MCMCdef_jags runs JAGS via package rjags. It uses model information from modelInfo. It does not use MCMCinfo.
MCMCdef_stan runs Stan via package rstan. It does not use modelInfo. It accepts the following elements of the MCMCinfo list:
• file: file argument to stan_model function in rstan. This can alternatively be provided via stan_model_args$file.
• data: data argument to sampling function in rstan. This can alternatively be provided via sampling_args$data.
• inits: inits argument to sampling function in rstan. This can alternatively be provided via sampling_args$inits.
• stan_model_args: list of arguments to stan_model. Note that this can provide the Stan model in the model_code element (as a character string) or in the file element (an alternative way to provide the file name).
• sampling_args: list of arguments to sampling.

The elements file, data, and inits take precedence over corresponding entries in stan_model_args or sampling_args.

If elements warmup, iter, and/or thin are provided in sampling_args, those take precedence over corresponding values in the MCMCcontrol argument to compareMCMCs. Otherwise iter is set to MCMCcontrol$niter and warmup is set to MCMCcontrol$niter/2. Only one chain will be run.

Total sampling time for Stan is recorded via system.call(sampling(...)). This is similar to how time is recorded for other MCMCs. The warmup time (called "burnin" in compareMCMCs for consistency across different MCMCs) is obtained from rstan function get_elapsed_time. The post-burnin time is the total sampling time minus the burnin time.

---

**MCMCresult**  
*R6 class to hold MCMC samples, timing results, and metrics*

**Description**

R6 class to hold MCMC samples, timing results, and metrics

R6 class to hold MCMC samples, timing results, and metrics

**Public fields**

- **MCMC** Optional name for the MCMC method.
- **samples** Matrix of MCMC samples. Rows are for MCMC iterations. Columns are for parameters. Columns must be named.
- **times** A list of times including elements for setup, burnin, postburnin (sampling for recorded samples), and sampling (normally burnin + postburnin). Each list element should be a single numeric value.
- **metrics** A list of MCMC performance metrics such as effective sample size (ESS), efficiency, mean, median, and credible interval boundaries. metrics' is organized as a list with three elements: byMCMC, byParameter, and other' (currently unused).
  - byMCMC is for metrics with one number for an entire MCMC sample (as opposed to one number for each parameter). byMCMC is a data frame with one row and columns for MCMC name each metric. These would be metrics where there is a single
byParameter is for metrics with one number for each parameter in each MCMC sample. byParameter is a data.frame with one row for each MCMC-x-parameter combination and columns for MCMC method, parameter name, and each metric. There will only be one MCMC method name (all entries in the MCMC column will be the same). The MCMC columns in byMCMC and byParameter are useful for combining metrics from a list of MCMCresult objects, such as done by combineMetrics, and for retaining MCMC method labels if these data.frames are copied and used outside of an MCMCresult object.

other is simply an arbitrary list. This allows arbitrarily structured metrics to be saved.

Elements of metrics are normally populated by addMetrics or compareMCMCs (which calls addMetrics).

sessionInfo Result of running sessionInfo() prior to calling an MCMC engine, if requested.

### Methods

#### Public methods:
- MCMCresult$new()
- MCMCresult)setSamples()
- MCMCresult$rename()
- MCMCresult$initializeMetrics()
- MCMCresult$clearMetrics()
- MCMCresult$addMetricResult()
- MCMCresult$clone()

**Method new():** Create a new MCMCresult object.

**Usage:**
MCMCresult$new(...)

**Arguments:**
... Arbitrary initialization. If a matrix is passed, it will be used to initialize samples and the metrics elements. If a list with a matrix element named samples is passed, this element will be used as if the matrix itself was passed. Any other named elements of a list that correspond to fields of an MCMCresult object will be initialized from them.

**Method setSamples():** Populate the samples and initialize the metrics

**Usage:**
MCMCresult)setSamples(samples)

**Arguments:**
samples A data.frame with MCMC output.

**Returns:** NULL

**Method rename():** Change the MCMC method name from oldName to newName

**Usage:**
MCMCresult$rename(newName, oldName)

**Arguments:**
newName New name for MCMC method in metrics
oldName Old name for MCMC method in metrics

Details: This change the MCMC field and the corresponding columns of metrics$byParameter and metrics$byMCMC.
If oldName is not the MCMC method name, this function does nothing.

Returns: NULL

Method initializeMetrics(): Initialize metrics if necessary

Usage:
MCMCresult$initializeMetrics(silent = FALSE)

Arguments:
silent logical indicating whether to emit warnings

Details: This function does nothing if metrics are already initialized. It does not clear metrics.
See clearMetrics for information on how metrics are initialized.

Returns: logical indicating whether metrics is well-formed or not.

Method clearMetrics(): Clear (reset) byParameter and/or byMCMC metrics

Usage:
MCMCresult$clearMetrics(byParameter = TRUE, byMCMC = TRUE)

Arguments:
byParameter logical indicating whether to clear byParameter metrics
byMCMC logical indicating whether to clear byMCMC metrics

Details: byParameter metrics are initialized to a data.frame with columns for MCMC (all the same entry, the MCMC field) and Parameter (taken from column names of the samples). byMCMC metrics are initialized to a data.frame with a column for MCMC.

Method addMetricResult(): Add one set of metric results

Usage:
MCMCresult$addMetricResult(metricResult)

Arguments:
metricResult A list with possible elements byParameter, byMCMC, and other. These are typically returned from a metric function called via addMetric. Each is combined with previous metrics already in the corresponding elements of metrics.

Method clone(): The objects of this class are cloneable with this method.

Usage:
MCMCresult$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

See Also
renameMCMC to change the name of an MCMC method throughout the structure of a list of MCMCresult objects.
Built-in metrics for MCMC result objects

Description
These functions are normally called via compareMCMCs or addMetric.

Usage
MCMCmetric_mean(result, ...)
MCMCmetric_median(result, ...)
MCMCmetric_sd(result, ...)
MCMCmetric_CI95(result, ...)
MCMCmetric_CI95low(result, ...)
MCMCmetric_CI95upp(result, ...)
MCMCmetric_ESS(result, options = NULL)
MCMCmetric_efficiency(result, options = NULL)

Arguments
result An MCMCResult object, normally a list element returned by compareMCMCs
... Possible additional arguments to metric functions.
options A (metric-specific) list of named control options accepted by some metrics.

Details
A metric is a summary of MCMC output. The summary may include results for each parameter, for each MCMC sample (across all parameters), and/or by arbitrary list. The last option is not used by any built-in metrics.

The built-in metrics include:
• mean: mean for each parameter
• median: median for each parameter
• sd: standard deviation for each parameter
• CI95: both ends of 95% credible interval, a combination of CI95low and CI95upp
• CI95low: lower end of 95% credible interval
• CI95upp: upper end of 95% credible interval
modifyMetrics

Manipulate metrics in one or more MCMCResult object(s)

Description
Clear metrics or add metrics to MCMC results.

Usage

clearMetrics(results, byParameter = TRUE, byMCMC = TRUE)

addMetrics(
results,
metrics = c("mean", "median", "sd", "CI95_low", "CI95_upp", "ESS", "efficiency"),
options = list()
)

Arguments

results an MCMCResult object or list of MCMCResult objects.
byParameter TRUE or FALSE: whether to clear byParameter metrics
byMCMC TRUE or FALSE: whether to clear byMCMC metrics
metrics character vector of metric names to add. See metrics.
options named list of options. When calling a metric function (e.g. mean), if there is a named element with that name (e.g. "mean"), it will be passed as the second argument to the metric function.
These functions provide ways to manipulate the collection of metrics inside one or more `MCMCresult` objects.

The `MCMCresult` class is fairly simple. One can also modify contents of an `MCMCresult` object using class methods or direct manipulation of contents.

Metrics are organized as "byParameter", when there is one result for each parameter (column) of MCMC output, and "byMCMC", when there is one result for an entire MCMC sample (across all parameters).

`clearMetrics` clears all metrics by parameter, by MCMC, or both.

`addMetrics` populates a set of metrics. See package vignette for more information.

**See Also**

`combineMetrics`

---

**Description**

Register, unregister and access page components used by `make_MCMC_comparison_pages`

**Usage**

```r
registerPageComponents(pageComponents)
unregisterPageComponents(name)
getPageComponents()
```

**Arguments**

- `pageComponents`: A named list of new page components to register
- `name`: Character name of a page component to unregister

**Details**

A page component is an element that can be included in an MCMC comparison page by naming it in the `pageComponents` argument to `make_MCMC_comparison_pages`. See package vignette for explanation page components.

**See Also**

`make_MCMC_comparison_pages`
registerMCMCengine

Register an MCMC function for use by compareMCMCs

Description

Register an MCMC function for use by compareMCMCs

Usage

registerMCMCengine(name, fun)

Arguments

name
The name by which the MCMC function (or "engine") is identified in the MCMCs argument to compareMCMCs.

fun
The function that runs and times an MCMC.

Details

See package vignette for information about the arguments that will be passed to fun from compareMCMCs and the MCMCresult object that should be returned by fun.

For more information, see builtin_MCMCs.

MCMCs from nimble are run in a different way, since there can be multiple MCMCs for the same nimble model. These are run by runNIMBLE, which is not exported.

registerMetrics

Register, unregister, or access registered MCMC metric functions for use by compareMCMCs or addMetrics

Description

Register, unregister, or access registered MCMC metric functions for use by compareMCMCs or addMetrics

Usage

registerMetrics(metrics)

unregisterMetric(name)

getMetrics()

Arguments

metrics
A named list of new metric functions to register

name
Character name of a metric function to unregister
renameMCMC

Details
These functions are called for their "side effects" of modifying the list metric functions for MCMC results that will be recognized by name from the compareMCMCs or addMetrics functions. Those functions take a metrics argument that can be a character vector or a list. Names in the character vector will be looked up from the registered metric functions.

registerMetrics takes a named list and adds its elements to the list of recognized metrics with the corresponding names.
unregisterMetric removes one metric from the list at a time.
getMetrics returns the list of registered metrics.

Value
registerMetrics and getMetrics return the environment of registered metrics.
unregisterMetric returns the result (which should be NULL) of a call to rm that attempts to remove a metric.

renameMCMC Rename an MCMC method throughout a list of MCMCresult objects

Description
This is useful because an MCMC method name appears in multiple places

Usage
renameMCMC(MCMCresult, newName, oldName)

Arguments
MCMCresult One or a named list of MCMCresult objects, such as returned by compareMCMCs.
newName A new (replacement) name for one of the MCMC method names
oldName An old (existing) name for one of the MCMC method names

Details
This replaces the MCMC label oldName with newName anywhere they appear in the MCMCresult list. This includes various places in the metrics elements of the MCMCresult objects.
If oldName is omitted, MCMCresult must be a single MCMCresult object, in which the existing MCMC method name will be replaced by newName. Hence oldName is only necessary if MCMCresult is a list of MCMCresult objects.
addMetrics, 5
addMetrics (modifyMetrics), 12
applyConversions, 2

builtin_MCMCs, 5, 14
builtin_MCMCs (MCMCdef_dummy), 7

clearMetrics (modifyMetrics), 12
combineMetrics, 3, 9, 13
compareMCMCs, 2, 4, 6, 7, 11, 14, 15

getMetrics (registerMetrics), 14
getPageComponents (pageComponents), 13

make_MCMC_comparison_pages, 6, 13
MCMCdef_dummy, 7
MCMCdef_jags (MCMCdef_dummy), 7
MCMCdef_stan (MCMCdef_dummy), 7
MCMCmetric_CI95 (metrics), 11
MCMCmetric_CI95low (metrics), 11
MCMCmetric_CI95upp (metrics), 11
MCMCmetric_efficiency (metrics), 11
MCMCmetric_ESS (metrics), 11
MCMCmetric_mean (metrics), 11
MCMCmetric_median (metrics), 11
MCMCmetric_sd (metrics), 11
MCMCresult, 2, 8, 13
metrics, 11, 12
modifyMetrics, 3, 12

pageComponents, 13

registerMCMCengine, 5, 7, 14
registerMetrics, 14
registerPageComponents
 (pageComponents), 13
renameMCMC, 10, 15

unregisterMetric (registerMetrics), 14
unregisterPageComponents
 (pageComponents), 13