Package ‘RBi’

May 4, 2016

Version 0.1-1

Date 2015-11-16

Title R interface to LibBi

Author Pierre E. Jacob <`pierre.jacob.work@gmail.com`>, Anthony Lee <`awllee@gmail.com`>, Sebastian Funk <`sebastian.funk@lshtm.ac.uk`>

Maintainer Pierre E. Jacob <`pierre.jacob.work@gmail.com`>

Depends methods

Imports reshape2,
ncdf4

LinkingTo Rcpp

Suggests

Description R interface to LibBi

License GPL (>= 3)

URL `https://github.com/libbi/RBi`

BugReports

LazyLoad no

RoxygenNote 5.0.1

R topics documented:

`RBi-package` *RBi - R interface for* `libbi`

Description

RBi is an interface to `libbi`, a library for Bayesian Inference

Details

The package includes a wrapper for the `libbi` script, allowing to launch the `libbi` command from within R. It also provides various utility functions to browse the output from `libbi`, for instance to plot the results.

The package will ultimately be made of various components.

- First there is a wrapper around `libbi` called `libbi`.
- Then there are funtions to manipulate the results of the `libbi` command, which are stored in NetCDF files. Those functions allow to extract variables of interest, and to plot them in various ways.

Author(s)

Pierre E. Jacob <`pierre.jacob.work@gmail.com`>, Sebastian Funk <`sebastian.funk@lshtm.ac.uk`>

References

`http://libbi.org/`

See Also

`libbi`

Examples

```
demo(PZ_generate_dataset)
demo(PZ_PMMH)
demo(PZ_SMC2)
demo(PZ_filtering)
```

`absolute_path` *Absolute Path*

Description

This function is used to convert relative file paths to absolute file paths without checking if the file exists as tools::file_as_absolute_path does

Usage

```
absolute_path(filename, dirname)
```

Arguments

`filename`	name of a file, absolute or relative to a folder
`dirname`	name of a folder where the file is supposed to be

`bi_dim_len` *NetCDF dimension length*

Description

This function returns the length of a dimension in a NetCDF file.

Usage

```
bi_dim_len(filename, dim)
```

Arguments

`filename`	path to a NetCDF file

Value

dimension length

`bi_dim_values` *NetCDF dimension values*

Description

This function returns the values of a dimension in a NetCDF file.

Usage

```
bi_dim_values(filename, dim)
```

Arguments

`filename`	path to a NetCDF file

Value

dimension values

`bi_file_ncdump` *NetCDF File Print*

Description

This function prints the content of a file using the ncdump command line

Usage

```
bi_file_ncdump(filename)
```

Arguments

`filename`	path to a NetCDF file

Value

None

`bi_file_summary` *NetCDF File Summary*

Description

This function prints a little summary of the content of a NetCDF file, as well as its creation time. You can then retrieve variables of interest using `bi_read`.

Usage

```
bi_file_summary(filename)
```

Arguments

`filename`	path to a NetCDF file

Value

None

`bi_generate_dataset` *Bi Generate Dataset*

Description

This is a wrapper around `libbi sample --target joint`, to generate synthetic dataset from a model. Parameters can be passed via the 'init' option (see `libbi_run`, otherwise they are generated from the prior specified in the model.

Usage

```
bi_generate_dataset(endtime, noutputs, ...)
```

Arguments

`endtime`	final time index, so that data is generated from time 0 to time "endtime".
`noutputs`	number of output points to be extracted from the hidden process; default is noutputs = endtime.
`...`	arguments to be passed to `libbi` (with run = TRUE), especially 'model'

Value

path to the output file.

`bi_init_file` *Create init files for LibBi, retained for backwards compatibility*

Description

Users should use `bi_write` instead

Usage

```
bi_init_file(...)
```

Arguments

`...`	parameters passed to `bi_write`

Value

whatever `bi_write` returns

See Also

`bi_write`

`bi_model` *Bi Model*

Description

`bi_model` creates a model object for `Rbi` from a libbi file. Once the instance is created, the model can either be fed to a `libbi` object.

Arguments

`filename`	is the file name of the model file

See Also

`bi_model_fix`, `bi_model_propose_prior`, `bi_model_insert_lines`, `bi_model_update_lines`, `bi_model_remove_lines`, `bi_model_write_model_file`,

Examples

```
bi_sir <- bi_model$new(filename = "sir.bi")
```

`bi_model_fix` *Fix noise term, state or parameter of a libbi model*

Description

Replaces all variables with fixed values as given in 'fixed'; note that this will not replace differential equations and lead to an error if applied to states that are changed inside an "ode" block

Arguments

`...`	values to be assigned to the (named) variables

Value

a bi model object of the new model

See Also

`bi_model`

Examples

```
model_file_name <- system.file(package="bi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ$fix(alpha = 0)
```

`bi_model_insert_lines` *Insert lines in a libbi model*

Description

Inserts one or more lines into a libbi model. If one of `before` or `after` is given, the line(s) will be inserted before or after a given line number, respectively. If neither is given, the line(s) will be added at the end.

Arguments

`before`	line number before which to insert line(s)
`after`	line number after which to insert line(s)
`lines`	vector or line(s)

Value

the updated bi model

See Also

`bi_model`

Examples

```
model_file_name <- system.file(package="bi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ$insert_lines(lines = "noise beta", after = 8)
```

---

bi_model_propose_prior *Propose from the prior in a libbi model*

Description

Generates a version of the model where the proposal blocks are replaced by the prior blocks. This is useful for exploration of the likelihood surface.

Value

a bi model object of the new model

See Also

bi_model

Examples

```
model_file_name <- system.file(package="bi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ$propose_prior()
```

---

bi_model_remove_lines *Remove line(s) in a libbi model*

Description

Removes one or more lines in a libbi model.

Arguments

num	line number(s) to remove

Value

the updated bi model

See Also

bi_model

Examples

```
model_file_name <- system.file(package="bi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ$remove_lines(2)
```

`bi_model_update_lines` *Update line(s) in a libbi model*

Description

Updates one or more lines in a libbi model.

Arguments

`num`	line number(s) to update
`lines`	vector of iline(s)

Value

the updated bi model

See Also

`bi_model`

Examples

```
model_file_name <- system.file(package="bi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ$update_lines(23, "alpha ~ normal(mu, sigma)")
```

`bi_model_write_model_file` *Writes a bi model to a file.*

Description

Writes a bi model to a file given by `filename`. The extension '.bi' will be added if necessary.

Arguments

`filename`	name of the file to be written

Value

the return value of the `writeLines` call.

See Also

`bi_model`

Examples

```
model_file_name <- system.file(package="bi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ$write_model_file("PZ")
```

`bi_obs_file` *Create Observation Files for LibBi*

Description

This function creates a NetCDF obsersation file given a numeric vector. This file can then be passed to `libbi` using the `--obs-file` option.

Usage

```
bi_obs_file(filename, variable, name = "Y")
```

Arguments

`filename`	a path to a NetCDF file to write the variable into, which will be overwritten if it already exists.
`variable`	a `numeric` vector of observations.
`name`	a string representing the name to be used in the NetCDF file; default to "Y".

Value

None, but creates a NetCDF file at the specified path.

Note

Note that it creates a time variable with indices starting from 1, and not from 0.

`bi_open` *Bi open*

Description

This function opens an NetCDF file The file can be specified as a string to the filepath, in which case a NetCDF connection is opened, or directly as a NetCDF connection.

Usage

```
bi_open(read)
```

Arguments

`read`	either a path to a NetCDF file, or a NetCDF connection created using `nc_open`, or a `libbi` object from which to read the output

Value

open NetCDF connection

`bi_read` *Bi Read*

Description

This function reads all variable from a NetCDF file or the output of a `libbi` object. The file can be specified as a string to the filepath, in which case a NetCDF connection is opened, or directly as a NetCDF connection.

Usage

```
bi_read(read, vars, dims, missval.threshold, variables, time_dim, vector, thin,
  verbose)
```

Arguments

`read`	either a path to a NetCDF file, or a NetCDF connection created using `nc_open`, or a `libbi` object from which to read the output
`vars`	variables to read; if not given, all will be read
`dims`	factors for dimensions
`missval.threshold`	upper threshold for the likelihood
`variables`	only extract given variables (for space saving)
`time_dim`	name of time dimension (if any)
`vector`	if TRUE, will return results as vectors, not data.frames.
`thin`	thinning (keep only 1/thin of samples)

Value

list of results

`bi_read_var` *Bi Read Variable*

Description

This function reads a variable from a NetCDF file. The file can be specified as a string to the filepath, in which case a NetCDF connection is opened, or directly as a NetCDF connection.

Usage

```
  bi_read_var(resultfile, name, coord, ps, ts)
```

Arguments

`resultfile`	either a path to a NetCDF file, or a NetCDF connection created using open.ncdf(filename)
`name`	name of the variable to read (use `bi_file_summary` to learn about the variable names of a specific file)
`coord`	dimension indices (not implemented yet)
`ts`	time indices (not implemented yet)

bi_write — *Create (init or observation) files for LibBi*

Description

This function creates an init file to specify parameter values and initial conditions. This file can then be passed to `libbi` using the `--init-file` option.

Usage

```
bi_write(filename, variables, timed, ...)
```

Arguments

`filename`	a path to a NetCDF file to write the variables into, which will be overwritten if it already exists.
`variables`	a `list` object, which names should be the variable names and values should be either single values, vectors of equal length, or data frames; or a single element of the type
`timed`	if TRUE, any elements of `variables` that are vectors will be assumed to have a time dimension
`...`	arguments passed to `netcdf_create_from_list`
`time_dim`	name of the time dimension, if one exists

Value

None, but creates a NetCDF file at the specified path.

get_traces — *Get the parameter traces*

Description

This function takes the provided `libbi` which has been run and returns a data frame with the parameter traces.

Usage

```
get_traces(run, all = FALSE, model, ...)
```

Arguments

`run`	a `libbi` object which has been run, or a list of data frames containing parameter traces (as returned by from `bi_read`); if it is not a `libbi` object, either 'all' must be TRUE or a model given
`all`	whether all variables in the run file should be considered (otherwise, just parameters)
`model`	a model to get the parameter names from; not needed if 'run' is given as a `libbi` object or 'all' is set to TRUE
`...`	parameters to `bi_read` (e.g., dimensions)

Value

data frame with parameter traces; this can be fed to coda routines

---

`libbi` *LibBi Wrapper*

Description

`libbi` allows to call `libbi`. Upon creating a new libbi object, the following arguments can be given. Once the instance is created, `libbi` can be run through the `run` method documented in `libbi_run`. Note that `libbi` objects can be plotted using `plot` if the `RBi.helpers` package is loaded.

Arguments

`client`	is either "draw", "filter", "sample"... see LibBi documentation.
`model`	either a character vector giving the path to a model file (typically ending in ".bi"), or a `bi_model` object
`config`	path to a configuration file, containing multiple arguments
`global_options`	additional arguments to pass to the call to `libbi`, on top of the ones in the config file
`working_folder`	path to a folder from which to run `libbi`; default to a temporary folder.
`path_to_libbi`	path to `libbi` binary; by default it tries to locate `libbi`
`input`	input file (given as file name or `libbi` object or a list of data frames
`init`	init file (given as file name or `libbi` object or a list of data frames
`obs`	observation file (given as file name or `libbi` object or a list of data frames
`run`	(boolean) whether to run the model using the `which` Unix command, after having loaded "~/.bashrc" if present; if unsuccessful it tries "~/PathToBiBin/libbi"; if unsuccessful again it fails.

See Also

`libbi_run`

Examples

```
bi_object <- libbi$new(client = "sample",
                       model = system.file(package="bi", "PZ.bi"),
                       global_options = list(sampler = "smc2"))
```

libbi_run *Using the LibBi wrapper to launch LibBi*

Description

The method run of an instance of libbi allows to launch libbi with a particular set of command line arguments.

Arguments

add_options	additional arguments to pass to the call to libbi
output_file_name	path to the result file (which will be overwritten)
stdoutput_file_name	path to a file to text file to report the output of libbi
init	initialisation of the model, either supplied as a list of values and/or data frames, or a (netcdf) file name, or a libbi object which has been run (in which case the output of that run is used as input)
input	input of the model, either supplied as a list of values and/or data frames, or a (netcdf) file name, or a libbi object which has been run (in which case the output of that run is used as input)
obs	observations of the model, either supplied as a list of values and/or data frames, or a (netcdf) file name, or a libbi object which has been run (in which case the output of that run is used as observations)
verbose	if TRUE, will run libbi with the '–verbose' option
...	any onrecognised options will be added to add_options

Value

a list containing the absolute paths to the results; it is stored in the result field of the instance of libbi.

See Also

libbi

Examples

```
bi_object <- libbi$new(client = "sample",
                       model_file_name = system.file(package="bi", "PZ.bi"),
                       global_options = list(sampler = "smc2"))
bi_object$run(add_options=list(nthreads = 1), verbose = TRUE)
bi_file_summary(bi_object$result$output_file_name)
```

`log2normw` *Normalize log weights*

Description

This function takes a vector of real values, then takes the exponential and divides by the sum. Substracting the max of the original values increases the numerical stability.

Usage

```
log2normw(lw)
```

Arguments

`lw`	a vector of real values

Value

a vector of normalized values (summing to 1)

`netcdf_create_from_list` *Create NetCDF File from R list*

Description

This function creates a NetCDF file given a list.

Usage

```
netcdf_create_from_list(filename, variables, time_dim = "nr",
  value_column = "value")
```

Arguments

`filename`	a path to a NetCDF file to write the variable into, which will be overwritten if it already exists.
`variables`	a `list`
`time_dim`	the name of the time dimension, if one exists; "nr" by default
`value_column`	if any `variables` are data frames, which column contains the values (default: "value")

Details

The list of variables must follow the following rules. Each element of the list must itself be one of:

1) a list with two keys; the first key must be named "values" and contains a numeric vector; the second key must be named "dimension" and contains a string giving the dimension name.

2) a data frame with a `value_column` column (see option 'value_column') and any number of other columns indicating one or more dimensions

3) a numeric vector of length one, with no dimensions

The name of the list elements itself is used to create the corresponding variable in the NetCDF file.

Value

None, but creates a NetCDF file at the specified path.

Note

Two elements of the given list can possibly have the same dimension name.

Examples

```
filename <- tempfile(pattern="dummy", fileext=".nc")
a <- list(values = 1:3, dimension = "dim_a")
b <- list(values = 1:5, dimension = "dim_b")
c <- list(values = 5:9, dimension = "dim_b")
d <- 3
e <- data.frame(dim_a = rep(1:3, time = 2), dim_c = rep(1:2, each = 3), value = 1:6)
variables <- list(a=a, b=b, c=c, d=d, e=e)
netcdf_create_from_list(filename, variables)
bi_file_ncdump(filename)
```

`option_list` *Convert string to option list*

Description

This function is used to convert an option string into a list of options. If a list is given, it will be kept as is

Usage

```
option_list(...)
```

Arguments

`...`	any number of strings to convert

Value

option list

`option_string` *Convert Options*

Description

This function is used to convert a list of options into an options string. If a string is given, it will be taken as such.

Usage

```
option_string(...)
```

Arguments

`...`	any number of lists of options, or strings (which will be left unmodified). If lists are given, later arguments will override earlier ones

---

`read_var_input` *Read variable from NetCDF file.*

Description

Read variable from NetCDF file.

Usage

```
read_var_input(nc, name, coord, ps, ts)
```

Arguments

`nc`	NetCDF file handle
`name`	Name of the variable
`coord`	(optional) Demsions index.
`ps`	(optional) Path indices.
`ts`	(optional) Time indices.

Value

read values

Author(s)

Lawrence Murray, `<lawrence.murray@csiro.au>`

Index