Package 'rbi'

May 14, 2018

Version 0.9.1

Title R Interface to LibBi

Imports ncdf4, data.table, reshape2

Suggests coda, R.rsp, testthat, stringi

Description Provides a complete R interface to LibBi, a library for Bayesian inference (see <http://libbi.org> and <doi:10.18637/jss.v067.i10> for more information). This includes functions for manipulating LibBi models, for reading and writing LibBi input/output files, for converting LibBi output to provide traces for use with the coda package, and for running LibBi from R.

License GPL-3

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

BugReports https://github.com/libbi/RBi/issues

SystemRequirements LibBi (>= 1.2.0)

LazyLoad no

RoxygenNote 6.0.1

VignetteBuilder R.rsp

NeedsCompilation no

Author Pierre E. Jacob [aut],
Anthony Lee [ctb],
Lawrence M. Murray [ctb],
Sebastian Funk [aut, cre]

Maintainer Sebastian Funk <sebastian.funk@lshtm.ac.uk>

Repository CRAN

Date/Publication 2018-05-14 16:28:05 UTC

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 is made of various components:

- A wrapper around `libbi` called `libbi`.
- A `bi_model` class that can be used to load and manipulate `libbi` models
- Functions to manipulate the results of the `libbi` command, which are stored in NetCDF files. Those functions allow to extract variables of interest.

Author(s)

Pierre E. Jacob <pierre.jacob.work@gmail.com>, Anthony Lee <awllee@gmail.com>, Lawrence Murray <lawrence.murray@csiro.au>, Sebastian Funk <sebastian.funk@lshtm.ac.uk>

References

http://libbi.org/

See Also

`libbi`

Examples

```
example_output_file <- system.file(package="rbi", "example_output.nc")
bi_file_summary(example_output_file)
mu_sigma <- bi_read(example_output_file, c("mu", "sigma"))
bi_write("mu_sigma.nc", mu_sigma)

## examples for running libbi from rbi (will take a few minutes)
## Not run: demo(PZ_generate_dataset)
## Not run: demo(PZ_PMMH)
## Not run: demo(PZ_SMC2)
## Not run: demo(PZ_filtering)
```

---

add_block *Add a block to a LibBi model*

Description

Add a block to a LibBi model. If that block exists, it will be removed first.

Usage

```
## S3 method for class 'bi_model'
add_block(x, name, lines, options, ...)
```

Arguments

x	a bi_model object
name	name of the block
lines	character vector, lines in the block
options	any options to the block
...	ignored

Value

a bi_model object containing the new block

---

attach_file *Attach a new file to a libbi object*

Description

Adds an (output, obs, etc.) file to a libbi object. This is useful to recreate a libbi object from the model and output files of a previous run

Usage

```
## S3 method for class 'libbi'
attach_file(x, file, data, force = FALSE, ...)
```

Arguments

x	a libbi object
file	the type of the file to attach, one of "output", "obs", "input" or "init"
data	name of the file to attach, or a list of data frames that contain the outputs
force	attach the file even if one like this already exists in the libbi object
...	ignored

Examples

```
bi <- libbi(model = system.file(package="rbi", "PZ.bi"))
example_output_file <- system.file(package="rbi", "example_output.nc")
bi <- attach_file(bi, "output", example_output_file)
```

bi_contents *Bi contents*

Description

This function gets the name of all the variables in the passed file, list or libbi object

Usage

```
bi_contents(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

vector of variable names

Examples

```
example_output_file <- system.file(package="rbi", "example_output.nc")
bi_contents(example_output_file)
```

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
dim	name of the dimension to check

Value

dimension length

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(...)
```

Arguments

...	Any extra parameters to bi_open, especially x and file

Value

None

Examples

```
example_output_file <- system.file(package="rbi", "example_output.nc")
bi_file_summary(example_output_file)
```

bi_generate_dataset *Bi Generate Dataset*

Description

This is a wrapper around libbi sample --target joint --nsamples 1, to generate a synthetic dataset from a model. Parameters can be passed via the 'init' option (see run.libbi, otherwise they are generated from the prior specified in the model. The end time should be specified using the "end_time" option. If this is not given, only a parameter set is sampled.

Usage

```
bi_generate_dataset(..., seed)
```

Arguments

...	arguments to be passed to libbi and sample, especially 'model' and 'end_time'.
seed	random seed; see the seed option of sample for details.

Value

generated data set

bi_model *Bi Model*

Description

bi_model creates a model object for Rbi from a libbi file, URL or character vector. Once the instance is created, the model can be fed to a libbi object.

Usage

```
bi_model(filename, lines, ...)
```

Arguments

filename	the file name of the model file
lines	lines of the model (if no filename is given), a character vector
...	ignored

See Also

fix, insert_lines, remove_lines, replace_all, get_name, set_name, write_model

Examples

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

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(x, file = "output")
```

Arguments

x	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
file	file to open (out of "input", "init", "obs", "output"), if x is given as a libbi object; by default, will read output file

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(x, vars, dims, model, type, file, missval.threshold, coord_dims, vector,
  thin, verbose, clear_cache, init.to.param = FALSE)
```

Arguments

`x`	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
`model`	model file or a `bi_model` object (if `x` is not a `libbi` object)
`type`	vector of types of variable to read (out of "param", "state", "noise", "obs"). This needs 'x' to be a `libbi` object or `model` to be specified
`file`	which file to read (if `x` is given as a `libbi` object): one of "output" (default), "init", "input", "obs"
`missval.threshold`	upper threshold for the likelihood
`coord_dims`	any `coord` dimensions, given as a named list of character vectors, where each element corresponds to the variable of the same name, and the character vector are the `coord` dimensions
`vector`	if TRUE, will return results as vectors, not data.frames
`thin`	thinning (keep only 1/thin of samples)
`verbose`	if TRUE, will print variables as they are read
`clear_cache`	if TRUE, will clear the cache and re-read the file even if cached data exists
`init.to.param`	logical; if TRUE, convert states to initial values

Value

list of results

Examples

```
example_output_file <- system.file(package="rbi", "example_output.nc")
d <- bi_read(example_output_file)
```

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

Description

This function creates a NetCDF file for LibBi from the given list of vectors and/or data frames. Since any files can be passed to `libbi` directly via the `init`, `input` and `obs` options, this is mostly used internally, this is mostly used internally.

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. If necessary, ".nc" will be added to the file name
`variables`	a `list` object, the names of which should be the variable names and values should be either single values or data frames
`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`

Value

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

Examples

```
filename <- tempfile(pattern="dummy", fileext=".nc")
a <- 3
b <- c(1, 3, 6)
c <- 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)
bi_write(filename, variables)
bi_file_summary(filename)
```

Extract.bi_model *Subset and replace model lines*

Description

Extracts a subset of lines from the model and, if used with the assignment operator, assigns new character strings.

Usage

```
## S3 replacement method for class 'bi_model'
x[i, ...] <- value
```

Arguments

x	A bi_model
i	A vector of line numbers
...	ignored
value	A vector of the same length as i, containing the replacement strings

Examples

```
model_file_name <- system.file(package="rbi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ[3:4]
PZ[3:4] <- c("const e = 0.4", "const m_l = 0.05")
```

extract_sample *Extract a sample from a* LibBi *run.*

Description

This function takes the provided libbi results and extracts a data frame.

Usage

```
extract_sample(x, np, ...)
```

Arguments

x	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
np	iteration to extract; if set to "last", the last sample will be extracted. If not given a random sample will be extracted
...	parameters to bi_read (e.g., dimensions)

Value

list of parameters and trajectories

`filter` *Using the LibBi wrapper to filter*

Description

The method `filter` launches `libbi` to filter state trajectories. See the options to `run.libbi` for how to specify the various components of sampling with LibBi, and the LibBi manual for all options that can be passed when the client is `filter`.

If `x` is given as a 'bi_model', a `libbi` object will be created from the model For the help page of the base R `filter` function, see `filter`.

Usage

```
## S3 method for class 'libbi'
filter(x, ...)

## S3 method for class 'bi_model'
filter(x, ...)
```

Arguments

`x`	a `libbi` or `bi_model` object, or the name of a file containing the model
`...`	options to be passed to `run.libbi`

Value

a `libbi` object

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

Description

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

For the help page of the base R `fix` function, see `fix`.

Usage

```
## S3 method for class 'bi_model'
fix(x, ...)
```

Arguments

x	a bi_model object
...	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="rbi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ <- fix(PZ, alpha = 0)
```

---

get_block *Get the contents of a block in a LibBi model*

Description

Returns the contents of a block in a LibBi model as a character vector of lines.

Usage

```
## S3 method for class 'bi_model'
get_block(x, name, ...)
```

Arguments

x	a bi_model object
name	name of the block
...	ignored

Value

a character vector of the lines in the block

get_name *Get the name of a bi model*

Description

Extracts the name of a bi model (first line of the .bi file).

Usage

```
## S3 method for class 'bi_model'
get_name(x, ...)
```

Arguments

x	a bi_model object
...	ignored

Value

the name of the model

See Also

bi_model

Examples

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

get_traces *Get the parameter traces*

Description

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

Usage

```
get_traces(x, model, burnin, all = FALSE, ...)
```

Arguments

x	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
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
burnin	proportion of iterations to discard as burn-in (if between 0 and 1), or number of samples to discard (if >1)
all	whether all variables in the run file should be considered (otherwise, just parameters)
...	parameters to bi_read (e.g., dimensions)

Value

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

---

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 or block name, respectively. If one of at_beginning of or at_end_of is given, the lines will be inserted at the beginning/end of the block, respectively.

Usage

```
## S3 method for class 'bi_model'
insert_lines(x, lines, before, after, at_beginning_of,
  at_end_of, ...)
```

Arguments

x	a bi_model object
lines	vector or line(s)
before	line number before which to insert line(s)
after	line number after which to insert line(s)
at_beginning_of	block at the beginning of which to insert lines(s)
at_end_of	block at the end of which to insert lines(s)
...	ignored

Value

the updated bi model

See Also

bi_model

Examples

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

---

join *Join multiple libbi objects*

Description

This function can be used to join multiple libbi objects into one (e.g., parallel MCMC runs into one long change)

Usage

```
## S3 method for class 'libbi'
join(x, ...)
```

Arguments

x	a libbi object
...	ignored

---

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 sample, filter, or optimise, or rewrite methods. Note that libbi objects can be plotted using plot if the rbi.helpers package is loaded.

Usage

```
libbi(model, path_to_libbi, dims, use_cache = TRUE, ...)
```

Arguments

model	either a character vector giving the path to a model file (typically ending in ".bi"), or a bi_model object
path_to_libbi	path to LibBi binary; by default it tries to locate the libbi binary using the which Unix command, after having loaded "~/.bashrc" if present; if unsuccessful it tries "~/PathToBiBin/libbi"; if unsuccessful again it fails.
dims	any named dimensions, as list of character vectors
use_cache	logical; whether to use the cache (default: true)
...	options passed to run.libbi

Value

a libbi object

See Also

sample, filter, optimise, rewrite

Examples

```
bi_object <- libbi(model = system.file(package="rbi", "PZ.bi"))
```

---

logLik *Using the LibBi wrapper to logLik*

Description

The method logLik extracts the log-likelihood of a libbi object. This can be done, for example, after a call to sample to inspect the chain log-likelihoods.

For the help page of the base R logLik function, see logLik.

Usage

```
## S3 method for class 'libbi'
logLik(object, ...)
```

Arguments

object	a libbi object
...	options to be passed to run.libbi

Value

a vector of log-likelihood

`netcdf_create_from_list` *Create NetCDF File from R list*

Description

Internal function that creates a NetCDF file given a list.

Usage

```
netcdf_create_from_list(filename, variables, time_dim, coord_dims, dim_factors,
  value_column = "value", guess_time = FALSE, guess_coord = FALSE,
  verbose)
```

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; default: "time"
`coord_dims`	the names of the coordinate dimension, if any; should be a named list of character vectors, they are matched to variables names
`dim_factors`	factors that dimensions have; this corresponds to the `dims` element of a `libbi` object
`value_column`	if any `variables` are data frames, which column contains the values (default: "value")
`guess_time`	whether to guess time dimension; this would be a numerical column in the data frame given which is not the `value_column`; only one such column must exist
`guess_coord`	whether to guess the coordinate dimension; this would be a column with varying value which is not the time or value column
`verbose`	if TRUE, will print variables as they are read

Details

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

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

2) 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

A list of the time and coord dims, and factors in extra dimensions, if any

Note

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

`optimise` *Using the LibBi wrapper to optimise*

Description

The method `optimise` launches `libbi` to optimise the parameters with respect to the likelihood or posterior distribution. See the options to `run.libbi` for how to specify the various components of sampling with LibBi, and the LibBi manual for all options that can be passed when the client is `optimise`.

If `x` is given as a 'bi_model', a `libbi` object will be created from the model For the help page of the base R `optimise` function, see `optimise`.

Usage

```
## S3 method for class 'libbi'
optimise(x, ...)

## S3 method for class 'bi_model'
optimise(x, ...)
```

Arguments

`x`	a `libbi` or `bi_model` object, or the name of a file containing the model
`...`	options to be passed to `run.libbi`

Value

a `libbi` object

`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

---

predict *Using the LibBi wrapper to predict*

Description

The method `predict` is an alias for `sample(target="prediction")`. Usually, an `init` object or file should be given containing posterior samples.

For the help page of the base R `optimise` function, see `optimise`.

Usage

```
## S3 method for class 'libbi'
predict(object, ...)
```

Arguments

object	a `libbi` object
...	ignored

read_libbi *Read results of a* LibBi *run from an RDS file. This completely reconstructs the saved* LibBi *object*

Description

This reads all options, files and outputs of a LibBi run to an RDS file specified

Usage

```
read_libbi(file, ...)
```

Arguments

file	name of the RDS file to read
...	any extra options to pass to read_libbi when creating the new object

Value

a libbi object

remove_lines *Remove line(s) and/or block(s) in a libbi model*

Description

Removes one or more lines in a libbi model.

Usage

```
## S3 method for class 'bi_model'
remove_lines(x, what, only, ...)
```

Arguments

x	a bi_model object
what	either a vector of line number(s) to remove, or a vector of blocks to remove (e.g., "parameter")
only	only remove lines assigning given names (as a vector of character strings)
...	ignored

Value

the updated bi model

See Also

bi_model

Examples

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

---

replace_all *Replace all instances of a string with another in a model*

Description

Takes every occurrence of one string and replaces it with another

Usage

```
## S3 method for class 'bi_model'
replace_all(x, from, to, ...)
```

Arguments

x	a bi_model object
from	string to be replaced (a regular expression)
to	new string (which can refer to the regular expression given as from)
...	ignored

Value

the updated bi model

See Also

bi_model

rewrite *Using the LibBi wrapper to rewrite*

Description

The method `rewrite` launches `LibBi` to rewrite a model to inspect its internal representation in `LibBi`

If `x` is given as a 'bi_model', a `libbi` object will be created from the model

Usage

```
## S3 method for class 'libbi'
rewrite(x, ...)

## S3 method for class 'bi_model'
rewrite(x, ...)
```

Arguments

`x`	a `libbi` or `bi_model` object, or the name of a file containing the model
`...`	options to be passed to `run.libbi`

Value

a `bi_model` object

run *Using the LibBi wrapper to launch LibBi*

Description

The method `run` launches `LibBi` with a particular set of command line #' arguments. Normally, this function would not be run by the user, but instead one of the client functions `sample`, `filter`, or `optimise`, or `rewrite`, which pass any options on to `run`. Note that any options specified here are stored in the `libbi` object and do not have to be specified again if another command is run on the object.

Usage

```
## S3 method for class 'libbi'
run(x, client, proposal = c("model", "prior"), model, fix,
  options, config, add_options, log_file_name, init, input, obs, time_dim,
  coord_dims, working_folder, output_all, sample_obs, thin, chain = TRUE,
  seed = TRUE, ...)
```

Arguments

x	a libbi object
client	client to pass to LibBi
proposal	proposal distribution to use; either "model" (default: proposal distribution in the model) or "prior" (propose from the prior distribution)
model	either a character vector giving the path to a model file (typically ending in ".bi"), or a bi_model object; by default, will use any model given in x
fix	any variable to fix, as a named vector
options	list of additional arguments to pass to the call to LibBi. Any arguments starting with 'enable'/'disable' can be specified as boolean (e.g., 'assert=TRUE'). Any 'dry-' options can be specified with a '"dry"' argument, e.g., 'dry="parse"'. Any options that would be specified with 'with'/'without' can be specified as character vector to an option named 'with'/'without', respectively, e.g. with="transform-obs-to-state".
config	path to a configuration file, containing multiple arguments
add_options	deprecated, replaced by options
log_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). If the object given as x has been run before, it will be used here with init-np set to the last iteration of the previous run, unless init is given explicitly.
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)
time_dim	The time dimension in any R objects that have been passed (init, input) and obs); if not given, will be guessed
coord_dims	The coord dimension(s) in any obs R objects that have been passed; if not given, will be guessed
working_folder	path to a folder from which to run LibBi; default to a temporary folder.
output_all	logical; if set to TRUE, all parameters, states and observations will be saved; good for debugging
sample_obs	logical; if set to TRUE, will sample observations
thin	any thinning of MCMC chains (1 means all will be kept, 2 skips every other sample etc.); note that LibBi itself will write all data to the disk. Only when the results are read in with bi_read will thinning be applied.
chain	logical; if set to TRUE and x has been run before, the previous output file will be used as init file, and init-np will be set to the last iteration of the previous run (unless target=="prediction"). This is useful for running inference chains.

seed	Either a number (the seed to supply to LibBi), or a logical variable: TRUE if a seed is to be generated for RBi, FALSE if LibBi is to generate its own seed
...	any unrecognised options will be added to options

Value

a libbi object, except if client is 'rewrite', in which case a bi_model object will be returned

See Also

libbi

Examples

```
bi_object <- libbi(model = system.file(package="rbi", "PZ.bi"))
## Not run: run(bi_object, options=list(client="sample", sample="smc2"))
if (bi_object$run_flag) {
  bi_file_summary(bi_object$output_file_name)
}
```

---

sample *Using the LibBi wrapper to sample*

Description

The method sample launches libbi to sample from a (prior, posterior or joint) distribution. See the options to run.libbi for how to specify the various components of sampling with LibBi, and the LibBi manual for all options that can be passed when the client is sample.

If x is given as a 'bi_model', a libbi object will be created from the model For the help page of the base R sample function, see sample.

Usage

```
## S3 method for class 'libbi'
sample(x, ...)

## S3 method for class 'bi_model'
sample(x, ...)
```

Arguments

x	a libbi or bi_model object, or the name of a file containing the model
...	options to be passed to run.libbi

Value

a libbi object

save_libbi *Write results of a* LibBi *run to an RDS file*

Description

This saves all options, files and outputs of a LibBi run to an RDS file specified

Usage

```
## S3 method for class 'libbi'
save_libbi(x, filename, supplement, ...)
```

Arguments

x	a libbi object
filename	name of the RDS file to save to
supplement	any supplementary data to save
...	any options to saveRDS

set_name *Set the name of a bi model*

Description

Changes the name of a bi model (first line of the .bi file) to the specified name.

Usage

```
## S3 method for class 'bi_model'
set_name(x, name, ...)
```

Arguments

x	a bi_model object
name	Name of the model
...	ignored

Value

the updated bi model

See Also

bi_model

Examples

```
model_file_name <- system.file(package="rbi", "PZ.bi")
PZ <- bi_model(filename = model_file_name)
PZ <- set_name(PZ, "new_PZ")
```

---

summary *Print summary information about a* libbi *object*

Description

This reads in the output file of the libbi object (which has been run before) and prints summary information of parameters

Usage

```
## S3 method for class 'libbi'
summary(object, ...)
```

Arguments

object	a libbi object
...	ignored

---

var_names *Get variables*

Description

Get all variable names of one or more type(s)

This returns all variable names of a certain type ("param", "state", "obs", "noise", "const") contained in the model of a libbi object

Usage

```
## S3 method for class 'bi_model'
var_names(x, type, dim = FALSE, opt = FALSE,
  aux = FALSE, ...)
```

Arguments

x	a bi_model object
type	a character vector of one or more types
dim	logical; if set to TRUE, names will contain dimensions in brackets
opt	logical; if set to TRUE, names will contain options (e.g., has_output)
aux	logical; if set to TRUE, auxiliary names will be returned
...	ignored

Value

variable names

`write_model` *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.

Usage

```
## S3 method for class 'bi_model'
write_model(x, filename, update.name = TRUE, ...)

## S3 method for class 'libbi'
write_model(x, filename, ...)
```

Arguments

`x`	a `bi_model` object, or a `libbi` object containing a model
`filename`	name of the file to be written
`update.name`	whether to update the model name with the file name
`...`	ignored

Value

the return value of the `writeLines` call.

See Also

`bi_model`

Examples

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

Index