Package 'LSDinterface'

September 22, 2021

Type Package

Title Interface Tools for LSD Simulation Results Files

Version 1.1.0

Date 2021-9-22

Description Interfaces R with LSD simulation models. Reads object-oriented data in results files (.res[.gz]) produced by LSD and creates appropriate multi-dimensional arrays in R. Supports multiple core parallel threads of multi-file data reading for increased performance. Also provides functions to extract basic information and statistics from data files. LSD (Laboratory for Simulation Development) is free software developed by Marco Valente (documentation and downloads available at <https://www.labsimdev.org/>).

Depends R (>= 3.2.0)

Imports stats, boot, utils, abind, parallel

Suggests LSDsensitivity

Language en-US

Encoding UTF-8

License GPL-3

NeedsCompilation no

Author Marcelo C. Pereira [aut, cre] (<https://orcid.org/0000-0002-8069-2734>)

Maintainer Marcelo C. Pereira <mcper@unicamp.br>

Repository CRAN

Date/Publication 2021-09-22 20:10:02 UTC

R topics documented:

`LSDinterface-package` *Interface Tools for LSD Simulation Results Files*

Description

Interfaces R with LSD simulation models. Reads object-oriented data in results files (.res[.gz]) produced by LSD and creates appropriate multi-dimensional arrays in R. Supports multiple core parallel threads of multi-file data reading for increased performance. Also provides functions to extract basic information and statistics from data files. LSD (Laboratory for Simulation Development) is free software developed by Marco Valente (documentation and downloads available at <https://www.labsimdev.org/>).

Details

There are specific `read.xxx.lsd()` functions for different types of LSD data structures.

`read.raw.lsd()` simply import LSD saved data in tabular (data frame) format (variables in columns and time steps in rows). `read.single.lsd()` is appropriate to simple LSD data structures where each saved variable is single-instanced (inside an object with a single copy). `read.multi.lsd()` reads all instances of all variables from the LSD results file, renaming multi-instanced variables. `read.list.lsd()` is similar to `read.multi.lsd()` but saves multiple-instanced variables as R lists, preventing renaming.

`read.3d.lsd()` and `read.4d.lsd()` are specialized versions for extracting data from multiple LSD results files simultaneously. The files must have the same structure (selected variables and number of time steps). They are frequently used to acquire data from Monte Carlo experiments or sensitivity analysis. `read.3d.lsd()` operates like `read.single.lsd()` but add each additional results file into a separate dimension of the produced 3-dimensional array (variable x time step x file). `read.4d.lsd()` adds the ability to read each instance of a multi-instanced variable to the fourth dimension of the generated 4D array (variable x instance x time step x file).

`list.files.lsd()` is a helper function to simplify the collection of results files to be used by the other functions in this package. It can be directly used to supply the `files` argument in the `read.xxx.lsd()` family of functions.

select.colattrs.lsd() and select.colnames.lsd() provide methods to extract/summarize information from previously imported LSD data structures.

info.xxx.lsd() functions provide information about LSD data structures. name.xxx.lsd() functions offer tools for dealing with LSD variable names in R.

For a complete list of exported functions, use library(help = "LSDinterface").

Author(s)

NA

Maintainer: NA

References

LSD documentation is available at https://www.labsimdev.org/.

The latest LSD binaries and source code can be downloaded at https://github.com/marcov64/Lsd/.

---

info.details.lsd *Get detailed information from a LSD results file*

Description

This function reads, analyze and organize the information from a LSD results file (.res).

Usage

```
info.details.lsd( file )
```

Arguments

file	the name of the LSD results file which the data are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory, getwd(). Tilde-expansion is performed where supported. This can be a compressed file (see file) and must include the appropriated extension (usually .res or .res.gz).

Value

Returns a data frame containing detailed description (columns) of all variables (rows) contained in the selected results file.

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() info.init.lsd(), info.names.lsd() info.dimensions.lsd()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# get details about all variables in first file
info.details.lsd( files[ 1 ] )
```

`info.dimensions.lsd` *Dimension information for a LSD results file*

Description

This function reads some dimension information from a LSD results file (.res): number of time steps, number of variables and the original column (variable) names.

Usage

```
info.dimensions.lsd( file )
```

Arguments

`file`	the name of the LSD results file which the data are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory, `getwd()`. Tilde-expansion is performed where supported. This can be a compressed file (see file) and must include the appropriated extension (usually `.res` or `.res.gz`).

Details

The returned number of time steps does not include the initial value (t = 0) for lagged variables (the second line of a .res format file).

Value

Returns a list containing two integer values and a character vector describing the selected results file.

`tSteps`	Number of time steps in file
`nVars`	Number of variables (including duplicated instances) in file
`varNames`	Names of variables (including duplicated instances) in file, after R name conversion

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() info.details.lsd(), info.names.lsd(), info.init.lsd()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# get dimensions from second file
info.dimensions.lsd( files[ 2 ] )
```

---

info.init.lsd *Read initial conditions from a LSD results file*

Description

This function reads the initial condition values from a LSD results file (.res).

Usage

```
info.init.lsd( file )
```

Arguments

file	the name of the LSD results file which the data are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory, getwd(). Tilde-expansion is performed where supported. This can be a compressed file (see file) and must include the appropriated extension (usually .res or .res.gz).

Value

Returns a 1 line matrix containing the initial conditions (row 1) of all variables contained in the selected results file.

Note

The returned matrix contains all variables in the results file, even the ones that don't have an initial condition (indicated as NA). Only variables automatically initialized automatically by LSD in t = 1 are included here.

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() info.details.lsd(), info.names.lsd() info.dimensions.lsd()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# get initialization data from first and second files
init1 <- info.init.lsd( files[ 1 ] )
init1[ , 4 : 8 ]

init2 <- info.init.lsd( files[ 2 ] )
init2[ , 4 : 8 ]
```

`info.names.lsd` *Read unique variable names from a LSD results file (no duplicates)*

Description

This function reads the variable names (columns) from a LSD results file (.res). The names returned are converted to the original LSD names whenever possible and duplicates are removed.

Usage

```
info.names.lsd( file )
```

Arguments

`file`	the name of the LSD results file which the data are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory, `getwd()`. Tilde-expansion is performed where supported. This can be a compressed file (see file) and must include the appropriated extension (usually `.res` or `.res.gz`).

Value

Returns a character vector containing the names of all unique variables contained in the selected results file.

Note

Not all names can be automatically reconverted to the original LSD names. The conversion may be incorrect if the original LSD variable is named in the format "X_...".

Author(s)

Marcelo C. Pereira

See Also

`list.files.lsd()` `info.details.lsd()`, `info.init.lsd()` `info.dimensions.lsd()`

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# get variable names from first file
info.names.lsd( files[ 1 ] )
```

`info.stats.lsd` *Compute Monte Carlo statistics from a set of LSD runs*

Description

This function reads a 3 or 4-dimensional array produced by `read.3d.lsd` or `read.4d.lsd` and produces a list with 2D data frames containing the (Monte Carlo) mean, the standard deviation, the maximum, the minimum, and other optional statistics for each variable, at each time step.

Usage

```
info.stats.lsd( array, rows = 1, cols = 2, median = FALSE,
                ci = c( "none", "mean", "median", "auto" ),
                ci.conf = 0.95, ci.boot = NULL, boot.R = 999,
                na.rm = TRUE, inf.rm = TRUE )
```

Arguments

`array`	an 3D or 4D array as produced by `read.3d.lsd` and `read.4d.lsd`, where in the first dimension (rows) you have the time steps, in the second (columns), the variables and in the third/fourth dimension, the Monte Carlo experiments, and the instances in the third dimension (4D arrays only).
`rows`	an integer array dimension to be used as the rows for the statistics matrices, default is to use first array dimension.
`cols`	an integer array dimension to be used as the columns for the statistics matrices, default is to use second array dimension.
`median`	a logical value indicating if (`TRUE`) the median should also be computed.
`ci`	a character string specifying the type of confidence interval to compute, must be one of `"none"` (default) for no confidence interval computation, `"mean"`, to compute a confidence interval for the mean, `"median"`, for the median, or `"auto"`, to use the option set for the `median` argument (above). This option can be abbreviated.
`ci.conf`	confidence level of the confidence interval
`ci.boot`	a character string specifying the type of bootstrap confidence interval to compute, must be one of `"basic"`, `"perc"` (percentile interval), or `"bca"` (BCa - adjusted percentile interval). If set to `NULL` a regular asymptotic confidence interval is produced (no bootstrap), assuming normal distribution for the mean or using a non-parametric rank test for the median. Non-bootstrap percentiles are much faster to compute but generally less accurate.

boot.R	number of bootstrap replicates.
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
inf.rm	a logical value indicating whether non-finite values should be stripped before the computation proceeds.

Value

Returns a list containing four to seven matrices, with the original size and naming of the selected 2 dimensions of the argument.

avg	a matrix with the mean of the MC experiments
sd	a matrix with the standard deviation of the MC experiments
max	a matrix with the maximum value of the MC experiments
min	a matrix with the minimum value of the MC experiments
med	a matrix with the median of the MC experiments (only present if argument median = TRUE)
ci.hi	a matrix with the maximum value of the MC experiments (only present if argument ci is not set to "none")
ci.lo	a matrix with the minimum value of the MC experiments (only present if argument ci is not set to "none")

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() read.3d.lsd(), read.4d.lsd(), info.dimensions.lsd()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# read first instance of all variables from MC files (3D array)
inst1Array <- read.3d.lsd( files )

# create statistics data frames for the variables
inst1Stats <- info.stats.lsd( inst1Array )
print( inst1Stats$avg[ 10 : 20, ] )
print( inst1Stats$sd[ 10 : 20, ] )

# organize the stats, including medians, by variable (dim=2) and file (dim=3)
inst1Stats2 <- info.stats.lsd( inst1Array, rows = 2, cols = 3, median = TRUE )
print( inst1Stats2$med[ , 1 : 2 ] )

# the same but for all instance of all variables (from a 4D array)
# and a normal (non-boostrap) confidence intervals for the means
```

```
allArray <- read.4d.lsd( files )
allStats <- info.stats.lsd( allArray, ci = "auto" )
print( allStats$ci.lo[ 3, 1 : 7 ] )
print( allStats$avg[ 3, 1 : 7 ] )
print( allStats$ci.hi[ 3, 1 : 7 ] )

# organize the stats by file (dim=4) and variable (dim=2)
# plus boostrat confidence intervals for the median
allStats2 <- info.stats.lsd( allArray, rows = 4, cols = 2, median = TRUE,
                             ci = "auto", ci.boot = "bca" )
print( allStats2$ci.lo[ , 1 : 3 ] )
print( allStats2$med[ , 1 : 3 ] )
print( allStats2$ci.hi[ , 1 : 3 ] )
```

`list.files.lsd` *List results files from a set of LSD runs*

Description

This function produce a character vector of the names of results files produced after the execution of LSD simulation runs. The list can be used with all function in this package requiring the argument `files`.

Usage

```
list.files.lsd( path = ".", conf.name = "",
                type = c( "res", "tot", "csv" ),
                compressed = NULL, recursive = FALSE,
                join = FALSE, full.names = FALSE,
                sensitivity = FALSE )
```

Arguments

`path`	a character vector of full or relative path name to the base directory from where to search the files; the default corresponds to the working directory, `getwd()`. Tilde expansion is performed. Alternatively, the full path and name of the corresponding LSD configuration file (including the `.lsd` extension) can be provided.
`conf.name`	the LSD configuration file name (optionally including the `.lsd` extension) used to generate the desired results files; the default is to return all results files, irrespective of the configuration file used. Alternatively, a regular expression can be supplied. This argument takes precedence of any configuration file name provided together with the `path` argument.
`type`	the type (format/extension) of LSD results files to use among the options `c( "res","tot","csv" )`, used to define the extension of the files to be considered. `"res"` is the default. This option can be abbreviated.

`compressed` a logical value indicating if (`TRUE`) to look only for compressed files with `.gz` extension, or uncompressed ones otherwise (`FALSE`). The default (`NULL`) is to list files irrespective if compressed or not.

`recursive` a logical value indicating if the listing should recurse into sub-directories of `path`. The default (`FALSE`) is to scan just the sub-directory with the same name as `conf.name` (without the `.lsd` extension or numeric tags), if present (regular expression in `conf.name` is not considered), and `path`. If `TRUE`, the entire sub-directory tree, starting at `path`, is scanned for files.

`join` a logical value indicating if results files from multiple sub-directories should be joined together in the return list. The default (`FALSE`) is to list files from just a single sub-directory, the first one found during the search starting from `path`.

`full.names` a logical value specifying if (`TRUE`) the file names should be expanded to absolute path names. The default (`FALSE`) is to use relative (to `path`) file names.

`sensitivity` a logical value specifying if (`TRUE`) the target results files are part of a sensitivity analysis design of experiment (DoE), which are double numbered in a particular format (`conf.name_XXX_YYY.res[.gz]`). The default (`FALSE`) is to assume files are just single numbered, which is usually inappropriate for DoE results files. See LSDsensitivity package documentation for details.

Details

The order by which sub-directories are explored may be relevant. By default, the function scans for results files in a sub-directory named as `conf.name`, if present, in the given initial directory `path`. Next, if `conf.name` has a numeric suffix in the format `name_XXX`, where XXX is any number of algarisms, it searches the sub-directory `name`, if present. Finally, it scans the initial `path` itself. If results files are present in more than one sub-directory, function returns only the files found in first one (except if `join = TRUE`), and issues a warning message. If `recursive = TRUE`, file search starts from `path` and proceeds until it encompasses the entire sub-directory tree. In this case, if multiple sub-directories contain the desired files, only the initial `path` takes precedence, and the rest of the tree is recurred in alphabetical order.

Please note that joining files from different sub-directories (`join = TRUE`) may combine results with incompatible data which cannot be processed together by the `read.xxx.lsd()` family of functions.

Value

A character vector containing the names of the found results files in the specified (sub) directories (empty if there were no files). If a path does not exist or is not a directory or is unreadable it is skipped.

Note

File naming conventions are platform dependent. The pattern matching works with the case of file names as returned by the OS.

`path` must specify paths which can be represented in the current codepage, and files/directories below path whose names cannot be represented in that codepage will most likely not be found.

Author(s)

Marcelo C. Pereira

See Also

read.3d.lsd(), read.4d.lsd(), read.raw.lsd(), read.single.lsd(), read.multi.lsd(), read.list.lsd(), LSDsensitivity package,

Examples

```
# get the names of all files the example directory
list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# expand search to the entire example directory tree
# for results from a configuration file named "Sim1.lsd"
# and join files found in all sub-directories conatining data
list.files.lsd( system.file( "extdata", package = "LSDinterface" ),
                "Sim1.lsd", recursive = TRUE, join = TRUE )
```

---

name.check.lsd *Check a set of LSD variables names against a LSD results file*

Description

This function checks if all variable names in a set are valid for a LSD results file (.res). If no name is provided, the function returns all the valid unique variable names in the file.

Usage

```
name.check.lsd( file, col.names = NULL, check.names = TRUE )
```

Arguments

file	the name of the LSD results file which the data are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory, getwd(). This can be a compressed file (see file) and must include the appropriated extension (usually .res or .res.gz).
col.names	a vector of optional names for the variables. The default is to read all (unique) variables.
check.names	logical. If TRUE then the names of the variables are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by make.names) so that they are, and also to ensure that there are no duplicates.

Value

Returns a character vector containing the valid variable names contained in the results file.

Author(s)

Marcelo C. Pereira

See Also

`list.files.lsd()` `info.names.lsd()`,

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# check all variable names
name.check.lsd( files[ 1 ] )

# check just two names
name.check.lsd( files[ 2 ], col.names = c( "GDP", "_growth1" ) )
```

---

name.clean.lsd *Get clean (R) variable name*

Description

This function produces a more appropriate variable name from R initial column name conversion.

Usage

```
name.clean.lsd( r.name )
```

Arguments

`r.name`	a character vector, or an object which can be coerced to a character vector by as.character, from the column names produced by reading a LSD results file.

Details

The function removes the extra/ending '.' characters introduced by R and introduces a '_' between time span values.

Value

A character vector of with the same attributes as x (after possible coercion) and the format `NAME.POSITION.INI_END`.

Author(s)

Marcelo C. Pereira

See Also

name.var.lsd(), name.nice.lsd(), info.names.lsd()

Examples

```
name.clean.lsd( "Var1.1_1..1.100." )

name.clean.lsd( c( "Var1.1_1..1.100.", "Var2.1_2_3..50.70." ) )
```

---

name.nice.lsd *Get a nice (R) variable name*

Description

This function produces a nicer variable name from R initial column name conversion, in particular removing leading underscores.

Usage

```
name.nice.lsd( r.name )
```

Arguments

r.name	a character vector, or an object which can be coerced to a character vector by as.character, from the column names produced by reading a LSD results file.

Details

The function removes the extra/ending '.' characters introduced by R and introduces a '_' between time span values and deletes leading underscores ('_'), converted to 'X_' by R.

Value

A character vector of with the same attributes as x (after possible coercion) and the format NAME[.POSITION.INI_END].

Author(s)

Marcelo C. Pereira

See Also

name.var.lsd(), name.clean.lsd(), info.names.lsd()

Examples

```
name.nice.lsd( "X_Var1.1_1..1.100." )

name.nice.lsd( c( "_Var1.1_1..1.100.", "X_Var2.1_2_3..50.70." ) )

name.nice.lsd( c( "_Var1", "X_Var2" ) )
```

`name.var.lsd` *Get original LSD variable name*

Description

This function generates the original LSD variable name, as it was defined in LSD and before R adjusts the name, from a R column name (with or without position or timing information appended).

Usage

```
name.var.lsd( r.name )
```

Arguments

`r.name`	a character vector, or an object which can be coerced to a character vector by as.character, from the column names produced by reading a LSD results file.

Details

The conversion may be incorrect if the original LSD variable is named in the format "X_...". No checking is done to make sure the variable really exists.

Value

A character vector of with the same attributes as x (after possible coercion).

Author(s)

Marcelo C. Pereira

See Also

`name.clean.lsd()`, `info.names.lsd()`

Examples

```
name.var.lsd( "label" )

name.var.lsd( c( "label", "X_underlinelabel" ) )
```

`read.3d.lsd` *Read one instance of LSD variables (time series) from multiple LSD results files into a 3D array*

Description

This function reads the data series associated to a specific instance of each selected variable from a set of LSD results files (.res) and saves them into a 3-dimensional array (time step x variable x file).

Usage

```
read.3d.lsd( files, col.names = NULL, nrows = -1, skip = 0,
             check.names = TRUE, instance = 1, nnodes = 1 )
```

Arguments

`files`	a character vector containing the names of the LSD results files which the data are to be read from. If they do not contain an absolute path, the file names are relative to the current working directory, `getwd()`. These can be compressed files and must include the appropriated extension (usually `.res` or `.res.gz`).
`col.names`	a vector of optional names for the variables. The default is to read all variables.
`nrows`	integer: the maximum number of time steps (rows) to read in. Negative and other invalid values are ignored. The default is to read all rows.
`skip`	integer: the number of time steps (rows) of the results file to skip before beginning to read data. The default is to read from the first time step (t = 1).
`check.names`	logical. If TRUE then the names of the variables are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by make.names) so that they are, and also to ensure that there are no duplicates.
`instance`	integer: the instance of the variable to be read, for variables that exist in more than one object. This number is based on the position (column) of the variable in the results file. The default (1) is to read first instances.
`nnodes`	integer: the maximum number of parallel computing nodes (parallel threads) in the current computer to be used for reading the files. The default, `nnodes = 1`, means single thread processing (no parallel threads). If equal to zero, creates up to one node per CPU core. Only `PSOCK` clusters are used, to ensure compatibility with any platform. Please note that each node requires its own memory space, so memory usage increases linearly with the number of nodes.

Value

Returns a 3D array containing data series from the selected variables.

Note

If the selected files don't have the same columns available (names and instances), an error is produced.

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() read.4d.lsd(), read.single.lsd(), read.multi.lsd(), read.list.lsd(), read.raw.lsd()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# read first instance of all variables from files (one level each),
# pasting the directory where the example files are (not required if in working dir)
inst1Array <- read.3d.lsd( files )
print( inst1Array[ 5 : 10, 1 : 7, 1 ] )
print( inst1Array[ 5 : 10, 1 : 7, 2 ] )
print( inst1Array[ 5 : 10, 1 : 7, 3 ] )

# read first instance of a set of variables named '_A1p' and '_growth1'
ab1Array <- read.3d.lsd( files, c( "_A1p", "_growth1" ) )
print( ab1Array[ 20 : 25, , 1 ] )
print( ab1Array[ 20 : 25, , 2 ] )
print( ab1Array[ 20 : 25, , 3 ] )

# read instance 2 of all variables, skipping the initial 20 time steps
# and keeping up to 30 time steps (from t = 21 up to t = 30)
inst2Array21_30 <- read.3d.lsd( files, skip = 20, nrows = 30, instance = 2 )
print( inst2Array21_30[ , 1 : 7, 1 ] )
print( inst2Array21_30[ , 1 : 7, 2 ] )
```

---

read.4d.lsd *Read all instances of LSD variables (time series) from multiple LSD results file into a 4D array*

Description

This function reads the data series associated to all instances of each selected variable from a set of LSD results files (.res) and saves them into a 4-dimensional array (time step x variable x instance x file).

Usage

```
read.4d.lsd( files, col.names = NULL, nrows = -1, skip = 0,
             check.names = TRUE, pool = FALSE, nnodes = 1 )
```

Arguments

files	a character vector containing the names of the LSD results files which the data are to be read from. If they do not contain an absolute path, the file names are relative to the current working directory, getwd(). These can be compressed files and must include the appropriated extension (usually .res or .res.gz).
col.names	a vector of optional names for the variables. The default is to read all variables.
nrows	integer: the maximum number of time steps (rows) to read in. Negative and other invalid values are ignored. The default is to read all rows.
skip	integer: the number of time steps (rows) of the results file to skip before beginning to read data. The default is to read from the first time step (t = 1).
check.names	logical. If TRUE then the names of the variables are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by make.names) so that they are, and also to ensure that there are no duplicates.
pool	logical. If TRUE, variables instances from all files are concatenated (by columns) as a single file. If FALSE (the default), each file is saved as a separated dimension (fourth) in the array.
nnodes	integer: the maximum number of parallel computing nodes (parallel threads) in the current computer to be used for reading the files. The default, nnodes = 1, means single thread processing (no parallel threads). If equal to zero, creates up to one node per CPU core. Only PSOCK clusters are used, to ensure compatibility with any platform. Please note that each node requires its own memory space, so memory usage increases linearly with the number of nodes.

Value

Returns a 4D array containing data series for each instance from the selected variables.

Note

If the selected files don't have the same columns available (names), an error is produced. When 'pool = TRUE', the produced array is still 4-dimensional but the fourth dimension has just one value (= 1). Pooling require that all files contains EXACTLY the same variables (number of instances may be different).

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() read.3d.lsd(), read.single.lsd(), read.multi.lsd(), read.list.lsd(), read.raw.lsd()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# read all instances of all variables from files,
allArray <- read.4d.lsd( files )
print( allArray[ 1 : 10, 1 : 7, 1, 1 ] ) # 1st instance of 1st file (7 vars and 10 times)
print( allArray[ 1 : 10, 9, , 2 ] ) # all instances of 9th variable in 2nd file (10 times)
print( allArray[ 50, 8, , ] ) # all instances of all files of 8th variable for t=50

# the same, but pooling all files into a single one
allArrayPool <- read.4d.lsd( files, pool = TRUE )
print( allArrayPool[ 1 : 10, 1 : 7, 1, 1 ] ) # 1st instance of 1st file (7 vars and 10 times)
print( allArrayPool[ 1 : 10, 9, 4 : 9 , 1 ] ) # 6 instances of 9th variable in 2nd file (10 times)
print( allArrayPool[ 50, 8, 4 : 9, ] ) # 6 instances of all files of 8th variable for t=50

# read instances of a set of variables named '_A1p' and '_growth1'
abArray <- read.4d.lsd( files, c( "_A1p", "_growth1" ) )
print( abArray[ 1 : 10, , 1, 2 ] ) # 1st instances of 2nd file (all vars and 10 times)
print( abArray[ 1 : 10, 2, , 3 ] ) # all instances of 2nd variable in3rd file (10 times)
print( abArray[ 50, 1, , ] ) # all instances of all files of 1st variable for t=50

# read all variables/variables, skipping the initial 20 time steps
# and keeping up to 30 time steps (from t = 21 up to t = 30)
allArray21_30 <- read.4d.lsd( files, skip = 20, nrows = 30 )
print( allArray21_30[ , 9, , 2 ] ) # all instances of 9th variable in 2nd file
print( allArray21_30[ 10, 8, , ] ) # all instances of all files of 8th variable for t=30
```

`read.list.lsd` *Read one or all instances of LSD variables (time series) from a LSD results file into a list*

Description

This function reads the data series associated to a specific or all instances of each selected variable from a LSD results file (.res) and saves them into separated matrices (one per variable).

Usage

```
read.list.lsd( files, col.names = NULL, nrows = -1, skip = 0,
               check.names = TRUE, instance = 0, pool = FALSE, nnodes = 1 )
```

Arguments

`files`	a character vector containing the names of the LSD results files which the data are to be read from. If they do not contain an absolute path, the file names are relative to the current working directory, `getwd()`. These can be compressed files and must include the appropriated extension (usually `.res` or `.res.gz`).
`col.names`	a vector of optional names for the variables. The default is to read all variables.

nrows	integer: the maximum number of time steps (rows) to read in. Negative and other invalid values are ignored. The default is to read all rows.
skip	integer: the number of time steps (rows) of the results file to skip before beginning to read data. The default is to read from the first time step (t = 1).
check.names	logical. If TRUE then the names of the variables are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by make.names) so that they are, and also to ensure that there are no duplicates.
instance	integer: the instance of the variable to be read, for variables that exist in more than one object. This number is based on the position (column) of the variable in the results file. The default (0) is to read all instances.
pool	logical. If TRUE, variables instances from all files are concatenated (by columns) into a single matrix. If FALSE (the default), each file is saved as a separated matrix.
nnodes	integer: the maximum number of parallel computing nodes (parallel threads) in the current computer to be used for reading the files. The default, nnodes = 1, means single thread processing (no parallel threads). If equal to zero, creates up to one node per CPU core. Only PSOCK clusters are used, to ensure compatibility with any platform. Please note that each node requires its own memory space, so memory usage increases linearly with the number of nodes.

Value

Returns a list containing matrices with the selected variables' time series in the results file. If pool = TRUE, the list contains a single, consolidated matrix (column names are not unique).

Note

When using the option 'pool = TRUE', columns from multiple files are consolidated with their original names, so names will not be unique anymore. You may use unique to change column names so they become unique, if required. The returned matrices may be potentially very wide, in particular if variables are not well selected(see col.names above) or if there is a large number of instances.

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() read.single.lsd(), read.multi.lsd(), read.3d.lsd(), read.4d.lsd(), read.raw.lsd(),

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# read all instances of all variables from three files (one matrix each),
```

```
tableList <- read.list.lsd( files )
print( tableList[[ 1 ]][ 1 : 5, 1 : 7 ] )
print( tableList[[ 2 ]][ 1 : 5, 1 : 7 ] )
print( tableList[[ 3 ]][ 1 : 5, 1 : 7 ] )

# read all instances of a set of variables named '_A1p' and '_growth1'
# and pool data
abTable <- read.list.lsd( files, c( "_A1p", "_growth1" ), pool = TRUE )
print( abTable[[ 1 ]][ 10 : 20, 5 : 9 ] )

# read instance 4 of all variables, skipping the initial 20 time steps
# and keeping up to 30 time steps (from t = 21 up to t = 30)
inst4List21_30 <- read.list.lsd( files, skip = 20, nrows = 30, instance = 4 )
print( inst4List21_30[[ 1 ]] )
print( inst4List21_30[[ 2 ]] )
```

`read.multi.lsd` *Read all instances of LSD variables (time series) from a LSD results file*

Description

This function reads the data series associated to all instances of each selected variable from a LSD results file (.res).

Usage

```
read.multi.lsd( file, col.names = NULL, nrows = -1, skip = 0, check.names = TRUE )
```

Arguments

`file`	the name of the LSD results file which the data are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory, `getwd()`. This can be a compressed file (see file) and must include the appropriated extension (usually `.res` or `.res.gz`).
`col.names`	a vector of optional names for the variables. The default is to read all variables.
`nrows`	integer: the maximum number of time steps (rows) to read in. Negative and other invalid values are ignored. The default is to read all rows.
`skip`	integer: the number of time steps (rows) of the results file to skip before beginning to read data. The default is to read from the first time step (t = 1).
`check.names`	logical. If TRUE then the names of the variables are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by make.names) so that they are, and also to ensure that there are no duplicates.

Value

Returns a list of matrices, each containing one of the selected variables' time series from the results file.

Note

For extracting data from multiple similar files (like sensitivity analysis results), see `read.list.lsd`.

Author(s)

Marcelo C. Pereira

See Also

`list.files.lsd()` `read.single.lsd()`, `read.list.lsd()`, `read.3d.lsd()`, `read.4d.lsd()`, `read.raw.lsd()`

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# load first .res file into a simple matrix (all instances),
macroList <- read.multi.lsd( files[ 1 ] )
length( macroList )
print( macroList[[ 1 ]][ 1 : 5, ] )
print( macroList[[ 8 ]][ 1 : 5, ] )

# read first instance of 2 variables, skipping the initial 20 time steps
# and keeping up to 30 time steps (from t = 21 up to t = 30)
varsList21_30 <- read.multi.lsd( files[ 2 ], c( "_A1p", "_growth1" ),
                                 skip = 20, nrows = 30 )
print( varsList21_30[[ 1 ]][ , 1 : 3 ] )
print( varsList21_30[[ 2 ]][ , 1 : 3 ] )
```

`read.raw.lsd` *Read LSD results file and clean variables names*

Description

This function reads all the data series in a LSD results file (.res).

Usage

```
read.raw.lsd( file, nrows = -1, skip = 0 )
```

Arguments

`file`	the name of the LSD results file which the data are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory, `getwd()`. This can be a compressed file (see file) and must include the appropriated extension (usually `.res` or `.res.gz`).

nrows	integer: the maximum number of time steps (rows) to read in. Negative and other invalid values are ignored. The default is to read all rows.
skip	integer: the number of time steps (rows) of the results file to skip before beginning to read data. The default is to read from the first time step (t = 1).

Value

Returns a single matrix containing all variables' time series contained in the results file.

Note

The returned matrix may be potentially very wide. For extracting data in a more selective way, see read.single.lsd and read.multi.lsd. To use multiple results files simultaneously, see read.list.lsd and read.3d.lsd. Variable names are never "cleaned", even for single instanced variables.

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() read.single.lsd(), read.multi.lsd(), read.list.lsd(), read.3d.lsd(), read.4d.lsd()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# read all instances of all variables of first file,
bigTable <- read.raw.lsd( files[ 1 ] )
print( bigTable[ 1 : 5, 1 : 7 ] )

# read all instances of all variables, skipping the initial 20 time steps
# and keeping up to 30 time steps (from t = 21 up to t = 30)
all21_30 <- read.raw.lsd( files[ 2 ], skip = 20, nrows = 30 )
print( all21_30[ , 1 : 7 ] )
```

---

read.single.lsd *Read LSD variables (time series) from a LSD results file (a single instance of each variable only)*

Description

This function reads the data series associated to one instance of each selected variable from a LSD results file (.res). Just a single instance (time series of a single LSD object) is read at each call.

Usage

```
read.single.lsd( file, col.names = NULL, nrows = -1, skip = 0,
                 check.names = TRUE, instance = 1 )
```

Arguments

file	the name of the LSD results file which the data are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory, getwd(). This can be a compressed file (see file) and must include the appropriated extension (usually .res or .res.gz).
col.names	a vector of optional names for the variables. The default is to read all (unique) variables.
nrows	integer: the maximum number of time steps (rows) to read in. Negative and other invalid values are ignored. The default is to read all rows.
skip	integer: the number of time steps (rows) of the results file to skip before beginning to read data. The default is to read from the first time step (t = 1).
check.names	logical. If TRUE then the names of the variables are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by make.names) so that they are, and also to ensure that there are no duplicates.
instance	integer: the instance of the variable to be read, for variables that exist in more than one object. This number is based on the position (column) of the variable in the results file. The default is to read the first instance.

Value

Returns a matrix containing the selected variables' time series contained in the results file.

Note

This function is useful to extract time series for variables that are single instanced, like summary statistics. For multi-instanced variables, see read.multi.lsd. For extracting data from multiple similar files (like sensitivity analysis results), see read.list.lsd (multi-instanced variables) and read.3d.lsd (single-instanced variables).

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() read.multi.lsd(), read.list.lsd(), read.3d.lsd(), read.4d.lsd(), read.raw.lsd()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# load first .res file into a simple matrix (first instances only)
```

```
macroVar <- read.single.lsd( files[ 1 ] )
print( macroVar[ 1:10, ] )

# read second instance of a set of variables named '_A1p' and '_growth1'
ag2Table <- read.single.lsd( files[ 2 ], c( "_A1p", "_growth1" ), instance = 2 )
print( ag2Table[ 10:15, ] )

# read first instance of all variables, skipping the initial 20 time steps
# and keeping up to 30 time steps (from t = 21 up to t = 30)
var21_30 <- read.single.lsd( files[ 3 ], skip = 20, nrows = 30 )
print( var21_30[ , 1 : 7 ] )
```

`select.colattrs.lsd` *Select a subset of a LSD results matrix (by variable attributes)*

Description

This function select a subset of a LSD results matrix (as produced by `read.raw.lsd`) by the variable attributes, considering the LSD object position and the time span.

Usage

```
select.colattrs.lsd( dataSet, info, col.names = NULL, posit = NULL,
                     init.value = NA, init.time = NA, end.time = NA )
```

Arguments

`dataSet`	matrix produced by the invocation of `read.raw.lsd`, `read.single.lsd`, `read.multi.lsd` or `read.list.lsd` (a single matrix a time) functions.
`info`	data frame produced by `info.details.lsd` for the same results file from where `dataSet` was extracted.
`col.names`	a vector of optional names for the variables to select from. The default is to select from all variables.
`posit`	a string, a vector of strings or an integer vector describing the LSD object position of the variable(s) to select. If a string or an integer vector, it should define the position of a SINGLE LSD object. If a vector of strings, each element of the vector should define a different LSD object, so the returning matrix will contain variables from more than one object.
`init.value`	initial value attributed to the variable(s) to select.
`init.time`	initial time attributed to the variable(s) to select.
`end.time`	end time attributed to the variable(s) to select.

Details

Selection restriction parameters can be provided as needed; when not specified, each selection dimension include all available cases.

Value

Returns a single matrix containing the selected variables' time series contained in the original data set.

Note

If only variable names selection is needed, `select.colnames.lsd` is more efficient because information preprocessing (`info.details.lsd`) is not required.

Author(s)

Marcelo C. Pereira

See Also

`list.files.lsd`() `info.details.lsd`(), `select.colnames.lsd`()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# read all instances of all variables of first file
bigTable <- read.raw.lsd( files[ 1 ] )

# build the info table
info <- info.details.lsd( files[ 1 ] )

# read some instances of a set of variables named '_A1p' and '_growth1'
abFirst2 <- select.colattrs.lsd( bigTable, info, c( "_A1p", "_growth1" ),
                                 posit = c( "1_2", "1_5" ) )
print( abFirst2[ 50 : 60, ] )

# read instances of variable '_A1p' that start at time step t = 1
a50 <- select.colattrs.lsd( bigTable, info, make.names( "_A1p" ), init.time = 1 )
print( a50[ 1 : 10, ] )
```

`select.colnames.lsd` *Select a subset of a LSD results matrix (by column/variable names)*

Description

This function select a subset of a LSD results matrix (as produced by `read.raw.lsd`) by the column (variable) names, considering only the name part of the column labels.

Usage

```
select.colnames.lsd( dataSet, col.names, instance = 0 )
```

Arguments

dataSet	matrix produced by the invocation of read.raw.lsd, read.single.lsd, read.multi.lsd or read.list.lsd (a single matrix a time) functions.
col.names	a vector of optional names for the variables. The default is to read all variables. The names must to be in R format.
instance	integer: the instance of the variable to be read, for variables that exist in more than one object. This number is based on the position (column) of the variable in the results file. The default (0) is to read all instances.

Value

Returns a single matrix containing the selected variables' time series contained in the original data set.

Note

The variable/column names must be valid R column names (e.g., names do not start with a underscore). Use make.names if required.

Author(s)

Marcelo C. Pereira

See Also

list.files.lsd() select.colattrs.lsd(), make.names()

Examples

```
# get the list of file names of example LSD results
files <- list.files.lsd( system.file( "extdata", package = "LSDinterface" ) )

# read all instances of all variables in first file
bigTable <- read.raw.lsd( files[ 1 ] )
print( bigTable[ 1 : 10, 1 : 7 ] )

# extract all instances of a set of variables named '_A1p' and '_growth1'
abTable <- select.colnames.lsd( bigTable, make.names( c( "_A1p", "_growth1" ) ) )
print( abTable[ 20, ] )
```

Index