Package ‘LSDinterface’

February 9, 2018

Type Package

Title Reading LSD Results (.res) Files

Version 0.4.0

Date 2018-2-7

Author Marcelo C. Pereira

Maintainer Marcelo C. Pereira <marcelocpereira@uol.com.br>

Description Interfaces R with LSD. Reads object-oriented data in results files (.res) produced by LSD and creates appropriate multi-dimensional arrays in R. Supports multiple core parallelization 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 <http://labsimdev.org>).

Depends R (>= 3.2.0)

Imports stats, utils, abind, parallel

License GPL-3

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-09 20:01:38 UTC

R topics documented:

`LSDinterface-package` *Reading LSD Results (.res) Files*

Description

Interfaces R with LSD. Reads object-oriented data in results files (.res) produced by LSD and creates appropriate multi-dimensional arrays in R. Supports multiple core parallelization 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 <http://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).

`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)

Marcelo C. Pereira

Maintainer: Marcelo C. Pereira <marcelocpereira@uol.com.br>

References

LSD documentation is available at `http://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

`info.init.lsd`, `info.names.lsd` `info.dimensions.lsd`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

info1 <- info.details.lsd( paste0( path, "/", "Sim1_1.res" ) )
View( info1 )

info2 <- info.details.lsd( paste0( path, "/", "Sim1_2.res.gz" ) )
View( info2 )
```

`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

`info.details.lsd`, `info.names.lsd`, `info.init.lsd`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

info.dimensions.lsd( paste0( path, "/", "Sim1_1.res" ) )
info.dimensions.lsd( paste0( path, "/", "Sim1_2.res.gz" ) )
```

`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

`info.details.lsd`, `info.names.lsd` `info.dimensions.lsd`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

init1 <- info.init.lsd( paste0( path, "/", "Sim1_1.res" ) )
View( init1 )

init2 <- info.init.lsd( paste0( path, "/", "Sim1_2.res.gz" ) )
View( init2 )
```

`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

`info.details.lsd`, `info.init.lsd` `info.dimensions.lsd`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

info.names.lsd( paste0( path, "/", "Sim1_1.res" ) )

info.names.lsd( paste0( path, "/", "Sim1_2.res.gz" ) )
```

`info.stats.lsd` *Compute Monce 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 average, the standard deviation, the maximum and the mininum for each variable, at each time step.

Usage

```
info.stats.lsd(array, rows = 1, cols = 2)
```

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`	integer: array dimension to be used as the rows for the statistics matrices, default is to use first array dimension.
`cols`	integer: array dimension to be used as the columns for the statistics matrices, default is to use second array dimension.

Value

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

`avg`	a matrix with the average 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

Author(s)

Marcelo C. Pereira

See Also

read.3d.lsd, read.4d.lsd, info.dimensions.lsd

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# reads first instance of all variables from three MC files (3D array)
inst1Array <- read.3d.lsd( c( paste0( path, "/", "Sim1_1.res" ),
                              paste0( path, "/", "Sim1_2.res" ),
                              paste0( path, "/", "Sim1_3.res" ) ) )

# creates statistics data frames for the variables
inst1Stats <- info.stats.lsd( inst1Array )

# See matrix in the data viewer (require package 'utils')
View( inst1Stats )

# organize the stats by variable (dim=2) and file (dim=3)
inst1Stats2 <- info.stats.lsd( inst1Array, rows = 2, cols = 3 )
View( inst1Stats2 )

# the same but for all instance of all variables (from a 4D array)
allArray <- read.4d.lsd( c( paste0( path, "/", "Sim1_1.res" ),
                            paste0( path, "/", "Sim1_2.res" ),
                            paste0( path, "/", "Sim1_3.res" ) ) )
allStats <- info.stats.lsd( allArray )
View( allStats )

# organize the stats by file (dim=4) and variable (dim=2)
allStats2 <- info.stats.lsd( allArray, rows = 4, cols = 2 )
View( allStats2 )
```

---

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

`info.names.lsd`,

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

name.check.lsd( paste0( path, "/", "Sim1_1.res" ) )

name.check.lsd( paste0( path, "/", "Sim1_1.res" ),
                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 parallelization). 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

`read.4d.lsd`, `read.single.lsd`, `read.multi.lsd`, `read.list.lsd`, `read.raw.lsd`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# reads first instance of all variables from three files (one level each),
# pasting the directory where the example files are (not required if in working dir)
inst1Array <- read.3d.lsd( c( paste0( path, "/", "Sim1_1.res" ),
                              paste0( path, "/", "Sim1_2.res" ),
                              paste0( path, "/", "Sim1_3.res" ) ) )

# See parts of the 3D array in the data viewer (require package 'utils')

View( inst1Array[ , , 1 ] )
View( inst1Array[ , , 2 ] )
View( inst1Array[ , , 3 ] )


# read first instance of a set of variables named '_A1p' and '_growth1'
ab1Array <- read.3d.lsd( c( paste0( path, "/", "Sim1_1.res.gz" ),
                            paste0( path, "/", "Sim1_2.res.gz" ),
                            paste0( path, "/", "Sim1_3.res.gz" ) ),
                         c( "_A1p", "_growth1" ) )

View( ab1Array[ , , 1 ] )
View( ab1Array[ , , 2 ] )
View( ab1Array[ , , 3 ] )


# reads instance 2 of all variables, skipping the initial 20 time steps
# and keeping up to 50 time steps (from t = 21 up to t = 70)
inst2Array21_70 <- read.3d.lsd( c( paste0( path, "/", "Sim1_1.res" ),
                                   paste0( path, "/", "Sim1_2.res" ) ),
                                skip = 20, nrows = 50, instance = 2 )

View( inst2Array21_70[ , , 1 ] )
View( inst2Array21_70[ , , 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 colums) as a single file. If FALSE (the default), each file is saved as a separated dimension (4th) 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 parallelization). 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

`read.3d.lsd`, `read.single.lsd`, `read.multi.lsd`, `read.list.lsd`, `read.raw.lsd`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# reads all instances of all variables from three files,
# pasting the directory where the example files are (not required if in working dir)
allArray <- read.4d.lsd( c( paste0( path, "/", "Sim1_1.res" ),
                            paste0( path, "/", "Sim1_2.res" ),
                            paste0( path, "/", "Sim1_3.res" ) ) )

# See parts of the 4D array in the data viewer (require package 'utils')

View( allArray[ , , 1, 1 ] )   # first instance of first file (all vars and times)
View( allArray[ , 9, , 2 ] )  # all instances of ninth variable in second file (all t's)
View( 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( c( paste0( path, "/", "Sim1_1.res" ),
                                paste0( path, "/", "Sim1_2.res" ),
                                paste0( path, "/", "Sim1_3.res" ) ),
                             pool = TRUE )

View( allArrayPool[ , , 1, 1 ] )   # first instance of first file (all vars and times)
View( allArrayPool[ , 9, , 1 ] )  # all instances of ninth variable in second file (all t's)
View( allArrayPool[ 50, 8, , ] )   # all 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( c( paste0( path, "/", "Sim1_1.res.gz" ),
                           paste0( path, "/", "Sim1_2.res.gz" ),
                           paste0( path, "/", "Sim1_3.res.gz" ) ),
                        c( "_A1p", "_growth1" ) )

View( abArray[ , , 1, 2 ] )   # first instances of second file (all vars and times)
View( abArray[ , 2, , 3 ] )   # all instances of second variable in third file (all t's)
View( abArray[ 50, 1, , ] )   # all instances of all files of first variable for t=50
```

```
# reads all variables/variables, skipping the initial 20 time steps
# and keeping up to 50 time steps (from t = 21 up to t = 70)
allArray21_70 <- read.4d.lsd( c( paste0( path, "/", "Sim1_1.res" ),
                                 paste0( path, "/", "Sim1_2.res" ) ),
                                 skip = 20, nrows = 50 )

View( allArray21_70[ , 9, , 2 ] )   # all instances of ninth variable in second file
View( allArray21_70[ 30, 8, , ] )   # all instances of all files of 8th variable for t=50
```

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 colums) 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 parallelization). 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

`read.single.lsd`, `read.multi.lsd`, `read.3d.lsd`, `read.4d.lsd`, `read.raw.lsd`, `unique`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# reads all instances of all variables from three files (one matrix each),
# pasting the directory where the example files are (not required if in working dir)
tableList <- read.list.lsd( c( paste0( path, "/", "Sim1_1.res" ),
                               paste0( path, "/", "Sim1_2.res" ),
                               paste0( path, "/", "Sim1_3.res" ) ) )

# See parts of the 3D array in the data viewer (require package 'utils')

View( tableList[[ 1 ]] )
View( tableList[[ 2 ]] )
View( tableList[[ 3 ]] )


# read all instances of a set of variables named '_A1p' and '_growth1'
# and pool data
abTable <- read.list.lsd( c( paste0( path, "/", "Sim1_1.res.gz" ),
                             paste0( path, "/", "Sim1_2.res.gz" ) ),
                          c( "_A1p", "_growth1" ), pool = TRUE )
```

```
View( abTable )

# reads instance 4 of all variables, skipping the initial 20 time steps
# and keeping up to 50 time steps (from t = 21 up to t = 70)
inst4List21_70 <- read.list.lsd( c( paste0( path, "/", "Sim1_1.res" ),
                                    paste0( path, "/", "Sim1_2.res" ) ),
                                 skip = 20, nrows = 50, instance = 4 )

View( inst4List21_70[[ 1 ]] )
View( inst4List21_70[[ 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

read.single.lsd, read.list.lsd, read.3d.lsd, read.4d.lsd, read.raw.lsd

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# Load a sample .res file into a simple matrix (all instances),
# pasting the directory where the example files are (not required if in working dir)
macroList <- read.multi.lsd( paste0( path, "/", "Sim1_1.res" ) )

# See matrix in the data viewer (require package 'utils')
length( macroList )

View( macroList[[ 1 ]] )
View( macroList[[ 8 ]] )


# reads first instance of 2 variables, skipping the initial 20 time steps
# and keeping up to 50 time steps (from t = 21 up to t = 70)
varsList21_70 <- read.multi.lsd( paste0( path, "/", "Sim1_1.res" ),
                                 c( "_A1p", "_growth1" ),
                                 skip = 20, nrows = 50 )

View( varsList21_70[[ 1 ]] )
View( varsList21_70[[ 2 ]] )
```

`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

`read.single.lsd`, `read.multi.lsd`, `read.list.lsd`, `read.3d.lsd`, `read.4d.lsd`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# reads all instances of all variables,
# pasting the directory where the example files are (not required if in working dir)
bigTable <- read.raw.lsd( paste0( path, "/", "Sim1_1.res" ) )

# See matrix in the data viewer (require package 'utils')
View( bigTable )

# reads all instances of all variables, skipping the initial 20 time steps
# and keeping up to 50 time steps (from t = 21 up to t = 70)
all21_70 <- read.raw.lsd( paste0( path, "/", "Sim1_2.res.gz" ),
                          skip = 20, nrows = 50 )
View( all21_70 )
```

`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

read.multi.lsd, read.list.lsd, read.3d.lsd, read.4d.lsd, read.raw.lsd

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# Load a sample .res file into a simple matrix (first instances only)
macroVar <- read.single.lsd( paste0( path, "/", "Sim1_1.res" ) )

# See matrix in the data viewer (require package 'utils')
View( macroVar )

# read second instance of a set of variables named '_A1p' and '_growth1'
ag2Table <- read.single.lsd( paste0( path, "/", "Sim1_2.res" ),
                             c( "_A1p", "_growth1" ), instance = 2 )
View( ag2Table )

# reads first instance of all variables, skipping the initial 20 time steps
# and keeping up to 50 time steps (from t = 21 up to t = 70)
var21_70 <- read.single.lsd( paste0( path, "/", "Sim1_1.res" ),
                             skip = 20, nrows = 50 )
View( var21_70 )
```

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 = NA, 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 pre-processing (`info.details.lsd`) is not required.

Author(s)

Marcelo C. Pereira

See Also

`info.details.lsd`, `select.colnames.lsd`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# reads all instances of all variables
bigTable <- read.raw.lsd( paste0( path, "/", "Sim1_1.res" ) )

# build the info table
info <- info.details.lsd( paste0( path, "/", "Sim1_1.res" ) )

# 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" ) )

# See matrix in the data viewer (require package 'utils')
View( abFirst2 )
```

```
# reads instances of variable '_A1p' that start at time step t = 1
a50 <- select.colattrs.lsd( bigTable, info, make.names( "_A1p" ), init.time = 1 )
View( a50 )
```

`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

`select.colattrs.lsd`, `make.names`

Examples

```
# Get the examples directory
path <- system.file( "extdata", package = "LSDinterface" )

# reads all instances of all variables
bigTable <- read.raw.lsd( paste0( path, "/", "Sim1_1.res" ) )

# See matrix in the data viewer (require package 'utils')
View( bigTable )

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

Index