Package 'MadingleyR'

March 4, 2021

Type Package

Title MadingleyR: General Ecosystem Model

Version 1.0.1

Author xxx xxx

Maintainer The package maintainer <xxx@xxx.comt>

Description The MadingleyR package is a wrapper around the C++ code of the Madingley model. It was intended as an accessible and transparent tool to lower the barrier of using the Madingley model, thereby broadening the Madingley comminity.

License GPL (¿=3)

Encoding UTF-8

LazyData true

Imports sp, raster, rgdal, data.table

R topics documented:

list_output_paths *MadingleyR: Get the output file paths*

Description

This function returns the file names and/or paths of the files written by the MadingleyR package. It only works when an output folder is specified in the madingley_run() function. This aids the reading of files for processing at a later stage (see example below).

Usage

```
list_output_paths(madingley_data, full_path = TRUE)
```

Arguments

madingley_data	output generated by madingley_run()
full_path	whether the function should return the full path (full_path = TRUE) or the names of the files only (full_path = FALSE)

Examples

```
# Set output folder
out_dir = 'path/to/outputs'

# Set spatial window
spatial_window = c(31, 35, -5, -1)

# Call the madingley_init function (default run)
m_data = madingley_init(spatial_window = spatial_window)

# Run the model
m_data2 = madingley_run(out_dir = out_dir,
                        madingley_data = m_data,
                        years = 4)

# Get the full paths of the output files
output_paths = list_output_paths(madingley_data = m_data2)

# Get names of the output files
output_names = list_output_paths(madingley_data = m_data2, full_path = F)

# Use output_paths to load a file written
cat(output_paths$cohort_properties[1])
BinnedCohortStatistics1 = read.csv(output_paths$cohort_properties[1])
head(BinnedCohortStatistics1)
```

madingley_compile *MadingleyR: Compiles the source code on Linux*

Description

This function compiles the source code for Linux. Usually the binary supplied with the package works on Linux systems. However, this function can be used to compile the source code and installed it into the pacakge library. This function is also useful when making alteration to the source code. The compiled binary file can be found within the specified source folder (labelled with the current date).

Usage

```
# Compile the source code
madingley_compile(madingley_path = "~/Documents/MadingleyR-master/SourceCode/madingley", install_new

# The source code can be downloaded using
src_path = madingley_download_source("~/Documents")

# src_path can be used in madingley_compile()
madingley_compile(madingley_path = src_path)
```

Arguments

madingley_path	The path of the madingley source code files (can be obtained using madingley download)
install_new_binary	a bool (TRUE or FALSE) indicating if the compiled source needs to be installed directly into the pacakge library (TRUE) or not (FALSE). By default this is set to FALSE.

Examples

```
# The source code can be downloaded using
src_path = madingley_download_source("~/Documents")

# Compile source code, src_path can be used in madingley_compile()
madingley_compile(madingley_path = src_path)
```

madingley_download_source *MadingleyR: Downloads the MadingleyR source code*

Description

This function downloads and extracts the MadingleyR source code (zip) into the specified folder.

Usage

```
# The source code can be downloaded using
src_path = madingley_download_source(madingley_path = "~/Documents")

# src_path and the downloaded files can be used in madingley_compile()
madingley_compile(madingley_path = src_path)
```

Arguments

madingley_path	The download folder

Examples

```
# The source code can be downloaded using
src_path = madingley_download_source(madingley_path = "~/Documents")

# Compile source code, src_path can be used in madingley_compile()
madingley_compile(madingley_path = src_path)
```

---

madingley_init *MadingleyR: Initialise the Madingley model*

Description

This function generates the cohort and stock data.frames (initialisation parameters) needed to run madingley_run().

Usage

```
# Simple default init
mdata = madingley_init(spatial_window)
# All init parameters manually
mdata = madingley_init(spatial_window, cohort_def, stock_def, spatial_input, max_cohorts)
```

Arguments

spatial_window	a vector holding coordinates that specify the spatial model domain, specified in the following order: minimum longitude, maximum longitude, minimum latitude, maximum latitude. For example spatial_window = c(31,35,-5,-1)
cohort_def	a data.frame holding the cohort definitions used for initialising the model. The template or default data.frame can be loaded using: madingley_inputs('cohort definition'). If the cohort_def is not specified, madingley_init() loads the cohort definitions internally using the madingley_inputs('cohort definition') function call

stock_def — a data.frame holding the stock definitions used for initialising the model. The template or default data.frame can be loaded using: madingley_inputs('stock definition'). If the stock_def is not specified, madingley_init() loads the stock definitions internally using the madingley_inputs('stock definition') function call

spatial_input — an R list that holds all of the spatial inputs required for initialising the Madingley model. The default spatial inputs can be loaded using: madingley_inputs('spatial inputs'). If the spatial_input is not specified, madingley_init() loads the inputs internally using the madingley_inputs("spatial inputs') function call

max_cohort — a single integer holding the maximum number of cohorts allowed per spatial grid cell. The Madingley model merges cohorts if they are highly similar and the maximum number of cohorts is exceeded. This is done for computational reasons (keeping the number of interactions at a reasonable level). The Madingley model merges cohorts if they are highly similar and the maximum number of cohorts is exceeded. Lowering this value will increase model performance at expenses of resolution of cohort diversity. max_cohors is normally set to 1000, but 500 may be sufficient for preliminary trials. Please note, that setting this value to e.g. 500 in madingley_init() makes sure that max 500 cohorts per grid cell are created. If in a consecutive madingley_run() max_cohort is set to e.g. 1000 the maximum number of cohorts (500 cohorts/grid cell) will increase during the simulation run until the new maximum (1000 cohorts/grid cell) is reached. For consistency it may be preferred to set max_cohort to the same number in a madingley_init() call and the consecutive madingley_run() simulation.

Examples

```
# Set spatial model domain
spatial_window = c(31, 35, -5, -1)

# Call the madingley_init function
m_data = madingley_init(spatial_window = spatial_window)

# Load objects needed for a non-default init
spatial_window = c(31, 35, -5, -1)
sptl_inp = madingley_inputs('spatial inputs')
chrt_def = madingley_inputs('cohort definition')
stck_def = madingley_inputs('stock definition')
mdl_prms = madingley_inputs('model parameters')

# Call the madingley_init function with manual inputs
m_data = madingley_init(cohort_def = chrt_def,
                        stock_def = stck_def, spatial_inputs = sptl_inp,
                        spatial_window =s patial_window, max_cohort = 200)
```

madingley_inputs *MadingleyR: Load the default Madingley inputs*

Description

This function loads the default MadingleyR input parameters in case the model parameters need to be checked or modified depending on the simulation scenario. The function takes one string input. The available options can be printed to the R console with madingley_inputs( ), see example code below.

Usage

```
madingley_inputs() # prints all available options
sptl_inp = madingley_inputs("spatial inputs") # loads the default spatial inputs
chrt_def = madingley_inputs("cohort definition") # loads the default cohort definitions
stck_def = madingley_inputs("stock definition") # loads the default stock definitions
mdl_prms = madingley_inputs("model parameters") # loads the default model parameters
```

Arguments

input_type	a string specifying the input type to load. The options are: "spatial inputs", "cohort definition", "stock definition" or "model parameters". Running the function without any string will print all available options as shown in the usage above or example below.

Examples

```
# Print all the available options to load to the R console
madingley_inputs( )

# Load spatial parameters
sptl_inp = madingley_inputs(input_type = "spatial inputs")

# Load default cohort property definitions
chrt_def = madingley_inputs(input_type = "cohort definition")

# Load default stock property definitions
stck_def = madingley_inputs(input_type = "stock definition")

# Load default model parameters
mdl_prms = madingley_inputs(input_type = "model parameters")

# Set spatial model domain
spatial_window = c(31, 35, -5, -1)

## ----------------------------##
# Change parameters accordingly #
##----------------------------##
```

```
# Call the madingley_init function
m_data = madingley_init(cohort_def = chrt_def,
                        stock_def = stck_def,
                        spatial_inputs = sptl_inp,
                        max_cohort = 500)

# Run the madingley model
m_data2 = madingley_run(madingley_data = m_data,
                        years = 5,
                        max_cohort = 500,
                        cohort_def = chrt_def,
                        stock_def = stck_def,
                        spatial_inputs = sptl_inp,
                        model_parameters = mdl_prms)
```

`madingley_plot` *MadingleyR: Create MadingleyR plots*

Description

Create all output plots included in the MadinleyR package. This function is intended as a tool to quickly visualize the results. All resulting plots can also be made individually by using the following functions:

- plot_densities()
- plot_foodweb()
- plot_spatialabundances()
- plot_spatialbiomass()
- plot_timelines()
- plot_trophicpyramid()

Usage

```
madingley_plot(madingley_data)
```

Arguments

`madingley_data`	output generated by madingley_run()

Examples

```
# Print all the available options to load to the R console
madingley_inputs( )

# Load spatial parameters
sptl_inp = madingley_inputs("spatial inputs")

# Load default cohort property definitions
```

```
chrt_def = madingley_inputs("cohort definition")

# Load default stock property definitions
stck_def = madingley_inputs("stock definition")

# Load default model parameters
mdl_prms = madingley_inputs("model parameters")

# Set spatial model domain
spatial_window = c(31, 35, -5, -1)

# Call the madingley_init function
m_data = madingley_init(cohort_def = chrt_def,
                        stock_def = stck_def,
                        spatial_inputs = sptl_inp,
                        spatial_window = spatial_window,
                        max_cohort = 500)

# Run the madingley model using the m_data as input
m_data2 = madingley_run(madingley_data = m_data,
                        years = 5,
                        max_cohort = 500,
                        cohort_def = chrt_def,
                        stock_def = stck_def,
                        spatial_inputs = sptl_inp,
                        model_parameters = mdl_prms)

# Create the MadingleyR plots
madingley_plot(m_data2)
```

---

madingley_run *MadingleyR: Run the Madingley model*

Description

This function runs the Madingley model using outputs generated by madingley_init() or by a previous model run.

Usage

```
madingley_run(out_dir = tempdir(),
              madingley_data = m_data,
              years = 5,
              max_cohort = 500,
              cohort_def = chrt_def,
              stock_def = stck_def,
              spatial_inputs = sptl_inp,
              model_parameters = mdl_prms,
              dispersal_off = FALSE,
              silenced = FALSE,
              parallel = TRUE)
```

Arguments

out_dir	path to the folder used for writing the Madingley simulation outputs (csv file). If out_dir is not specified, madingley_run() writes the outputs to the temporary R folder. The outputs in this folder are required for the plotting functions included in MadingleyR. If the outputs need to be analysed manually or plots need to be made at a later time, please specify the output folder to make sure your files are stored
madingley_data	madingley output list generated by madingley_init() or by a previous model run (output list of madingley_run())
years	a single integer to set the simulation length (in years)
cohort_def	a data.frame holding the cohort definitions used for initialising the model. The template or default data.frame can be loaded using: madingley_inputs('cohort definition'). If the cohort_def is not specified, madingley_run() loads the cohort definitions internally using the madingley_inputs('cohort definition') function call
stock_def	a data.frame holding the stock definitions used for initialising the model. The template or default data.frame can be loaded using madingley_inputs('stock definition'). If the stock_def is not specified, madingley_run() loads the stock definitions internally using the madingley_inputs('stock definition') function call
spatial_input	an R list that holds all the spatial inputs required for initialising the Madingley model. The default spatial inputs can be loaded using madingley_inputs('spatial inputs'). If the spatial_input is not specified, madingley_run() loads the inputs internally using the madingley_inputs("spatial inputs') function call
model_parameters	a data.frame with the default model parameters. The function call madingley_inputs('model parameters') loads the model inputs alongside a description corresponding to each parameter. If the model_parameters argument is not specified, madingley_run() loads the default model parameters internally using the madingley_inputs('model parameters') function call
max_cohort	a single integer holding the maximum number of cohorts allowed per spatial grid cell for computational reasons (keeping the number of interactions at a reasonable level). The Madingley model merges cohorts if they are highly similar and the maximum number of cohorts is exceeded. Lowering this value will increase model performance at expenses of resolution of cohort diversity. max_cohort is normally set to 1000, but 500 may be sufficient for preliminary trials.
output_timestep	by default the model writes the outputs during all (12) months of the last simulation year (years - 1). This can be altered by using the output_timestep argument. The output_timestep argument is formatted as a vector containing four integers: 1) the year to output the log10 binned cohort properties 2) the year to output the full cohort properties 3) the year to output the log10 binned feeding interactions (required by the food-web plot) and 4) the year to output the full stock (autotroph) files.

Please note, exporting the results will always start from the year set and continue until the end of the simulation. For example, when setting output_timestep = c(5,3,5,4) and years = 5 in Madingley_run():

— the log10 binned cohort properties (set to 5) are never written (years = output_timestep = 5)

— the full cohort properties (set to 3) are written during the last two years (after the 3rd year until the end of the simulation)

— the log10 binned feeding interactions (set to 5) are never written (years = output_timestep = 5)

— the full stock files (set to 4) are written only during the last year of the simulation (after the 4th year).

Be careful outputting many large files, it can easily fill up your hard disk. Also note that outputting the feeding interactions significantly slows down the model runtime, only use this option if needed! The fact that outputting results will start from the year set and continue to the end of the simulation, can produce a lot of results when outputs are required somewhere halfway the simulation. Let's say in a 100-year simulation we need to output the model results at year = 50, we will end up with around 50 years of outputs (from year 50 to year 100). For these cases I recommend to run the model for 50 years, output the results at the final simulation year and continue the simulation for another 50 years using a new madingley_run() instance using the previous 50 year simulation as an input (see final example below).

`dispersal_off` If TRUE turns the dispersal off between grid cells in the Madingley model. By default dispersal_off is set to FALSE, thereby allowing dispersal between grid cells

`silenced` When set to TRUE it suppresses the outputs printed to the R console. By default it is set to FALSE

`parallel` If the simulation run should be executed in parallel (using all cores available) or in serial. By default it is set to TRUE. Naturally, a parallel model run significantly speeds up the execution time, but might make performing other tasks on the machine more difficult.

Examples

```
# Print all the available options to load to the R console
madingley_inputs( )
# Load spatial parameters
sptl_inp = madingley_inputs("spatial inputs")
# Load default cohort property definitions
chrt_def = madingley_inputs("cohort definition")
# Load default stock property definitions
stck_def = madingley_inputs("stock definition")
```

```
# Load default model parameters
mdl_prms = madingley_inputs("model parameters")
# Set spatial model domain
spatial_window = c(31, 35, -5, -1)
## ------------------------------------------##
# Change parameters accordingly               #
# e.g.: sptl_inp, chrt_def, stck_def, mdl_prms #
##------------------------------------------##
# Call the madingley_init function
m_data = madingley_init(cohort_def = chrt_def,
                        stock_def = stck_def,
                        spatial_inputs = sptl_inp,
                        spatial_window = spatial_window,
                        max_cohort = 500)
## ------------------------------------------------##
# Change parameters accordingly                     #
# e.g.: m_data sptl_inp, chrt_def, stck_def, mdl_prms #
##------------------------------------------------##
# Run the madingley model using the m_data as input
m_data2 = madingley_run(madingley_data = m_data,
                        years = 5,
                        max_cohort = 500,
                        cohort_def = chrt_def,
                        stock_def = stck_def,
                        spatial_inputs = sptl_inp,
                        model_parameters = mdl_prms)
## ------------------------------------------------##
# Change parameters accordingly                     #
# e.g.: m_data2 sptl_inp, chrt_def, stck_def, mdl_prms  #
##------------------------------------------------##
# Continue simulation run using the m_data2 as input
m_data3 = madingley_run(out_dir = "C:/MadingleyOutputs/",
                        madingley_data = m_data2,
                        years = 50,
                        max_cohort = 500,
                        cohort_def = chrt_def,
                        stock_def = stck_def,
                        spatial_inputs = sptl_inp,
                        model_parameters = mdl_prms)

## ------------------------------------------##
# Export outputs for a single year halfway the simulation duration                #
##------------------------------------------##
# Call the madingley_init function
m_data1 = madingley_init(spatial_window = spatial_window, max_cohort = 500)
# Run for 50 years and export results during last simulation year
m_data2 = madingley_run(out_dir = "C:/MadingleyOutputs/", madingley_data = m_data1,
        years = 50, max_cohort = 500, output_timestep = c(49,49,49,49))
# Run for an additional 50 years and export during last simulation year
m_data3 = madingley_run(out_dir = "C:/MadingleyOutputs/", madingley_data = m_data2,
        years = 50, max_cohort = 500, output_timestep = c(49,49,49,49))
```

plot_densities *MadingleyR: Plot abundance or biomass density against body mass*

Description

It plots the abundance or biomass density against (log10) body mass.

Usage

```
plot_densities(m_data2, weighted = 'biomass' , xlim=c(-4, 5))
plot_densities(m_data2, weighted = 'abundance' , xlim=c(-4, 5))
```

Arguments

madingley_data	Output list returned by madingley_run()
weighted	select either 'biomass' or 'abundance' (pass as string)
xlim	min and max log10 body mass plotted
ylim	min and max density
col	the colours used (length needs to be equal to the number of groups plotted, by default length = 6)
plot	If TRUE it generates the plot, if FALSE it returns the data output formatted for plotting which can be used for further analysis

Examples

```
# set model params
spatial_window = c(31,35,-5,-1) # Serengeti

# init model
m_data = madingley_init(spatial_window = spatial_window)

# run model
m_data2 = madingley_run(madingley_data = m_data, years = 3)

# make plots
plot_densities(m_data2, weighted = 'biomass' , xlim=c(-4, 5))
plot_densities(m_data2, weighted = 'abundance' , xlim=c(-4, 5))
```

plot_foodweb *MadingleyR: Make food web plot*

Description

It plots the log10-binned body mass food web.

Usage

```
plot_foodweb(m_data2, max_flows = 5, sample_data = 100, node_opacity = 0.5, col=c("#74add1","#a50026",'
```

Arguments

madingley_data	output list returned by madingley_run()
max_flows	set the maximum number of flows per node
colour	TRUE (colour version of the plot) or FALSE (grey scale version of the plot)
col	vector with node colours, defined as hex values, length needs to be 3 (default values used are col=c("#74add1","#a50026","#fdae61"))
node_opacity	the opacity of the plotted food-web nodes, from 0 to 1, default node_opacity = 0.5
sample_data	By default the function takes all data into consideration (sample_data = 100) but this can be computationally intensive. To speed up the visualization of the results it might be useful to only consider a random sample of the data, e.g. sample_data = 30 will sample 30% of the data randomly and use it to make the food-web plot

Examples

```
# set model params
spatial_window = c(31,35,-5,-1) # Serengeti

# init model
m_data = madingley_init(spatial_window = spatial_window)

# run model
m_data2 = madingley_run(madingley_data = m_data, years = 3)

# make plot
plot_foodweb(m_data2, max_flows = 5, sample_data = 100)
```

plot_spatialabundances *MadingleyR: Create spatial abundance plot*

Description

It plots the abundances of functional groups spatially.

Usage

```
plot_spatialabundances(m_data2, functional_filter = TRUE)
```

Arguments

	Input arguments
	madingley output list returned by madingley_run()
functional_filter	
madingley_data	TRUE or FALSE
plot	If set to TRUE it makes the plot, if set to FALSE the function returns a raster or raster brick (default = TRUE)
	Visual plotting arguments
box	TRUE or FALSE (default = TRUE)
axes	TRUE or FALSE (default = TRUE)
names	plot titles: by default it uses the numbering of the functional groups following the cohort definitions (see madingley_inputs('cohort definition'))
cex_legend_title	font size of legend title (default = 1)
cex_axis	font size of legend numbering (default = 1.3)
legend_mar	the legend margin (default = 20)
distance_legend_title	the distance from the legend to the legend title (default = 4)
legend_width	the width of the legend bar (default = 2.5)
col	colours used, defined as a vector, when not defined default colours used

Examples

```
# set model params
spatial_window = c(31,35,-5,-1) # Serengeti

# init model
m_data = madingley_init(spatial_window = spatial_window)

# run model
m_data2 = madingley_run(madingley_data = m_data, years = 3)
```

```
# make plot
plot_spatialabundances(m_data2, functional_filter = TRUE)
```

---

plot_spatialbiomass *MadingleyR: Create spatial biomass plot*

Description

It plots the biomass per grid cell of functional groups spatially.

Usage

```
plot_spatialbiomass(m_data2, functional_filter = TRUE)
```

Arguments

	Input arguments
	madingley output list returned by madingley_run()
functional_filter	
madingley_data	TRUE or FALSE
plot	If set to TRUE it makes the plot, if set to FALSE the function returns a raster or raster brick (default = TRUE)
	Visual plotting arguments
box	TRUE or FALSE (default = TRUE)
axes	TRUE or FALSE (default = TRUE)
names	plot titles: by default it uses the numbering of the functional groups following the cohort definitions (see madingley_inputs('cohort definition'))
cex_legend_title	font size of legend title (default = 1)
cex_axis	font size of legend numbering (default = 1.3)
legend_mar	the legend margin (default = 20)
distance_legend_title	the distance from the legend to the legend title (default = 4)
legend_width	the width of the legend bar (default = 2.5)
col	colours used, defined as a vector, when not defined default colours used

Examples

```
# set model params
spatial_window = c(31,35,-5,-1) # Serengeti

# init model
m_data = madingley_init(spatial_window = spatial_window)

# run model
m_data2 = madingley_run(madingley_data = m_data, years = 3)

# make plot
plot_spatialbiomass(m_data2, functional_filter = TRUE)
```

---

plot_timelines *MadingleyR: Create timeline plot*

Description

Creates a time series plot of the functional groups or the feeding guilds.

Usage

```
plot_timelines(m_data2, select = 'functional groups')
plot_timelines(m_data2, select = 'feeding guilds')
```

Arguments

	Input arguments
	madingley output list returned by madingley_run()
madingley_data	can be set to: 'functional groups' or 'feeding guilds'
select	
plot	If FALSE it returns a data.frame with the time line data formatted for plotting (default = TRUE).
col	colours used for the time lines, defined as vector, when not specified default colours used.

Examples

```
# set model params
spatial_window = c(31,35,-5,-1) # Serengeti

# init model
m_data = madingley_init(spatial_window = spatial_window)

# run model
m_data2 = madingley_run(madingley_data = m_data, years = 3)

# make plots
plot_timelines(m_data2, select = 'functional groups')
plot_timelines(m_data2, select = 'feeding guilds')
```

`plot_trophicpyramid` *MadingleyR: Creates trophic pyramide plot*

Description

The function plots a trophic pyramid from the Madingley simulation outputs.

Usage

```
plot_trophicpyramid(m_data2)
```

Arguments

`madingley_data`	madingley output list returned by madingley_run()
`flow_round`	number of decimals for rounding the quantification of flows (default = 2)
`mass_round`	number of decimals for rounding the quantification of mass (default = 2)
`col`	vector including the colours for each trophic level
`text_col`	the colour of the text quantifying the mass of each trophic level

Examples

```
# set model params
spatial_window = c(31,35,-5,-1) # Serengeti

# init model
m_data = madingley_init(spatial_window = spatial_window)

# run model
m_data2 = madingley_run(madingley_data = m_data, years = 3)

# make plot
plot_trophicpyramid(m_data2)
```