Package ‘rflsgen’

October 14, 2022

Type Package

Title Neutral Landscape Generator with Targets on Landscape Indices

Version 1.2.0

Author Dimitri Justeau-Allaire, Grégoire Blanchard, Thomas Ibanez, Xavier Lorca, Ghislain Vieilledent, Philippe Birnbaum

Maintainer Dimitri Justeau-Allaire <dimitri.justeau@gmail.com>

Description Interface to the ‘flsgen’ neutral landscape generator <https://github.com/dimitri-justeau/flsgen>. It allows to
- Generate fractal terrain;
- Generate landscape structures satisfying user targets over landscape indices;
- Generate landscape raster from landscape structures.

License GPL-3

Encoding UTF-8

Depends rJava, terra (>= 1.5-12), jsonlite

Imports rgdal, checkmate, utils

SystemRequirements Java (>= 8)

RoxygenNote 7.2.0

Suggests testthat (>= 3.0.0), knitr, rmarkdown, landscapemetrics

VignetteBuilder knitr


BugReports https://github.com/dimitri-justeau/rflsgen/issues

NeedsCompilation no

Repository CRAN

Date/Publication 2022-10-11 12:10:02 UTC
### flsgen_create_class_structure

`flsgen_create_class_structure` creates a predefined landscape class structure that can be converted as JSON input for `flsgen_generate`.

**Description**

Creates a predefined landscape class structure that can be converted as JSON input for `flsgen_generate`.

**Usage**

```r
flsgen_create_class_structure(class_name, patch_areas, is_square = FALSE)
```

**Format**

- `CLASS_LEVEL_TARGETS`: Vector of available class targets
- `flsgen_create_class_structure`: Creates a predefined landscape class structure that can be converted as JSON input for `flsgen_generate`.

---

**CLASS_LEVEL_TARGETS**: Vector of available class targets

**Description**

Vector of available class targets

**Usage**

```r
CLASS_LEVEL_TARGETS
```

**Format**

An object of class character of length 16.
**flsgen_create_class_targets**

**Description**

Creates a set of targets for a landscape class, which can be converted into JSON for flsgen.

**Usage**

```r
flsgen_create_class_targets(
  class_name,
  NP = NULL,
  AREA = NULL,
  AREA_MN = NULL,
  CA = NULL,
  PLAND = NULL,
  PD = NULL,
  SPI = NULL,
  LPI = NULL,
  MESH = NULL,
  SPLI = NULL,
  NPRO = NULL,
  SDEN = NULL,
  COHE = NULL,
  DIVI = NULL,
  IS_SQUARE = FALSE,
  ALL_DIFFERENT = FALSE
)
```

**Arguments**

- `class_name` Name of the class
- `patch_areas` Vector of patch areas
- `is_square` If true, all patches are required to be squares

**Value**

A landscape class structure

**Examples**

```r
## Not run:
cls_1 <- flsgen_class_structure("class 1", c(10, 100, 1000))
## End(Not run)
```
Arguments

class_name  Name of the class
NP          number of patches target (must be a vector of length 2)
AREA        patch area target (must be a vector of length 2)
AREA_MN     mean patch area target (must be a vector of length 2)
CA          total class area target (must be a vector of length 2)
PLAND       proportion of landscape target (must be a vector of length 2)
PD          patch density target (must be a vector of length 2)
SPI         smallest patch index target (must be a vector of length 2)
LPI         largest patch index target (must be a vector of length 2)
MESH        effective mesh size target (must be a vector of length 2)
SPLI        splitting index target (must be a vector of length 2)
NPRO        net product target (must be a vector of length 2)
SDEN        splitting density target (must be a vector of length 2)
COHE        degree of coherence target (must be a vector of length 2)
DIVI        degree of landscape division target (must be a vector of length 2)
IS_SQUARE   if TRUE, the class is required to only produce square patches
ALL_DIFFERENT if TRUE, the class is required to have differently sized patches

Details

Note that NP and AREA targets can be set as NULL, if the class targets is used within the ‘generate_series’ function to generate landscape series with varying NP and/or AREA. However, flsgen won’t run if NP and AREA are not set elsewhere.

Value

A class targets object which can be converted to JSON for flsgen

Examples

```r
## Not run:
c1s_1 <- flsgen_create_class_targets("class 1", NP=c(1, 10), AREA=c(0, 1000))
```

## End(Not run)
flsgen_create_landscape_structure

*Creating a predefined landscape structure that can be converted as JSON Input for flsgen generate*

## Description

Creates a predefined landscape structure that can be converted as JSON Input for flsgen generate.

## Usage

```r
flsgen_create_landscape_structure(
    nb_rows,
    nb_cols,
    classes,
    mask_raster = NULL
)
```

## Arguments

- **nb_rows**: Number of rows
- **nb_cols**: Number of columns
- **classes**: list of class structures
- **mask_raster**: mask raster (path or terra::rast object)

## Details

The class structures must be created prior to the call to this function.

Either `nb_rows` and `nb_cols`, or `mask_raster` must be specified. The dimensions of the landscape are deduced from the mask raster if it is used.

## Value

A landscape structure object which can be converted to JSON for flsgen generate.

## Examples

```r
## Not run:
cls_1 <- flsgen_class_structure("class 1", c(10, 100, 1000))
cls_2 <- flsgen_class_structure("class 2", c(20, 200, 2000))
ls_struct <- flsgen_landscape_structure(200, 200, list(cls_1, cls_2))

## End(Not run)
```
flsgen_create_landscape_targets

Creates a set of targets for a landscape

**Description**

Creates a set of targets for a landscape, which can be converted into JSON for flsgen.

**Usage**

```r
flsgen_create_landscape_targets(
  nb_rows,
  nb_cols,
  classes,
  mask_raster = NULL,
  NON_FOCAL_PLAND = NULL
)
```

**Arguments**

- `nb_rows`: Number of rows
- `nb_cols`: Number of columns
- `classes`: list of class targets
- `mask_raster`: mask raster (path or terra::rast object)
- `NON_FOCAL_PLAND`: PLAND (proportion of landscape) target on the non-focal land-use class

**Details**

The class targets must be created prior to the call to this function.

Either `nb_rows` and `nb_cols`, or `mask_raster` must be specified. The dimensions of the landscape are deduced from the mask raster if it is used.

**Value**

A landscape targets object which can be converted to JSON for flsgen

**Examples**

```r
## Not run:
cls_1 <- flsgen_create_class_targets("class 1", NP=c(1, 10), AREA=c(0, 1000))
cls_2 <- flsgen_create_class_targets("class 2", NP=c(1, 10), AREA=c(0, 1000))
ls_targets <- flsgen_create_landscape_targets(200, 200, list(cls_1, cls_2))
## End(Not run)
```
flsgen_create_target_series

From a base landscape target object, create a series of landscape targets, with one target for one class varying according to a specified sequence.

Description

Create a series of landscape targets, with one target for one class varying according to a specified sequence.

Usage

flsgen_create_target_series(
  landscape_targets,
  class_name = NULL,
  class_id = NULL,
  target_key,
  sequence
)

Arguments

landscape_targets  Number of rows
class_name          Name of the class for the varying target
class_id            Index of the class for the varying target
target_key          Varying target key
sequence            sequence (list) of targets for the varying target

Details

Either the class name or id must be given to identify the class to use for generating the series.

Value

A list of landscape targets

Examples

```r
## Not run:
cls_1 <- flsgen_create_class_targets("class 1", NP=c(1, 10), AREA=c(0, 1000))
cls_2 <- flsgen_create_class_targets("class 2", AREA=c(0, 1000))
ls_targets <- flsgen_create_landscape_targets(200, 200, list(cls_1, cls_2))
target_series <- flsgen_create_target_series(ls_targets, class_name="class 2",
  target_key="NP", sequence=seq(1, 10, by=1))

## End(Not run)
```
flsgen_extract_structure_from_raster

_Extracts a landscape structure from an existing raster_

Description

Extracts a landscape structure from an existing raster

Usage

```r
flsgen_extract_structure_from_raster(
  raster_file,
  focal_classes,
  connectivity = 4
)
```

Arguments

- `raster_file` terra::rast object or path of the raster
- `focal_classes` vector of integers representing the raster values of the focal classes to extract the structure from
- `connectivity` Connectivity definition in the regular square grid (4 or 8).

Value

A JSON landscape structure that can be used with flsgen generate

Examples

```r
## Not run:
ls_struct <- flsgen_extract_structure_from_raster(raster_path, c(0, 1, 2))

## End(Not run)
```

flsgen_generate

_Landscape raster generator_

Description

Generate landscape raster from landscape structure
Usage

```r
flsgen_generate(
  structure_str,
  structure_file,
  terrain_file = NULL,
  roughness = 0.5,
  terrain_dependency = 0.5,
  min_distance = 2,
  min_max_distance = NULL,
  connectivity = 4,
  x = 0,
  y = 0,
  resolution_x = 1e-04,
  resolution_y = NULL,
  epsg = "EPSG:4326",
  max_try = 2,
  max_try_patch = 10,
  verbose = TRUE
)
```

Arguments

- `structure_str`: JSON-formatted string describing the landscape structure to generate
- `structure_file`: JSON file containing the landscape structure to generate
- `terrain_file`: Path of input terrain raster file, or terra::rast object. If NULL a terrain is generated with the diamond-square algorithm
- `roughness`: Roughness factor (or H), between 0 and 1 (only need when terrain_file is NULL)
- `terrain_dependency`: Terrain dependency factor for landscape generation, between 0 and 1
- `min_distance`: Minimum distance between patches of a same class
- `min_max_distance`: If defined, the minimum distance between patches of a same class is defined by a variable buffer of width between min_distance and min_max_distance
- `connectivity`: Connectivity definition in the regular square grid (4 or 8).
- `x`: X position (geographical coordinates) of the top-left output raster pixel
- `y`: Y position (geographical coordinates) of the top-left output raster pixel
- `resolution_x`: X spatial resolution (geographical units) of the output raster (i.e. pixel width)
- `resolution_y`: Y-spatial resolution (geographical units) of the output raster (i.e. pixel height), if null, resolution_x is used
- `epsg`: EPSG identifier of the output projection
- `max_try`: Maximum number of trials for landscape generation
- `max_try_patch`: Maximum number of trials for patch generation
- `verbose`: if TRUE print information about generation
Details

The input landscape structure must be either specified as a JSON-formatted string (structure_str parameter) or as a JSON file (structure_file parameter)

Value

A terra::rast object

Examples

```r
## Not run:
json <- "{
  "nbRows" : 200,
  "nbCols" : 200,
  "classes" : [
    {
      "name" : "Class A",
      "NP" : [1, 10],
      "AREA" : [300, 4000],
      "CA" : [1000, 5000],
      "MESH" : [225, 225]
    },
    {
      "name" : "Class B",
      "NP" : [2, 8],
      "AREA" : [200, 4000],
      "PLAND" : [40, 40]
    },
    {
      "name" : "Class C",
      "NP" : [5, 7],
      "AREA" : [800, 1200]
    }
  ]
}
structure <- flsgen_structure(targets_str = json)
landscape <- flsgen_generate(structure_str = structure)

## End(Not run)
```
Usage

flsgen_structure(
    targets_str,
    targets_file,
    nb_solutions = 1,
    time_limit = 60,
    search_strategy = "DEFAULT"
)

Arguments

targets_str  JSON-formatted string describing user targets
targets_file JSON file describing user targets
nb_solutions Number of solutions to generate
time_limit   Time limit in seconds (if time_limit = 0, no time limit is set)
search_strategy  Choco solver search strategy (for more details refer to Choco solver documentation: https://choco-solver.org/docs/)

Details

The input user targets must be either specified as a JSON-formatted string (targets_str parameter) or as a JSON file (targets_file parameter).

Value

A vector of JSON-formatted landscape structures satisfying user targets.

Examples

```r
## Not run:
json <- "{
  "nbRows" : 200,
  "nbCols" : 200,
  [  
    {  
      "name" : "Class A",
      "NP" : [1, 10],
      "AREA" : [300, 4000],
      "CA" : [1000, 5000],
      "MESH" : [225, 225]
    },
    {  
      "name" : "Class B",
      "NP" : [2, 8],
      "AREA" : [200, 4000],
      "PLAND" : [40, 40]
    },
    {  
      "name" : "Class C",
```
flsgen_terrain

"NP" : [5, 7],
"AREA" : [800, 1200]
}
]
}"
structure <- flsgen_structure(targets_str = json)

## End(Not run)

flsgen_terrain  Fractal terrain generator

Description
Fractal terrain generation with the diamond-square algorithm

Usage
flsgen_terrain(
  width, height,
  roughness = 0.5,
  x = 0, y = 0,
  resolution = 1e-04,
  epsg = "EPSG:4326"
)

Arguments
width          Width (in pixels) of output raster
height         Height (in pixels) of output raster
roughness      Roughness factor (or H), between 0 and 1
x              X position (geographical coordinates) of the top-left output raster pixel
y              Y position (geographical coordinates) of the top-left output raster pixel
resolution     Spatial resolution (geographical units) of the output raster (i.e. pixel dimension)
epsg           EPSG identifier of the output projection

Value
A terra::rast object

Examples
## Not run:
  terrain <- flsgen_terrain(200, 200)

## End(Not run)
Index

* datasets
  CLASS_LEVEL_TARGETS, 2

CLASS_LEVEL_TARGETS, 2
flsgen_create_class_structure, 2
flsgen_create_class_targets, 3
flsgen_create_terrain, 12