Package ‘rflsgen’

February 23, 2022

Type Package

Title R Interface to the 'Flsgen' Neutral Landscape Generator with Targets on Landscape Indices

Version 1.0.0

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, raster

Imports rgdal, checkmate, utils, jsonlite

SystemRequirements Java (>= 8)

RoxygenNote 7.1.2

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

VignetteBuilder knitr

URL https://dimitri-justeau.github.io/rflsgen/

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

NeedsCompilation no

Repository CRAN

Date/Publication 2022-02-23 12:30:02 UTC
R topics documented:

CLASS_LEVEL_TARGETS . . . . . . . . . . . . . . . . . . . . . . . . . . 2
flsgen_create_class_structure . . . . . . . . . . . . . . . . . . . . . . 2
flsgen_create_class_targets . . . . . . . . . . . . . . . . . . . . . . 3
flsgen_create_landscape_structure . . . . . . . . . . . . . . . . . . . 5
flsgen_create_landscape_targets . . . . . . . . . . . . . . . . . . . . 6
flsgen_create_target_series . . . . . . . . . . . . . . . . . . . . . . 7
flsgen_extract_structure_from_raster . . . . . . . . . . . . . . . . . 8
flsgen_generate . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
flsgen_structure . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
flsgenTerrain . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

Index 14

---

CLASS_LEVEL_TARGETS  Vector of available class targets

Description

Vector of available class targets

Usage

CLASS_LEVEL_TARGETS

Format

An object of class character of length 14.

---

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

flsgen_create_class_structure(class_name, patch_areas)
Arguments

class_name   Name of the class
patch_areas  vector of patch areas

Value

A landscape class structure

Examples

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

---

`flsgen_create_class_targets`

*Creates a set of targets for a landscape class*

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
)
```
**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)

**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:
cls_1 <- flsgen_create_class_targets("class 1", NP=c(1, 10), AREA=c(0, 1000))

## End(Not run)
```
flsngen_create_landscape_structure

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

Usage

```r
flsngen_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 raster 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 flsngen generate

Examples

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

```
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 raster 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
## 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

flsgen_extract_structure_from_raster(
  raster_file,
  focal_classes,
  connectivity = 4
)

Arguments

raster_file  
raster 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,
  output = tempfile(fileext = ".tif"),
  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
- `output`: Path of output raster file (temporary file by default)
- `terrain_file`: Path of input terrain raster file, 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 raster object

Examples

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

```r
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/](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 (target_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

```r
structure <- flsgen_structure(targets_str = json)
```

## End(Not run)

---

**flsgen_terrain**  
*Fractal terrain generator*

### Description

Fractal terrain generation with the diamond-square algorithm

### Usage

```r
flsgen_terrain(
  width,  
  height,  
  output = tempfile(fileext = ".tif"),  
  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
- **output**: Path of output raster file (temporary file by default)
- **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 raster object
Examples

```r
## 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_landscape_structure, 5
flsgen_create_landscape_targets, 6
flsgen_create_target_series, 7
flsgen_extract_structure_from_raster, 8
flsgen_generate, 8
flsgen_structure, 10
flsgenTerrain, 12