Package: TroublemakeR (via r-universe)

August 30, 2024

Title Generates Spatial Problems in R for 'AMPL'

Version 0.0.1

Description Provides methods for generating .dat files for use with the 'AMPL' software using spatial data, particularly rasters. It includes support for various spatial data formats and different problem types. By automating the process of generating 'AMPL' datasets, this package can help streamline optimization workflows and make it easier to solve complex optimization problems. The methods implemented in this package are described in detail in a publication by Fourer et al. (<doi:10.1287/mnsc.36.5.519>).

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Depends R (>= 4.1.0)

LazyData true

Imports data.table, dplyr, purrr, stringr, terra

URL https://github.com/Sustainscapes/TroublemakeR, https://sustainscapes.github.io/TroublemakeR/

BugReports https://github.com/Sustainscapes/TroublemakeR/issues

Repository https://sustainscapes.r-universe.dev

RemoteUrl https://github.com/sustainscapes/troublemaker

RemoteRef HEAD

RemoteSha 54d8daf058cef275c1c43e307c44d2d187188db7

Contents

`create_budget` *Create budget*

Description

This function generates or appends the budget and transition cost to a .dat file for ampl. The file will be written to the location specified by the `name` argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
create_budget(
  budget,
  Rastercurrentlanduse,
  landuses,
  name = "Problem",
  verbose = FALSE
)
```

Arguments

`budget`	maximum cost for the problem
`Rastercurrentlanduse`	raster object of current landuses
`landuses`	character vector with all landuses
`name`	The name of the output file
`verbose`	Logical whether messages will be written while the function is generating calculations, defaults to FALSE

Value

A .dat file. This function is used for the side-effect of writing values to a file.

Author(s)

Derek Corcoran

Examples

```
data(CurrentLanduse)
CurrentLU <- terra::unwrap(CurrentLanduse)


TroublemakeR::create_budget(budget = 2,
Rastercurrentlanduse = CurrentLU,
landuses = c("Agriculture", "Forest", "Urban"),
name = "Problem",
verbose = TRUE)

# delete the file so the test on cran can pass this

file.remove("Problem.dat")
```

Current *A PackedSpatRaster of 4 species with its projected distribution for current conditions*

Description

A PackedSpatRaster of 4 species with its projected distribution for current conditions

Usage

```
Current
```

Format

A PackedSpatRaster with 4 layer::

Spp1 Predicted presence absence for species 1 in current coditions

Spp2 Predicted presence absence for species 1 in current conditions

Spp3 Predicted presence absence for species 1 in current conditions

Spp4 Predicted presence absence for species 1 in current conditions

CurrentLanduse *A PackedSpatRaster of the current landuse*

Description

A PackedSpatRaster of the current landuse

Usage

```
CurrentLanduse
```

Format

A PackedSpatRaster with 1 layer::

Landuse current landuse

define_cells *Define Cells*

Description

This function takes a Raster object and identifies non NA cells and writes them to a .dat file. The file will be written to the location specified by the `name` argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
define_cells(Rasterdomain, name = "Problem")
```

Arguments

`Rasterdomain`	A Raster object with any value in the cells that are part of the problem and NA values where the problem is not to be solved
`name`	The name of the output file

Value

.dat file. This function is used for the side-effect of writing values to a file.

Author(s)

Derek Corcoran

Examples

```
data(Species)
library(terra)
Test <- Species[[1]] |>
terra::unwrap()

# Generate the "Problem.dat" file

define_cells(Test[[1]])

file.remove("Problem.dat")
```

`find_connections` *Find connections*

Description

This function takes a Raster object and identifies non NA cells and finds adjacent cells, to then store them as edges. Those then will be written to the location specified by the `name` argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
find_connections(Rasterdomain, name = "Problem", directions = "rook")
```

Arguments

`Rasterdomain`	A Raster object with any value in the cells that are part of the problem and NA values where the problem is not to be solved
`name`	The name of the output file
`directions`	character or matrix to indicated the directions in which cells are considered connected. The following character values are allowed: "rook" or "4" for the horizontal and vertical neighbors; "bishop" to get the diagonal neighbors; "queen" or "8" to get the vertical, horizontal and diagonal neighbors; or "16" for knight and one-cell queen move neighbors. If directions is a matrix it should have odd dimensions and have logical (or 0, 1) values

Value

.dat file. This function is used for the side-effect of writing values to a file.

Author(s)

Derek Corcoran

Examples

```
library(terra)
r <- rast(nrows=500, ncols=500)

set.seed(2023)
values(r) <- sample(x = 1:6, size = ncell(r),
replace = TRUE,
prob = c(0.03125, 0.0625, 0.09375, 0.15625, 0.25, 0.40625))

##
set.seed(2023)

ForNA <- sample(1:ncell(r), ceiling(ncell(r)/10))

values(r)[ForNA] <- NA

find_connections(Rasterdomain = r, name = "Edges")
file.remove("Edges.dat")
```

`LanduseCombination` *Calculate contiguity bonus based on similarity*

Description

Calculate contiguity bonuses based on landuse similarity and writes them to a .dat file. The file will be written to the location specified by the `name` argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
LanduseCombination(
  Landuses,
  Filters,
  parameter = "ContiguityBonus",
  name = "Problem",
  verbose = FALSE
)
```

Arguments

`Landuses`	is a character vector of the land-uses names
`Filters`	Commonalities to look for in land-uses
`parameter`	The name of the parameter to use
`name`	The name of the output file
`verbose`	Logical whether messages will be written while the function is generating calculations, defaults to FALSE

Value

.dat file. This function is used for the side-effect of writing values to a file.

Examples

```
LanduseCombination(Landuses = c("ForestDryPoor",
"ForestDryRich", "ForestWetPoor",
"ForestWetRich", "OpenDryPoor", "OpenDryRich", "OpenWetPoor",
"OpenWetRich"), Filters = c("Dry", "Wet"))

file.remove("Problem.dat")
```

---

landuse_names *Landuse names*

Description

This function takes a vector of landuse names and writes them to a .dat file. The file will be written to the location specified by the name argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
landuse_names(landuses = NULL, name = "Problem")
```

Arguments

landuses	a vector with the names of the landuses
name	The name of the output file

Value

.dat file. This function is used for the side-effect of writing values to a file.

Author(s)

Derek Corcoran

Examples

```
landuse_names(landuses =  c("Agriculture", "Forest", "Urban"))

# delete the file so the test on cran can pass this

file.remove("Problem.dat")
```

Species *A list of 4 species with its projected distribution for 4 landuses*

Description

A list of 4 species with its projected distribution for 4 landuses

Usage

```
Species
```

Format

A list of 4 Spatrasters with 4 layers each::

Species 1 Predicted presence absence for species 1 in current, forest, agriculture, and urban landuse

Species 2 Predicted presence absence for species 1 in current, forest, agriculture, and urban landuse

Species 3 Predicted presence absence for species 1 in current, forest, agriculture, and urban landuse

Species 4 Predicted presence absence for species 1 in current, forest, agriculture, and urban landuse

Species_Landuse *A list of 4 species with its projected distribution for 4 landuses*

Description

A list of 4 species with its projected distribution for 4 landuses

Usage

```
Species_Landuse
```

Format

A list of 4 Spatrasters with 4 layers each::

Species 1 Predicted presence absence for species 1 in forest, agriculture, and urban landuse

Species 2 Predicted presence absence for species 1 in forest, agriculture, and urban landuse

Species 3 Predicted presence absence for species 1 in forest, agriculture, and urban landuse

Species 4 Predicted presence absence for species 1 in forest, agriculture, and urban landuse

`species_names` *Species names*

Description

This function takes a vector of species names and writes them to a .dat file. The file will be written to the location specified by the `name` argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
species_names(species_names = NULL, name = "Problem")
```

Arguments

`species_names`	a vector with the names of species
`name`	The name of the output file

Value

.dat file. This function is used for the side-effect of writing values to a file.

Author(s)

Derek Corcoran

Examples

```
species_names(species_names = c("Spp1", "Spp2"))
file.remove("Problem.dat")
```

`species_suitability` *Calculate species suitability*

Description

Calculate species suitability from a given raster and species names and writes them to a .dat file. The file will be written to the location specified by the `name` argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
species_suitability(
  Rastercurrent,
  species_names,
  name = "Problem",
  parameter = "SpeciesSuitability",
  verbose = FALSE
)
```

Arguments

Rastercurrent	raster object of current suitability
species_names	character vector of species names
name	The name of the output file
parameter	The name of the parameter to use
verbose	Logical whether messages will be written while the function is generating calculations, defaults to FALSE

Value

.dat file. This function is used for the side-effect of writing values to a file.

Examples

```
library(terra)
data(Current)
Current <- terra::unwrap(Current)
species_suitability(Rastercurrent = Current, species_names = c("Spp1", "Spp2", "Spp3", "Spp4"))

file.remove("Problem.dat")
```

---

species_suitability_landuse

Calculate species suitability for each landuse

Description

Calculate species suitability from a given raster, species names and landuse and writes them to a .dat file. The file will be written to the location specified by the name argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
species_suitability_landuse(
  Rasterspecieslanduse,
  species_names,
  landuses,
  parameter = "SpeciesSuitabilityLanduse",
  name = "Problem",
  verbose = FALSE
)
```

Arguments

Rasterspecieslanduse	a list of species suitability for each landuse
species_names	character vector of species names
landuses	character vector with all landuses
parameter	The name of the parameter to use
name	The name of the output file
verbose	Logical whether messages will be written while the function is generating calculations, defaults to FALSE

Value

.dat file. This function is used for the side-effect of writing values to a file.

Examples

```
library(terra)
data("Species_Landuse")
Species_Landuse <- Species_Landuse |> purrr::map(terra::unwrap)
species_suitability_landuse(Rasterspecieslanduse = Species_Landuse,
species_names = c("Spp1", "Spp2", "Spp3", "Spp4"),
landuses = c("Agriculture", "Forest", "Urban"), name = "Test")
file.remove("Test.dat")
```

---

troublemaker *Troublemaker*

Description

This function is a metafunction with several functions inside of it it takes several spatial objects and generates a .dat file with a spatial dataset for AMPL

Usage

```
troublemaker(
  Rasterdomain = NULL,
  Rastercurrent = NULL,
  species_names = NULL,
  Rasterspecieslanduse = NULL,
  landuses = NULL,
  budget = NULL,
  Rastercurrentlanduse = NULL,
  directions = NULL,
  name = "Problem",
  verbose = FALSE
)
```

Arguments

Rasterdomain	A Raster object with any value in the cells that are part of the problem and NA values where the problem is not to be solved
Rastercurrent	raster object of current suitability
species_names	a vector with the names of species
Rasterspecieslanduse	a list of species suitability for each landuse
landuses	character vector with all landuses
budget	maximum cost for the problem
Rastercurrentlanduse	raster object of current landuses
directions	just as in adjacent character or matrix to indicated the directions in which cells are considered connected. The following character values are allowed: "rook" or "4" for the horizontal and vertical neighbors; "bishop" to get the diagonal neighbors; "queen" or "8" to get the vertical, horizontal and diagonal neighbors; or "16" for knight and one-cell queen move neighbors. If directions is a matrix it should have odd dimensions and have logical (or 0, 1) values
name	The name of the output file
verbose	Logical whether messages will be written while the function is generating calculations, defaults to FALSE

Value

A .dat file with the spatial problem formated for AMPL. This function is used for the side-effect of writing values to a file.

Author(s)

Derek Corcoran

Examples

```
# Example 1 with current suitabilities
data(Species)
data(Current)
library(terra)
Test <- Species[[1]] |>
terra::unwrap()

Current <- terra::unwrap(Current)

# Generate the "Problem.dat" file

TroublemakeR::troublemaker(Rasterdomain =Test[[1]],
Rastercurrent = Current,
species_names = c("Spp1", "Spp2", "Spp3", "Spp4"),
name = "Problem")

# delete the file so the test on cran can pass this

file.remove("Problem.dat")

# Example 2 with landuse suitabilities

data(Species)
data("Species_Landuse")

library(terra)
Test <- Species[[1]] |>
terra::unwrap()

Species_Landuse <- Species_Landuse |> purrr::map(terra::unwrap)

# Generate the "Problem2.dat" file

TroublemakeR::troublemaker(Rasterdomain =Test[[1]],
Rasterspecieslanduse = Species_Landuse,
species_names = c("Spp1", "Spp2", "Spp3", "Spp4"),
landuses = c("Agriculture", "Forest", "Urban"),
name = "Problem2")

# delete the file so the test on cran can pass this

file.remove("Problem2.dat")

 # Example 3 with budget and transition cost

 data("CurrentLanduse")
 CurrentLU <- terra::unwrap(CurrentLanduse)
 TroublemakeR::troublemaker(Rasterdomain =Test[[1]],
 Rasterspecieslanduse = Species_Landuse,
 species_names = c("Spp1", "Spp2", "Spp3", "Spp4"),
 landuses = c("Agriculture", "Forest", "Urban"),
```

```
Rastercurrentlanduse = CurrentLU,
budget = 2,
name = "Problem3",
verbose = FALSE)

file.remove("Problem3.dat")
```

`write_ampl_lines` *Writes an AMPL line*

Description

This function takes a character and writes them to a .dat file. The file will be written to the location specified by the `name` argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
write_ampl_lines(line, name = "Problem")
```

Arguments

`line`	line to be written to .dat file
`name`	The name of the output file

Value

.dat file. This function is used for the side-effect of writing values to a file.

Examples

```
write_ampl_lines("param s:= 1")

file.remove("Problem.dat")
```

write_cell_param *Write cell parameters*

Description

This function takes a Raster object, uses its values as a parameter and writes them to a .dat file. The file will be written to the location specified by the name argument. If the file already exists, it will be overwritten. The file format is plain text, with each line terminated by a newline character.

Usage

```
write_cell_param(
  Rasterparam,
  parameter,
  default = NULL,
  name = "Problem",
  verbose = FALSE
)
```

Arguments

Rasterparam	A Raster object with the values for the parameter
parameter	The name of the parameter to use
default	The value of the default value for the parameter if there is one, otherwise keep it as NULL
name	The name of the output file
verbose	Logical whether messages will be written while the function is generating calculations, defaults to FALSE

Value

.dat file. This function is used for the side-effect of writing values to a file.

Examples

```
library(terra)

A <- TroublemakeR::Current |> terra::unwrap()
A <- A[[1]]

write_cell_param(Rasterparam = A, parameter = "Suitability", name = "Problem")

 write_cell_param(Rasterparam = A, parameter = "Carbon", default = 1,
 name = "Problem")

 write_cell_param(Rasterparam = A, parameter = "Cost", default = 0,
 name = "Problem")
```

```
file.remove("Problem.dat")
```

Index