

# Package ‘geogrid’

August 19, 2018

**Title** Turn Geospatial Polygons into Regular or Hexagonal Grids

**Version** 0.1.0.1

**Description** Turn irregular polygons (such as geographical regions) into regular or hexagonal grids. This package enables the generation of regular (square) and hexagonal grids through the package 'sp' and then assigns the content of the existing polygons to the new grid using the Hungarian algorithm, Kuhn (1955) (<doi:10.1007/978-3-540-68279-0\_2>). This prevents the need for manual generation of hexagonal grids or regular grids that are supposed to reflect existing geography.

**Imports** methods, sp, sf, rgeos, Rcpp

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**LinkingTo** Rcpp, RcppArmadillo

**Suggests** testthat, covr, lintr

**RoxygenNote** 6.0.1

**URL** <https://github.com/jbaileyh/geogrid>

**BugReports** <https://github.com/jbaileyh/geogrid/issues>

**NeedsCompilation** yes

**Author** Joseph Bailey [aut, cre],  
Ryan Hafen [ctb],  
Lars Simon Zehnder [ctb] (RcppArmadillo implementation of Munkres'  
Assignment Algorithm)

**Maintainer** ORPHANED

**Repository** CRAN

**Date/Publication** 2018-08-19 08:51:30 UTC

**X-CRAN-Original-Maintainer** Joseph Bailey  
<jbailey@futurecities.catapult.org.uk>

**X-CRAN-Comment** Orphaned and corrected on 2018-08-19 as check problems were not corrected despite reminders.

## R topics documented:

assign_polygons . . . . .	2
calculate_cell_size . . . . .	3
calculate_grid . . . . .	4
geogrid . . . . .	5
get_shape_details . . . . .	5
get_shape_details_internal . . . . .	6
hungariansafe_cc . . . . .	6
hungarian_cc . . . . .	7
plot.geogrid . . . . .	7
read_polygons . . . . .	8

<b>Index</b>	<b>9</b>
--------------	----------

---

assign_polygons	<i>Assign the polygons in the original spatial data to their new location.</i>
-----------------	--

---

### Description

Assigns each polygon in the original file to a new location in the gridded geometry using the Hungarian algorithm.

### Usage

```
assign_polygons(shape, new_polygons)
```

### Arguments

shape	A "SpatialPolygonsDataFrame" object representing the original spatial polygons.
new_polygons	A "geogrid" object returned from <a href="#">calculate_grid</a> .

### Value

A SpatialPolygonsDataFrame.

### Examples

```
## Not run:
input_file <- system.file("extdata", "london_LA.json", package = "geogrid")
original_shapes <- read_polygons(input_file)

# calculate grid
new_cells <- calculate_grid(shape = original_shapes,
  grid_type = "hexagonal", seed = 1)
plot(new_cells)

#
```

```

grid_shapes <- assign_polygons(original_shapes, new_cells)
par(mfrow=c(1, 2))
sp::plot(original_shapes)
sp::plot(grid_shapes)

# look at different grids using different seeds
par(mfrow=c(2, 3), mar = c(0, 0, 2, 0))
for (i in 1:6) {
  new_cells <- calculate_grid(shape = original_shapes, grid_type = "hexagonal", seed = i)
  plot(new_cells, main = paste("Seed", i, sep=" "))
}

## End(Not run)

```

---

calculate\_cell\_size     *Calculate size of grid items (deprecated).*

---

## Description

Given an input multipolygon spatial data frame this function calculates the required cell size of a regular or hexagonal grid.

## Usage

```

calculate_cell_size(shape, shape_details = NULL, learning_rate = 0.03,
  grid_type = c("hexagonal", "regular"), seed = NULL, verbose = FALSE)

```

## Arguments

shape	A "SpatialPolygonsDataFrame" object representing the original spatial polygons.
shape_details	deprecated.
learning_rate	The rate at which the gradient descent finds the optimum cellsize to ensure that your gridded points fit within the outer boundary of the input polygons.
grid_type	Either "hexagonal" for a hexagonal grid (default) or "regular" for a regular grid.
seed	An optional random seed integer to be used for the grid calculation algorithm.
verbose	A logical indicating whether messages should be printed as the algorithm iterates.

---

calculate_grid	<i>Calculate grid from spatial polygons.</i>
----------------	--

---

### Description

Given an input multipolygon spatial data frame this function calculates a hexagonal or regular grid that strives to preserve the original geography.

### Usage

```
calculate_grid(shape, learning_rate = 0.03, grid_type = c("hexagonal",
  "regular"), seed = NULL, verbose = FALSE)
```

### Arguments

shape	A "SpatialPolygonsDataFrame" object representing the original spatial polygons.
learning_rate	The rate at which the gradient descent finds the optimum cellsize to ensure that your gridded points fit within the outer boundary of the input polygons.
grid_type	Either "hexagonal" for a hexagonal grid (default) or "regular" for a regular grid.
seed	An optional random seed integer to be used for the grid calculation algorithm.
verbose	A logical indicating whether messages should be printed as the algorithm iterates.

### Examples

```
## Not run:
input_file <- system.file("extdata", "london_LA.json", package = "geogrid")
original_shapes <- read_polygons(input_file)

# calculate grid
new_cells <- calculate_grid(shape = original_shapes,
  grid_type = "hexagonal", seed = 1)
plot(new_cells)

#
grid_shapes <- assign_polygons(original_shapes, new_cells)
par(mfrow=c(1, 2))
sp::plot(original_shapes)
sp::plot(grid_shapes)

# look at different grids using different seeds
par(mfrow=c(2, 3), mar = c(0, 0, 2, 0))
for (i in 1:6) {
  new_cells <- calculate_grid(shape = original_shapes, grid_type = "hexagonal", seed = i)
  plot(new_cells, main = paste("Seed", i, sep=" "))
}

## End(Not run)
```

---

geogrid	<i>geogrid</i>
---------	----------------

---

## Description

Turn irregular polygons (such as geographical regions) into regular grids.

## Author(s)

Joseph Bailey <jbailey@futurecities.catapult.org.uk>

## Examples

```
## Not run:
input_file <- system.file("extdata", "london_LA.json", package = "geogrid")
original_shapes <- read_polygons(input_file)

# calculate grid
new_cells <- calculate_grid(shape = original_shapes,
  grid_type = "hexagonal", seed = 1)
plot(new_cells)

#
grid_shapes <- assign_polygons(original_shapes, new_cells)
par(mfrow=c(1, 2))
sp::plot(original_shapes)
sp::plot(grid_shapes)

# look at different grids using different seeds
par(mfrow=c(2, 3), mar = c(0, 0, 2, 0))
for (i in 1:6) {
  new_cells <- calculate_grid(shape = original_shapes, grid_type = "hexagonal", seed = i)
  plot(new_cells, main = paste("Seed", i, sep=" "))
}

## End(Not run)
```

---

get_shape_details	<i>Extract details from provided polygons (deprecated).</i>
-------------------	---

---

## Description

Extract spatial extent, range and other geospatial features from the output of `read_polygons`. Items are returned as a list for use in `calculate_grid`.

## Usage

```
get_shape_details(input_shape)
```

**Arguments**

input\_shape      A "SpatialPolygonsDataFrame" object representing the original spatial polygons.

---

get\_shape\_details\_internal

*Extract details from provided polygons.*

---

**Description**

Extract spatial extent, range and other geospatial features from the output of read\_polygons. Items are returned as a list for use in [calculate\\_grid](#).

**Usage**

```
get_shape_details_internal(input_shape)
```

**Arguments**

input\_shape      A "SpatialPolygonsDataFrame" object representing the original spatial polygons.

---

hungariansafe\_cc

*hungariansafe\_cc*

---

**Description**

hungariansafe\_cc

**Usage**

```
hungariansafe_cc(cost)
```

**Arguments**

cost              cost matrix

---

hungarian_cc	<i>hungarian_cc</i>
--------------	---------------------

---

**Description**

hungarian\_cc

**Usage**

hungarian\_cc(cost)

**Arguments**

cost            cost matrix

---

plot.geogrid	<i>Plot a "geogrid" object</i>
--------------	--------------------------------

---

**Description**

Plot a "geogrid" object

**Usage**

```
## S3 method for class 'geogrid'  
plot(x, y, ...)
```

**Arguments**

x            An object of class "geogrid" to plot.  
y            ignored  
...          Additional parameters passed to the "sp" package's plot method.

---

read_polygons	<i>Import spatial data.</i>
---------------	-----------------------------

---

**Description**

Simple function to read spatial data into a SpatialPolygonsDataFrame. Based on `st_read` from package `sf`.

**Usage**

```
read_polygons(file)
```

**Arguments**

file	A file path pointing to a shapefile or GeoJSON file, or a character string holding GeoJSON data. See the <code>dsn</code> argument of <a href="#">st_read</a> for more details.
------	---

# Index

`assign_polygons`, 2

`calculate_cell_size`, 3

`calculate_grid`, 2, 4, 5, 6

`geogrid`, 5

`geogrid-package (geogrid)`, 5

`get_shape_details`, 5

`get_shape_details_internal`, 6

`hungarian_cc`, 7

`hungariansafe_cc`, 6

`plot.geogrid`, 7

`read_polygons`, 8

`st_read`, 8