Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. doi: 10.18637/jss.v000.i00

Stephanie Kobakian
Queensland University of Technology

Dianne Cook
Monash University

Abstract

This algorithm creates a tesselated hexagon display to represent each of the spatial polygons. It allocates these hexagon in a manner that preserves the spatial relationship of the geographic units. It showcases spatial distributions, by emphasising the small geographical regions that are often difficult to locate on geographic maps. Spatial distributions have been presented on alternative representations of geography for many years. In modern times, interactivity and animation have begun to play a larger role, as alternative representations have been popularised by online news sites, and atlas websites with a focus on public consumption. Applications are increasingly widespread, especially in the areas of disease mapping, and election results.

Keywords: spatial, statistics, cartogram.

1. Introduction

The current practice for presenting geospatial data involves a choropleth map display. These maps highlight the geographic patterns in geospatially related statistics (Moore and Carpenter 1999). The land on the map space is divided into geographic units, these boundaries are usually administrative, such as states or counties. The units are filled with colour to represent the value of the statistic (Tufte 1990).

Australian residents are increasingly congregating around major cities, the vast rural areas are often sparsely populated in comparison to the urban centres. In Australia, government bodies such as the Australian Bureau of Statistics, and the Australian Electoral Commission hold the responsibility for the division of geographic units. These boundaries are adjusted as the population increases. The division of the population into approximately equal population areas results in dramatically different square meterage of the geographic areas. This results in an unequal attention given to the statistic of each area, this can allow misrepresentation of the spatial distributions of human related statistics in geographic maps.

The solution to this visualisation problem begins with the geography. Cartograms apply a transformation to the geographic boundaries based on the value of the statistic of interest.

These displays result in a distortion of the map space to represent differences in the statistic across the areas (Dougenik, Chrisman, and Niemeyer 1985). The statistic of interest is used to determine the layout. When using the Australian population, the result is a population cartogram that fails to preserve a recognisable display due to the difference in size of metropolitan and rural areas (Dorling 2011), (Berry, Morrill, and Tobler 1964). Contiguous cartograms change the shape of an area, preserving boundary relationships of neighbours. Non-contiguous cartograms maintain the geographic shape of each area, but lose the connection to neighbours as each areas shrinks or grows.

Alternative maps shift the focus from land area and shape, to the value of the statistics in a group of areas. Alternative mapping methods allow increased understanding of the spatial distribution of a variable across the population, by fairly representing each administrative area. This acknowledges that the amount of residents can be different but recognises that each area, or person is equally important.

Tilegrams, Rectangular cartograms (van Kreveld and Speckmann 2007) and Dorling cartograms (Dorling 2011), all use one simple shape to represent each area. This allows preservation of spatial relationships and decreases the emphasis from the amount of geographic area. These maps focus on the relationship between neighbours attempting to preserve connections, and disregard the unique shapes of the administrative boundaries.

The **sugarbag** package provides a new method to create tesselated hexagon tilegrams. Extending the tilegram to Australian applications required preserving the spatial relationships. It emphasises the capital cities as population hubs, and emphasises the distances rather than size of large, rural geographic units.

2. Algorithm

This solution operates on a set of **sf** (Pebesma 2018) polygons.

There are four steps to create a tesselated hexagon tilegram. These steps can be executed by the main function, `create_hexmap`, or can be implemented separately for more flexibility. There are parameters used in the process that can be provided, if they are not, they will be automatically derived.

1. Create the set of centroids to allocate
2. Create the grid of hexagons locations to use
3. Allocate each centroid to an available hexagon
4. Transform the data for plotting

2.1. Parameters

The `create_hexmap` function requires several parameters, if they are not provided, the information will be derived from the simple features (**sf**) set of shapes used. Users may choose to only use the `allocate` function when they wish to use a set of centroids, rather than **sf** polygons.

The following must be provided to `create_hexmap`:

parameter	description
shp	an sf object containing the polygon information
sf_id	name of a column that distinguishes unique areas
focal_points	a data frame of reference locations used to allocate hexagons

2.2. Polygon set

The polygon set of Statistical Areas at Level 2 (SA2) (Australian Bureau of Statistics 2018) of Tasmania in 2016 is provided with the **sugarbag** package as `tas_sa2`. A single column of the data set must be used to identify the unique areas. In this case, the unique SA2 names for each SA2 have been used.

The longitude and latitude centre of the capital cities of Australia are used to allocate areas around the closest capital city. Hobart will be the common focal point, as this example uses only the areas in the state of Tasmania.

```
R> data(capital_cities)
```

The following parameters will be determined within `create_hexmap` if they are not provided. They are created throughout the following example:

parameter	description
buffer_dist	a float value for distance in degrees to extend beyond the geometry provided
hex_size	a float value in degrees for the diameter of the hexagons
hex_filter	amount of hexagons around centroid to consider for allocation
width	the angle used to filter the grid points around a centroid

2.3. Create the set of centroid points

Individual **sugarbag** functions can be used outside of the main function. The set of polygons should be provided as an **sf** object, this is a data frame containing a `geometry` column. The `read_shape` function can assist in creating this object.

The centroids can be derived from the set of polygons using the `create_centroids` function:

```
R> centroids <- create_centroids(shp_sf = tas_sa2, sf_id = "SA2_NAME16")
```

2.4. Create the hexagon grid points

A tilegram presents areas on a tellesated set of tiles. The grid is created to ensure tessellation between the hexagons.

The grid of possible hexagon centroids is made using the `create_grid` function. The grid creation requires several steps. It uses the centroids, the hexagon size and the buffer distance.

```
R> grid <- create_grid(centroids = centroids, hex_size = 0.2, buffer_dist = 1.2)
```

Step 1: Creating a tesselated grid

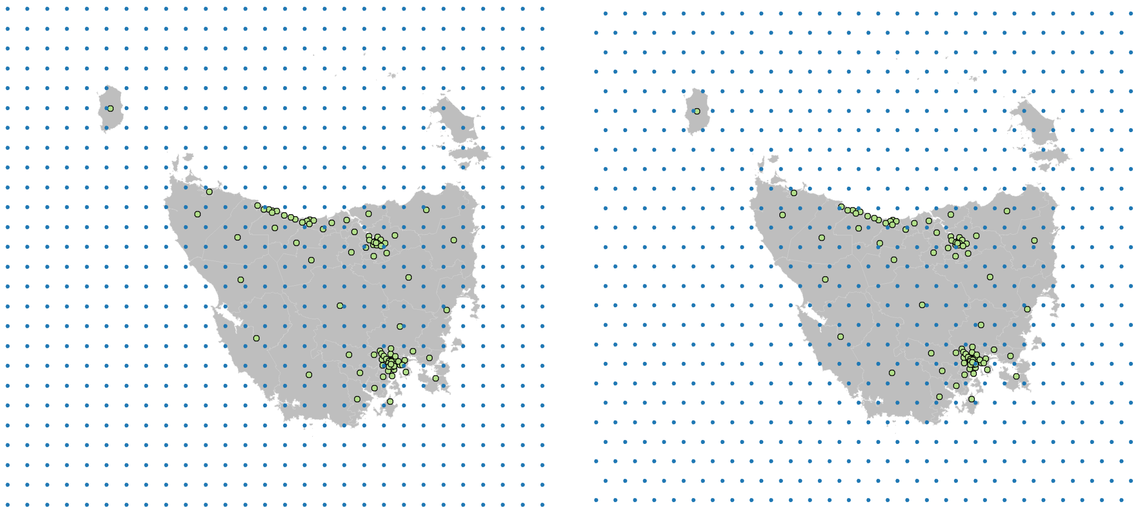


Figure 1: Grid points to create a tilegram.

A set of longitude columns, and latitude rows are created to define the locations of the hexagons. The distance between each row and column is defined by the size specified as `hex_size`. The minimum and maximum, longitude and latitude values of the centroid locations are found. Equally spaced columns are created from the minimum longitude minus the buffer distance, up to the maximum longitude plus the buffer distance. The rows are then created from the latitude values and the buffer distance. An individual hexagon location is created from all intersections of the longitude columns and latitude rows.

A square grid could be used for sqaure tiles, but it will not facilitate tessellated hexagons. Figure 1 allows for hexagons, as every second latitude row on the grid is shifted right, by half of the hexagon size.

Step 2: Rolling windows

Not all of the grid points will be used, especially if islands result in a large grid space. To filter the grid for appropriate points for allocation, the `create_buffer` function is called within `create_grid`. It finds the grid points needed to best capture the set of centroids on a hexagon tile map.

For each centroid location, the closest latitude row and longitude column are found. Then rows and columns of centroids are divided into 20 groups. The amount of rows in each latitude group and the amount of columns in each longitude group are used as the width of rolling windows. The rolling windows can be seen on the This will tailor the available grid points to those most likely to be used. It also helps reduce the amount of time taken, as it decreases the amount of points considered for each centroid allocation.

The first rolling window function finds the minimum and maximum centroid values for the sliding window groups of longitude columns and the groups of latitude rows.

The second rolling window function finds the average of the rolling minimum and maximum

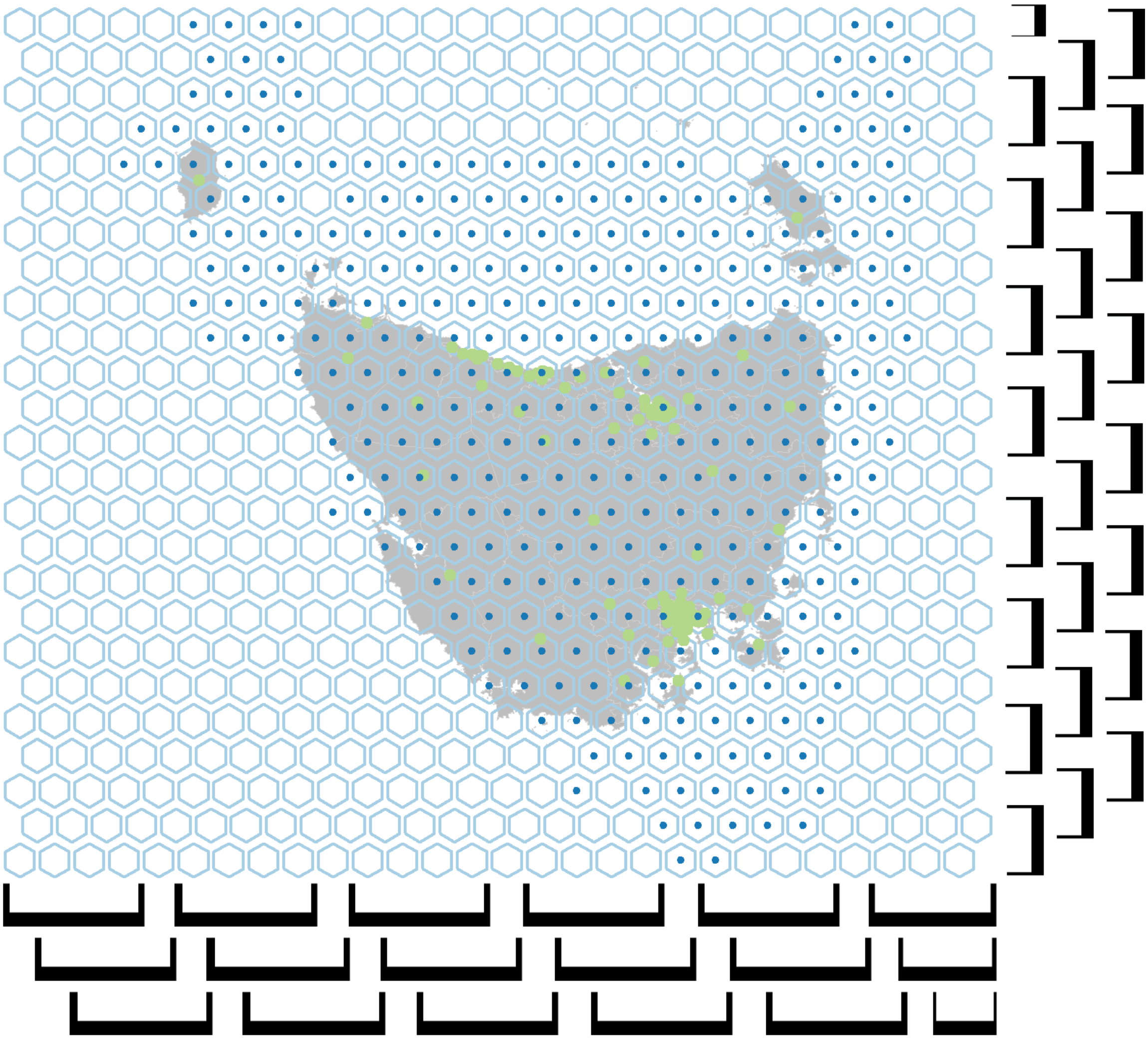


Figure 2: All possible hexagon locations from the initial grid are shown with blue outlines. The blue dots shown the grid points left as to choose from after the buffer step. The rolling windows to the right show the rows used to filter the hexagon locations.

centroid values, for the longitude columns and latitude rows.

Step 3: Filtering the grid

Only the grid points between the rolling average of the minimum and maximum centroid values are kept, for each row and column of the grid.

2.5. Centroid to focal point distance

The distance between each centroid in the set, and each of the focal points provided is calculated. The name of the closest focal point, and the distance and angle from focal point to polygon centroid is joined to polygon data set. To minimise time taken for this step, only Tasmania's capital city Hobart is provided. The order for allocation is determined by the distance between the polygon centroid and it's closest focal point. The points are arranged from the centroid closest to the focal points, to the furthest.

2.6. Allocate each centroid to a hexagon grid point

Allocation of all centroids takes place using the set of polygon centroids and the hexagon map grid. Centroid allocation begins with the closest centroid to a focal point. This will preserve spatial relationships with the focal point, as the inner city areas are allocated first, they will be placed closest to the capital, and the areas that are further will then be accommodated. Only the hexagon grid points that have not yet been allocated are considered.

The possible hexagon locations consider for a centroid location are determined by the `hex_filter`. This is the maximum amount of hexagons between the centroid and the furthest considered hexagon. It is used to subset possible grid points to only those surrounding the polygon centroid within an appropriate range. A smaller distance will increase speed, but can decrease accuracy if the angle width increases.

```
R> hexmap_allocation <- allocate(
R>   centroids = centroids %>% select(SA2_NAME16, longitude, latitude),
R>   sf_id = "SA2_NAME16",
R>   hex_grid = grid,
R>   hex_size = 0.2, # same size used in create_grid
R>   hex_filter = 10,
R>   width = 35,
R>   focal_points = capital_cities,
R>   verbose = TRUE)
```

The following example considers one of the Statistical Areas at Level 2. Within the algorithm, these steps are repeated for each polygon.

Step 1: Filter the grid for unassigned hexagon points

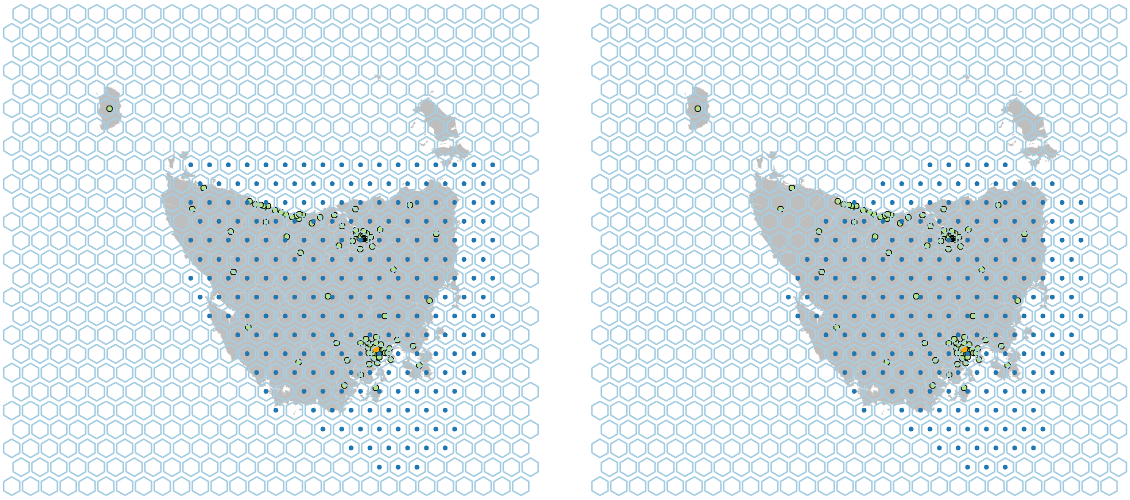
Keep only the available hexagon points, this will prevent multiple areas being allocated to the same hexagon.

Step 2: Filter the grid points for those closest to the centroid

This will allow only the closest points that are not yet assigned, to be considered.

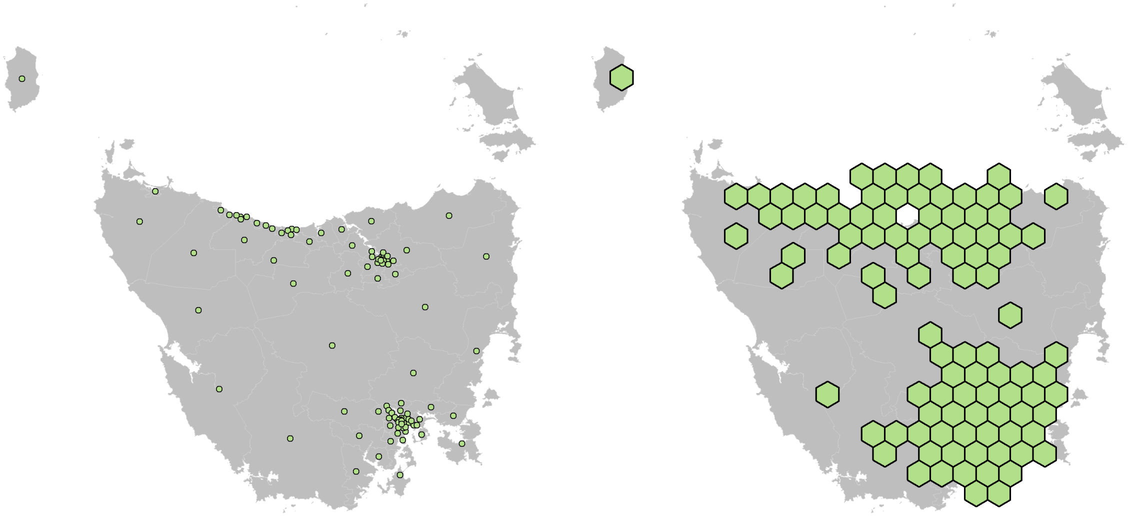
A box of possible hexagon locations around the centroid. The corners of the box may not look square as the buffer has already removed unnecessary points from over the ocean.

The algorithm then removes the outer corners of the square, creating a circle of points, by only keeping points within a certain radial distance around the original centroid location.



The `width` parameter is used to take a slice of the remaining points. This uses the angle from the closest capital city, to the current centroid. This allows the spatial relationship to be preserved, even when it is allocated to a hexagon that is further from the focal point then the original centroid location.

If no available hexagon grid point is found within the original filter distance and angle, the distance is expanded, only when a maximum distance is reached will the angle expand to accommodate more possible grid points.
The angle begins at 30 degrees by default, and will increase if no points can be found within the `hex_filter` distance. The allocation is returned and combined with the data relating to each polygon.



2.7. Neighbour relationships

It is possible to encourage the use of neighbourhood relationshipd, for stronger preservation of neighbour relations.

An additional step may be included to allow the neighbours that have alredy been allocated to influence the placement of the current centroid. This requires respecifying the `sf` object as the argument for the `use_neighbours` parameter. This calculates neighbours using intersections of their polygons. This occurs for all areas before any allocations begin.

For the current centroid, the list of neighbours is consulted. If any neighbour was already allocated, the surrounding hexagons on the grid are prioritised. For multiple neighbours, the neighbouring hexagon grid points are aggregated and considered in order of distance from the original centroid.

3. Using sugarbag

3.1. Installation

The package can be installed from CRAN:

```
R> install.packages("sugarbag")
```

and the development version can be install from the GitHub repository:

```
R> devtools::install_github("srkobakian","sugarbag")
```

Load the library into your R session with:

```
R> library(sugarbag)
```

3.2. Creating a hexagon tile map

The following code creates the hexagon tile map for all the Statistical Areas at Level 2 in Tasmania.

```
R> # Load data
R> data(tas_sa2)
R>
R> # Create centroids set
R> centroids <- create_centroids(tas_sa2, "SA2_NAME16")
R>
R> # Create hexagon grid
R> grid <- create_grid(centroids = centroids,
R>     hex_size = 0.2,
R>     buffer_dist = 1.2)
R>
```

```
R> # Allocate polygon centroids to hexagon grid points
R> hex_allocated <- allocate(
R>   centroids = centroids,
R>   hex_grid = grid,
R>   sf_id = "SA2_NAME16",
R>   # same column used in create_centroids
R>   hex_size = 0.2,
R>   # same size used in create_grid
R>   hex_filter = 10,
R>   use_neighbours = tas_sa2,
R>   focal_points = capital_cities %>% filter(points == "Hobart"),
R>   width = 35,
R>   verbose = FALSE)
R>
R> # Prepare to plot
R> fort_hex <- fortify_hexagon(data = hex_allocated, sf_id = "SA2_NAME16", hex_size = 0.2)
R>
R> # Make a plot
R> library(ggplot2)
R> ggplot(fort_hex) +
R>   geom_polygon(aes(x=long, y=lat, group=hex_id, fill = lat)) +
R>   scale_fill_distiller("", palette="PRGn")
```

4. Applications

4.1. Australian Cancer Atlas

XXX Show whole Australia male liver cancer with choropleth and hexagon tile map, and discuss the pros/cons of the result, for understanding the incidence.

The hexagon tile map visualisation method can be applied to the Australian Cancer Atlas data. A small example of Lung Cancer Standardised Incidence Rates (SIRs) allows two views of the same data produced by the Australian Cancer Atlas. This small example in Figure 3 shows the group of blue areas in the Hobart CBD more prominently in the hexagon tile map (b). The small red areas visible in the choropleth map (a) along the north coast are much larger in the hexagon tile maps. The hexagon tile map shows less yellow, this no longer overwhelms the map space with the information regarding the more rural areas.

XXX Move this image (ref: sir) to the end of the algorithm section to illustrate the final result for Tassie

4.2. Election results

XXX Code from web site, and a couple of paragraphs of explanation

5. Animation

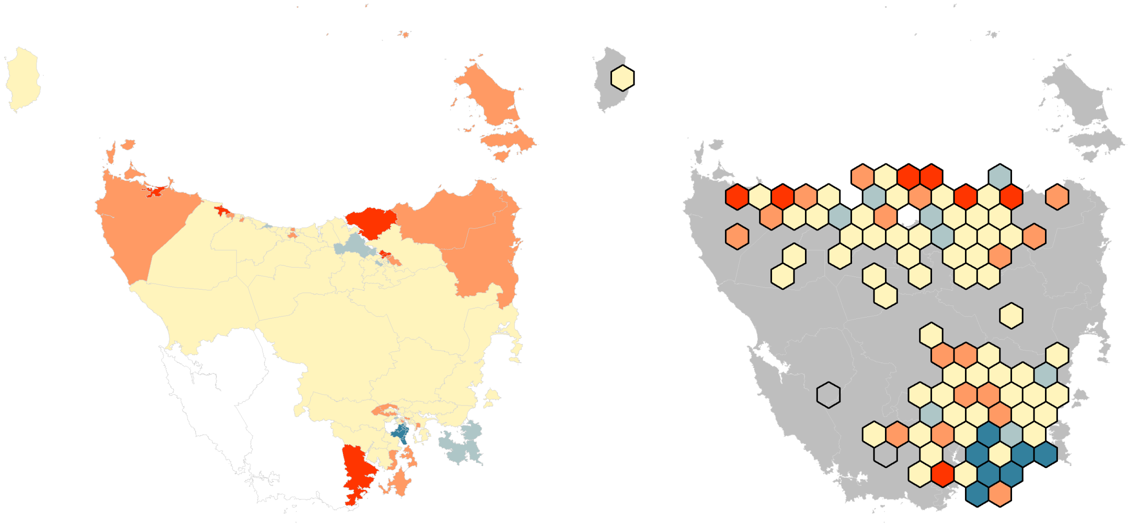


Figure 3: The Australian Cancer Atlas data has determined the colour of each Statistical Area of Australian at Level 2. A choropleth map (a) of Standardised Incidence Rates (SIRs) is paired with a hexagon tile map (b) to contrast the colours that are made obvious when every SA2 is equally represented.

XXX show how to make an animation, and make a link to a web page where the animation can be viewed

6. Conclusion

Structure of conclusions: (1) summarise paper, discuss limitations, suggest future directions

It is possible to use alternative maps to communicate spatial distributions.While a choropleth map display is the current practice spatial visualisation of geographical data. Current methods do not always work for Australia due to the large gaps between densely populated capital cities. The administrative boundaries may distract from the statistics, communicated using colour.

Alternative maps highlight the value of the statistics across the geographic units. Alternative mapping methods allow increased understanding of the spatial distribution of a variable across the population, by fairly representing each administrative area. This acknowledges that the amount of residents can be different but recognises that each population area is equally important. The solution to this visualisation problem has equally sized areas, with neighbourhood boundary connections. This map algorithm is implemented in the Kobakian and Cook (2019) package written for R Core Team (2012). The **sugarbag** package creates tesselated hexagon tilegrams. The Australian application preserves the spatial relationships, emphasising capital cities. The hexagon tile map is a visualisation solution that highlights spatial distributions.

7. Acknowledgements

The authors would like to thank the Australian Cancer Atlas team for discussions regarding alternative spatial visualizations. They would also like to thank Professor Kerrie Mengersen and Dr Earl Duncan for suggestions and comments. Mitch O'Hara-Wild provided assistance with some parts of the algorithm and code. Sayani Gupta provided helpful advice on parts of this paper.

The code for sugarbag can be found at XXX, along with details on installation and usage.

References

Australian Bureau of Statistics (2018). URL https://www.abs.gov.au/websitedbs/D3310114.nsf/home/Australian+Statistical+Geography+Standard+(ASGS).

Berry BJL, Morrill RL, Tobler WR (1964). "Geographic Ordering of Information: New Opportunities." *The Professional Geographer*, **16**, 39–44. doi:10.1111/j.0033-0124.1964.039_q.x.

Dorling D (2011). *Area Cartograms: Their Use and Creation*, volume 59, pp. 252 – 260. ISBN 9780470979587. doi:10.1002/9780470979587.ch33.

Dougenik JA, Chrisman NR, Niemeyer DR (1985). "AN ALGORITHM TO CONSTRUCT CONTINUOUS AREA CARTOGRAMS." *The Professional Geographer*, **37**(1), 75–81. doi:10.1111/j.0033-0124.1985.00075.x. https://doi.org/10.1111/j.0033-0124.1985.00075.x, URL https://doi.org/10.1111/j.0033-0124.1985.00075.x.

Kobakian S, Cook D (2019). *sugarbag: Create Tessellated Hexagon Maps.* Https://srkobakian.github.io/sugarbag/, https://github.com/srkobakian/sugarbag.

Moore DA, Carpenter TE (1999). "Spatial Analytical Methods and Geographic Information Systems: Use in Health Research and Epidemiology." *Epidemiologic Reviews*, **21**(2), 143–161. ISSN 1478-6729. doi:10.1093/oxfordjournals.epirev.a017993. http://oup.prod.sis.lan/epirev/article-pdf/21/2/143/6727658/21-2-143.pdf, URL https://doi.org/10.1093/oxfordjournals.epirev.a017993.

Pebesma E (2018). "Simple Features for R: Standardized Support for Spatial Vector Data." *The R Journal*, **10**(1), 439–446. doi:10.32614/RJ-2018-009. URL https://doi.org/10.32614/RJ-2018-009.

R Core Team (2012). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Tufte ER (1990). *Envisioning Information.* Graphics Press.

van Kreveld M, Speckmann B (2007). "On rectangular cartograms." *Computational Geometry*, **37**(3), 175 – 187. ISSN 0925-7721. doi:10.1016/j.comgeo.2006.06.002. Special Issue on the 20th European Workshop on Computational Geometry, URL http://www.sciencedirect.com/science/article/pii/S0925772106000770.

Affiliation:

Stephanie Kobakian
Queensland University of Technology
School of Mathematical Sciences, Science and Engineering Faculty, Brisbane, QLD, Australia
E-mail: stephanie.kobakian@qut.edu.au

Dianne Cook
Monash University
Department of Econometrics and Business Statistics, Melbourne, VIC, Australia
E-mail: dicook@monash.edu

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd