# Sea regions for rectangular cartograms

Document status and date:
Published: 01/01/2014

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Download date: 04. Oct. 2023

# Sea Regions for Rectangular Cartograms

Kevin Buchin[*]     Rafael G. Cano[†]     Pedro J. de Rezende[†]     Cid C. de Souza[†]     Bettina Speckmann[*]

## Abstract

In a rectangular cartogram, each region of a map is represented by a rectangle whose area is proportional to some statistical data of interest. Current techniques for constructing rectangular cartograms partition a large rectangle (the map) into a set of smaller rectangles which correspond to land or sea regions. The position and size of sea rectangles determine the outline of land masses. Therefore, sea regions have a direct impact on the recognizability and, thus, on the visual quality of cartograms. In this paper, we describe the first algorithm for the automated creation of sea regions for rectangular cartograms and present results obtained with our method.

## 1   Introduction

Cartograms are a well-known tool for visualizing statistical data associated with geographic regions. Usually, regions represent countries, states or counties, and they are deformed in the cartogram in such a way that their area is proportional to the numerical data related to them. It is not necessary that the regions keep their original outline; in fact, it is acceptable to represent them by simple geometric shapes. Although regions are not drawn with their true sizes or shapes, recognizability should be preserved.

There are several types of cartograms. Here, we are interested in *rectangular cartograms* (see Fig. 1). As the name suggests, in this kind of cartogram all relevant regions are represented by rectangles. This has the advantage that areas are easily estimated. However, it also means that most regions will have little resemblance to their actual shapes, and care should be taken to make the cartogram as recognizable as possible.

The quality of a cartogram depends on several factors, which are summarized in the following list:

- Cartographic error: for each region $A$, let $A_c$ denote the actual area of the region in the cartogram and $A_s$ denote the intended area of that
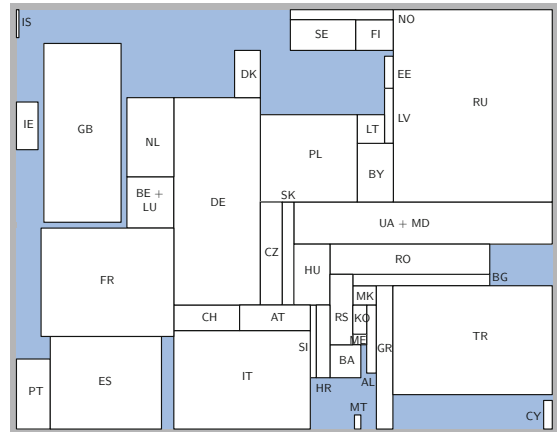


Figure 1: Europe population.

region given by the numerical data to be conveyed. The cartographic error is defined as $|A_c - A_s|/A_s$;

- Preserved adjacencies: two rectangles should be adjacent if and only if the corresponding regions are adjacent in the original map;

- Aspect ratio of the rectangles: large aspect ratios make it harder to visually compare the areas of two rectangles and should be avoided;

- Relative positions: rectangles should be placed in such a way that their relative position resembles that of the original map (e.g., the rectangle for the Netherlands should be west of the one for Germany).

It is not always possible to optimize all criteria simultaneously. Thus, often cartograms have (some) regions with non-zero cartographic error and incorrect adjacencies.

**Related work.** Van Kreveld and Speckmann [9] presented the first algorithm to compute rectangular cartograms. Their method is able to construct cartograms with small cartographic error, at the cost of slightly disturbed adjacencies. Speckmann et al. [7] later improved on these results using an iterative linear programming method. Buchin et al. [2] coupled an evolutionary algorithm with the approach described in [7] and showed that the resulting heuristic produces high quality cartograms with only correct adjacencies.

---

[*]Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands, `k.a.buchin@tue.nl`, `speckman@win.tue.nl`. K.B. and B.S. are supported by the Netherlands Organisation for Scientific Research (NWO) under project nos. 612.001.207 and 639.023.208, respectively.

[†]Institute of Computing, University of Campinas, Brazil, `[rgcano|rezende|cid]@ic.unicamp.br`. Supported by CNPq, FAPESP and FAEPEX/UNICAMP.
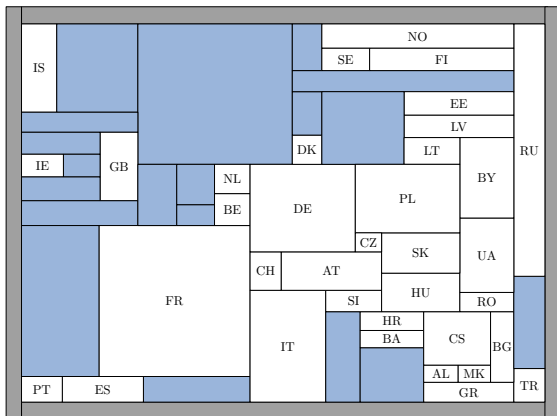
Figure 2: Rectangular dual with sea regions visible.

Inoue et al. [4] computed rectangular cartograms with a different strategy. They first triangulate the regions and later transform the triangles to obtain the desired area. However, this leads to large cartographic errors. Heilmann et al. [3] presented an algorithm that always gives the correct areas, but might greatly disturb adjacencies.

**Results and Organization.** The algorithms developed over the sequence of papers [2, 7, 9] use the dual graph of a map as input. This dual graph needs to be augmented with additional "sea vertices" to ensure a reasonable shape of the final cartogram. These sea vertices correspond to rectangular "sea regions" in the final cartogram – see Fig. 2 for an example of a rectangular dual of an augmented dual graph of Europe with outlined sea regions. The augmentation with sea vertices or regions has so far been performed manually. However, it is desirable that also the augmentation step becomes part of the automatic pipeline, to adapt the number and location of sea regions to particular data sets and visualization needs.

Here, we propose a first algorithm to automatically create sea regions to augment a given input map. In the following section, we sketch the general method underlying the algorithms of [2, 7, 9]. We describe our approach in Section 3 and show experimental results in Section 4.

## 2  Algorithmic outline

The algorithms in [2, 7, 9] all follow the same general method to generate rectangular cartograms. Initially, the adjacencies in the original map are represented in a *face graph* $F$, which contains a vertex for each region and an edge between two vertices if the corresponding regions are adjacent. The regions of the input map are considered to be connected, so the face graph is always planar.

Given a face graph $F$, its *rectangular dual* is defined as a partition of a rectangle $R$ into a set $\mathcal{R}$ of non-overlapping rectangles such that i) no four rectangles in $\mathcal{R}$ meet at the same point; ii) each $r \in \mathcal{R}$ is associated with one vertex of $F$; and iii) two rectangles in $\mathcal{R}$ share a common boundary if and only if there is an edge between the corresponding vertices in $F$.

A rectangular dual is then constructed with the help of a *regular edge labeling* (REL) of $F$, as described by Kant and He [5]. An REL contains information on the relative positions between pairs of rectangles and uniquely determines the rectangular dual (up to equivalence). There can be exponentially many RELs for a given face graph. In the most recent approach [2], an evolutionary algorithm is used to select a suitable REL based on the statistical data that is to be visualized. Then, a rectangular dual is built with the algorithm described in [5]. The final step assigns the desired area to each region of the rectangular dual. This is done by rearranging the segments in the rectangular dual in such a way that adjacencies are not altered. The currently best approach to do so is based on iterative linear programming [7].

Not every face graph $F$ has a rectangular dual. Koźmiński and Kinnen [6] and Ungar [8] independently proved that a face graph $F$ has a rectangular dual with four rectangles on the boundary if and only if ($i$) every interior face of $F$ is a triangle and the exterior face is a quadrangle; and ($ii$) $F$ does not have *separating triangles*, i.e., triangles with vertices in their interior as well as in their exterior. Hence, it is usually necessary to preprocess maps to ensure that the face graph satisfies the conditions above. In addition, as mentioned before, it often improves the visual quality significantly to augment the face graph with additional sea vertices (or the rectangular dual with sea regions) to ensure a suitable shape of the cartogram. These preprocessing and augmentation steps are so far performed manually.

## 3  Sea partitioning

In the following, we describe our approach for the automated creation of sea regions. We view the input map as a polygon with holes: the outer rectangular border of the map is the polygon and the continents and islands are the holes. We then compute the straight skeleton [1] of the polygon and use its cells as the basis to create sea regions (see Fig. 3).

Next, we prune the skeleton. This is done for two reasons. First, the number of regions needs to be reduced to speed up the construction of the cartogram. Second, we must ensure that the corresponding face graph has a rectangular dual, which is not always true for the original skeleton. In particular, separating triangles often arise from sequences of two or more convex vertices on the boundary of a country.

By definition, the rectangular dual cannot have four rectangles that meet in a point. This will occur if, af-
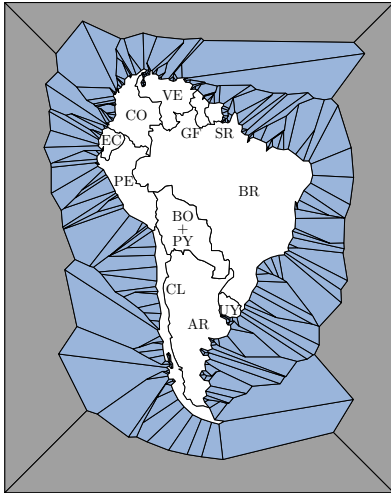
Figure 3: Straight skeleton for South America.

ter adding the edges of the straight skeleton, some vertex is left with degree more than three. To satisfy this first requirement, we identify these vertices and remove the branches from the skeleton that are incident to it.

In the next step, we try to reduce the number of sea regions as much as possible without affecting the quality of the final cartogram. To do this, for each country that has skeleton branches incident to its boundary, we select a vertex whose branch will not be erased (see Fig. 4). As a result, most land regions become adjacent to two or no sea regions.

This strategy must be altered in two cases. In a rectangular cartogram, every region except the four boundary regions must have at least one neighbor in each direction. This implies that islands must be surrounded by at least four sea regions. Moreover, countries that have only one adjacent land region, like Portugal, need to be adjacent to at least three sea regions. In these cases, we select more than one vertex on the country's coastline whose branch will not be erased.

As mentioned in Section 1, recognizability must be preserved in cartograms. This goal can be partially achieved by choosing suitable relative positions for the
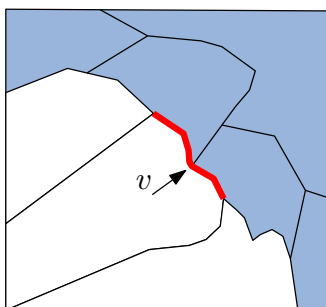


Figure 4: A vertex $v$ on the boundary of a country whose corresponding branch will not be erased.

rectangles. Consider the cartogram in Fig. 1. As expected, Belgium, Luxembourg and the Netherlands are placed north of France. Besides that, they are also aligned with the right vertical segment of the rectangle for France, which makes the cartogram more similar to the original map.

These details can be captured by the objective function of the evolutionary algorithm proposed by Buchin et al. [2]. In particular, a good measure in this case is the angle deviation of the region centroids. Nevertheless, when creating the sea regions, we must guarantee that the desired configuration for a group of regions can be realized in the cartogram. We next show that our strategy allows for all configurations but one, which is treated separately.

Suppose we are given three land regions $A$, $B$, and $C$ such that $\{A, B\}$ and $\{B, C\}$ are pairwise adjacent, as shown in Fig. 5. In addition, we assume that sea regions are placed to the right of $A$, $B$, and $C$. Our strategy creates one sea region adjacent to $\{A, B\}$ and another to $\{B, C\}$. Disregarding symmetries, there are six possible configurations for the land regions. In five of them, two sea regions suffice to create the cartogram. However, the sixth configuration is realizable only with an extra rectangle. In these cases, we create an extra region adjacent to $B$. As an example, note the sea regions adjacent to Brazil in Figs. 6 and 7.
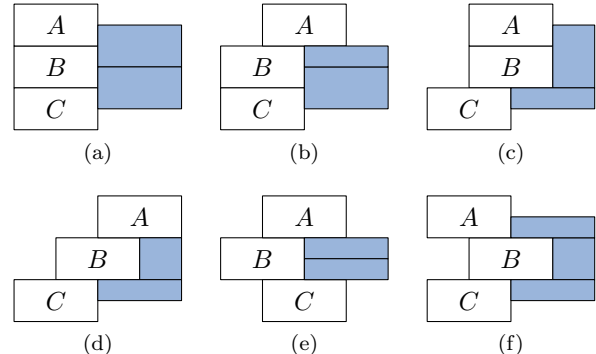


Figure 5: (a)–(e) five basic configurations that can be realized with only two sea regions; (f) the only configuration that needs an extra rectangle.

## 4 Experimental results

The algorithms for the construction of the cartograms were implemented in Java. Straight skeletons were computed with a C++ interface for CGAL 4.0.2.

We use two maps in our experiments: South America (12 countries) and Europe (42 countries). We experimented with several data sets which include information on demographics, economics and services.

Fig. 6 shows the subdivision for South America after pruning the skeleton. The pruning algorithm reduced the number of sea regions from 259 to 13 in less
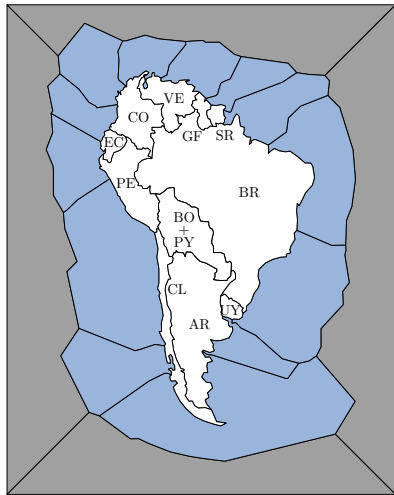
Figure 6: Sea regions for South America after pruning the skeleton.

than a second. Fig. 7 shows the final cartogram for population data. The cartographic error for all land regions is less than 0.1%.
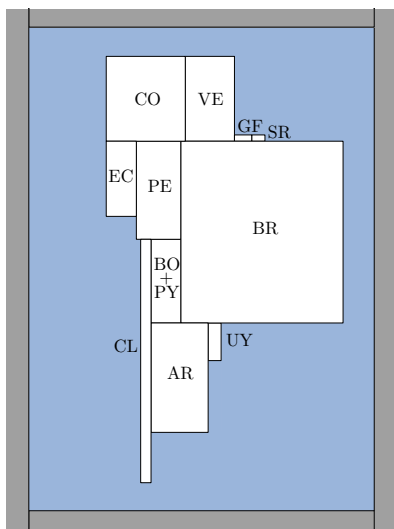


Figure 7: South America population.

Fig. 8 shows the number of cell phone lines in Europe. The skeleton creates 845 sea regions, which are later reduced to 44 after pruning. Again, the cartographic error for all land regions is less than 0.1% and the sea regions are created in less than a second.

## 5   Conclusion

We presented the first algorithm for the creation of sea regions for rectangular cartograms. Our method is based on computing and pruning the straight skeleton of the polygon surrounding the input map. Our pruning strategy not only is fast, but also achieves a significant reduction in the number of sea regions cre-
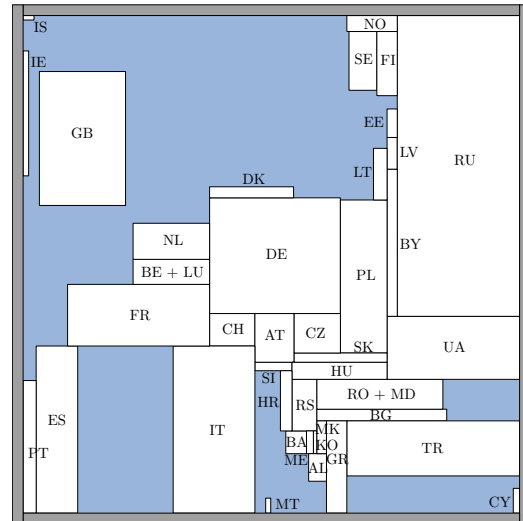


Figure 8: Europe cell phone lines.

ated by the skeleton. First experiments show that the cartograms generated with this algorithm are visually pleasing and have a very low cartographic error.

## References

[1] O. Aichholzer, F. Aurenhammer, D. Alberts, and B. Grtner. A novel type of skeleton for polygons. *Journal of Universal Computer Science*, 1(12):752–761, 1995.

[2] K. Buchin, B. Speckmann, and S. Verdonschot. Evolution strategies for optimizing rectangular cartograms. In *Proc. 7th International Conference on Geographic Information Science*, LNCS 7478, pages 29–42, 2012.

[3] R. Heilmann, D. A. Keim, C. Panse, and M. Sips. Recmap: Rectangular map approximations. In *Proc. IEEE Symp. on Information Visualization*, pages 33–40, 2004.

[4] R. Inoue, K. Kitaura, and E. Shimizu. New solution for construction of rectilinear area cartogram. In *Proc. 24th International Cartography Conference*, 2009.

[5] G. Kant and X. He. Regular edge labeling of 4-connected plane graphs and its applications in graph drawing problems. *Theoretical Computer Science*, 172 (12):175 – 193, 1997.

[6] K. Koźmiński and E. Kinnen. Rectangular duals of planar graphs. *Networks*, 15(2):145–157, 1985.

[7] B. Speckmann, M. van Kreveld, and S. Florisson. A linear programming approach to rectangular cartograms. In *Proc. 12th International Symposium on Spatial Data Handling*, pages 529–546, 2006.

[8] P. Ungar. On diagrams representing maps. *Journal of London Mathematical Society*, 28:336–342, 1953.

[9] M. van Kreveld and B. Speckmann. On rectangular cartograms. *Computational Geometry: Theory and Applications*, 37(3):175–187, 2007.