

Taylor & Francis
Taylor & Francis Group

Cartography and Geographic Information Science

ISSN: 1523-0406 (Print) 1545-0465 (Online) Journal homepage: www.tandfonline.com/journals/tcag20

Topology-preserving line densification for creating contiguous cartograms

Nihal Z. Miaji, Adi Singhania, Matthias E. Goh, Callista Le, Atima Tharatipyakul & Michael T. Gastner

To cite this article: Nihal Z. Miaji, Adi Singhania, Matthias E. Goh, Callista Le, Atima Tharatipyakul & Michael T. Gastner (2026) Topology-preserving line densification for creating contiguous cartograms, Cartography and Geographic Information Science, 53:5, 547-562, DOI: 10.1080/15230406.2026.2625329

To link to this article: https://doi.org/10.1080/15230406.2026.2625329

© 2026 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

View supplementary material

Published online: 10 Mar 2026.

Submit your article to this journal

Article views: 757

View related articles

View Crossmark data

Citing articles: 1 View citing articles

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=tcag20

CARTOGRAPHY AND GEOGRAPHIC INFORMATION SCIENCE
2026, VOL. 53, NO. 5, 547–562
https://doi.org/10.1080/15230406.2026.2625329

Taylor & Francis
Taylor & Francis Group

OPEN ACCESS Check for updates

Topology-preserving line densification for creating contiguous cartograms

Nihal Z. Miaji[a,b], Adi Singhania[a,b], Matthias E. Goh[b], Callista Le[b], Atima Tharatipyakul[a] and Michael T. Gastner[a]

[a]Information and Communications Technology Cluster, Singapore Institute of Technology, Singapore; [b]Mathematical, Computational and Statistical Sciences, Yale-NUS College, Singapore

ABSTRACT

Cartograms depict geographic regions with areas proportional to quantitative data. However, when created using density-equalizing map projections, cartograms may exhibit invalid topologies if boundary polygons are drawn using only a finite set of vertices connected by straight lines. Here we introduce a method for topology-preserving line densification that guarantees that cartogram regions remain connected and non-overlapping when using density-equalizing map projections. By combining our densification technique with a flow-based cartogram generator, we present a robust framework for strictly topology-preserving cartogram construction. Quantitative evaluations demonstrate that the proposed algorithm produces cartograms with greater accuracy and speed than alternative methods while maintaining comparable shape fidelity.

ARTICLE HISTORY
Received 3 March 2025
Accepted 26 January 2026

KEYWORDS
Density-equalizing map projections; computational geometry; cartogram metrics; quadtree; Delaunay triangulation

1. Introduction

Cartograms are maps in which the areas of regions (e.g. administrative divisions) are proportional to spatially extensive quantitative data (e.g. population). As cartogram areas are typically not proportional to land areas, the shapes or the contiguity of the polygons representing the geographic regions must be adjusted. If all adjacent polygons share boundaries even after resizing their areas, the cartogram is called contiguous, such as the example shown in Figure 1. In practice, cartograms have been applied to a wide range of data, including election results (Gastner et al., 2005), socio-economic indicators (Dorling, 1996), disease incidence (Selvin et al., 1988), and climate change risks (Döll, 2017).

Contiguous cartograms can be constructed using density-equalizing map projections, as demonstrated by W. R. Tobler (1963) and W. Tobler (2017). However, upon transformation via a density-equalizing map projection, invalid topologies – such as intersections or gaps between adjacent polygons – can arise. This issue occurs because computer-generated maps represent enumeration units as polygons with a finite set of vertices connected by straight lines. Here we introduce a method for topology-preserving line densification that solves this problem. Line densification involves augmenting the density of points along boundaries to allow finer representation of the underlying curves. When discretizing boundaries, line densification can ensure that neighboring regions remain connected but do not overlap, as illustrated in Figure 2.

To evaluate the proposed densification algorithm's effectiveness, we augment the flow-based cartogram generation algorithm by Gastner et al. (2018) with our densification method. We refer to the resulting method as 5FCarto, where "5F" refers to five key properties: flow-based, faultless (i.e. no topology violations), free (open-source under GPL v3), fast (quicker than other current methods), and focused (primarily allocating computational resources to regions that require significant transformation).

We compare 5FCarto to Sun's (2020a) force-based cartogram generation algorithm (F4Carto) and to the basic flow-based (BFB) method, which is identical to 5FCarto except that it does not include the densification step. The results, evaluated using four different performance metrics applied to 32 maps, demonstrate that 5FCarto produces topologically valid cartograms with polygon shapes comparable to those produced by other methods. Moreover, 5FCarto outperforms the alternatives in terms of area accuracy and speed, making it a robust and efficient solution for cartogram generation.

2. Problem statement

Density-equalizing map projections provide a mathematically principled method for creating contiguous

CONTACT Michael T. Gastner michael.gastner@singaporetech.edu.sg
Supplemental data for this article can be accessed online at https://doi.org/10.1080/15230406.2026.2625329

© 2026 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms on which this article has been published allow the posting of the Accepted Manuscript in a repository by the author(s) or with their consent.

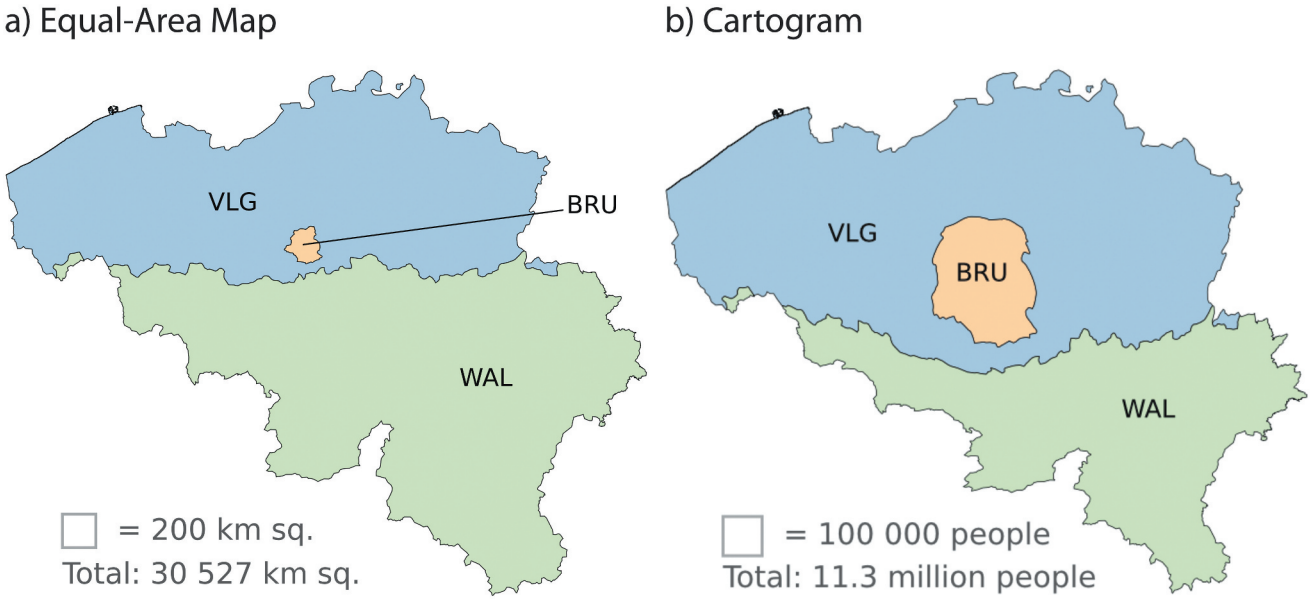


Figure 1. Maps of Belgium. a) Belgium's's regions on an equal-area map, abbreviated as BRU (Brussels), VLG (flanders), and WAL (Wallonia). b) Contiguous cartogram representing population. As Brussels is the most densely populated region, its area is enlarged in the cartogram.

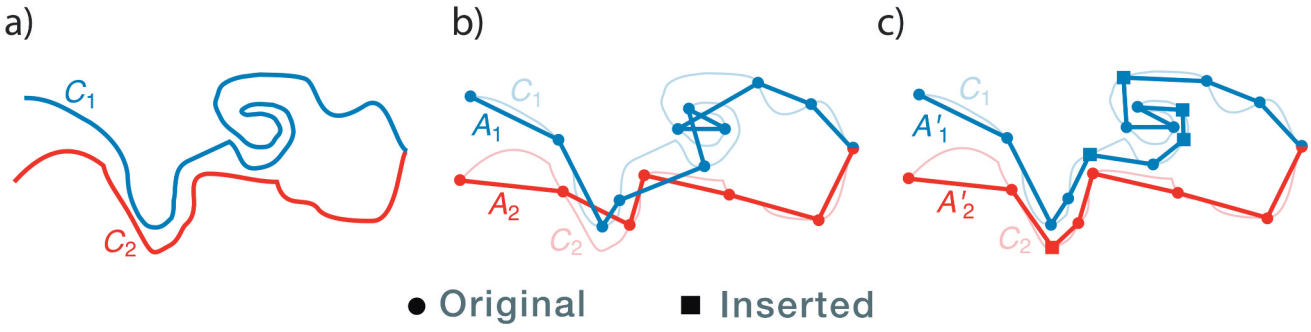


Figure 2. Illustration of line densification when approximating a curve with a finite number of points. Panel a) depicts the ground truth, showing the precise geometry of two curves, C_1 and C_2. Panel b) illustrates finite approximations, A_1 and A_2, that use a sparse distribution of points along the respective curves. Topology is violated as A_1 intersects itself and also A_2. Panel c) demonstrates line densification resulting in A'_1 and A'_2 preserving the true topology.

cartograms. In this approach, the input data are expressed as spatial densities (e.g. population per unit area), and each region is resized so that the density becomes uniform across the entire map. Afterward, the polygon vertices are projected and connected with lines to create the cartogram. A density-equalizing map projection Π must satisfy:

$$J_\Pi(x,y) = \frac{\rho(x,y)}{\overline{\rho}}\ , \tag{1}$$

where ρ is the density function, and $\overline{\rho}$ is the average density. The left-hand side of Equation (1) is the Jacobian determinant $J_\Pi(x,y) = \frac{\partial \Pi_x}{\partial x}\frac{\partial \Pi_y}{\partial y} - \frac{\partial \Pi_y}{\partial x}\frac{\partial \Pi_x}{\partial y}$ of the projection, which we assume to be differentiable in the entire mapping domain, for example, by applying pycnophylactic interpolation (Gastner et al., 2022). Additionally, we assume that the density is positive everywhere. This assumption poses no practical restriction because mixtures of positive and non-positive data (e.g. temperature anomalies or net migration) cannot be proportional to physical areas, and, thus, cartograms would be unsuitable for the visualization of such data; instead, alternative visualization techniques or appropriate data transformations must be considered. Under the assumption that $\rho(x,y)$ is strictly positive with a finite upper bound for all real-valued x and y, Hadamard's global inverse function theorem implies that Π is invertible (Ohkita & Tsukamoto, 2024). Thus, density-equalizing map projections theoretically guarantee that no intersections are generated by the transformation.

In practice, however, the boundary of a region needs to be approximated by a polygon with a finite number of points. This restriction can result in topology violations emerging after the transformation, even for continuous density-equalizing map projections, as illustrated by an intersection emerging in Figure 3.

Our topology-preserving line densification aims to prevent such violations. To capture topology preservation precisely, we define a line approximation of a curve as a finite sequence of points on the curve (endpoints included), with each consecutive pair connected by a straight segment. For instance, A_1 and A_2 are line approximations of C_1 and C_2 in Figure 2. A set of line approximations $A_1, \ldots, A_n$ for curves $C_1, \ldots, C_n$ that have no common points, with the only possible exception of common end points, is topology-preserving if all of the following three properties hold:

P1. A_i is free of self-intersections for all $i = 1, \ldots, n$.

P2. Any pair A_i and A_j with $i \neq j$ has no points in common except, possibly, the endpoints.

P3. At each endpoint where three or more curves meet, the approximations A_i, A_j, and A_k of any three such curves preserve the same winding order as the original curves C_i, C_j, and C_k.

In Figure 2, A_1' and A_2' preserve topology, whereas A_1 and A_2 do not.

3. Related work

3.1. Topology violations in cartograms

The problem of topology violations arising from cartogram construction is well-documented in the literature. Dougenik et al. (1985) noted that their force-based algorithm can produce intersecting borders, especially around panhandle-shaped regions. Similarly, House and Kocmoud (1998) observed the possibility of intersections during cartogram transformations. As a mitigation strategy, they introduced an intersection-specific penalty term in their relaxation-based method for cartogram construction. Furthermore, W. Tobler (2004) acknowledged the intersection problem and suggested a technique called underrelaxation, which reduces the displacements caused by the cartogram transformation. However, Tobler noted that, while underrelaxing alleviates the problem, it does not entirely prevent it.

A well-known strategy in data visualization to address the intersection problem caused by coordinate transformations is line densification, as additional vertices along the unprojected polygon edges increase the flexibility of the projected polygons. Software implementations of line densification include the R functions coord_munch() in the ggplot2 package (Wickham, 2016) and smooth_densify() in the smoothr package (Strimas-Mackey, 2023). These functions insert

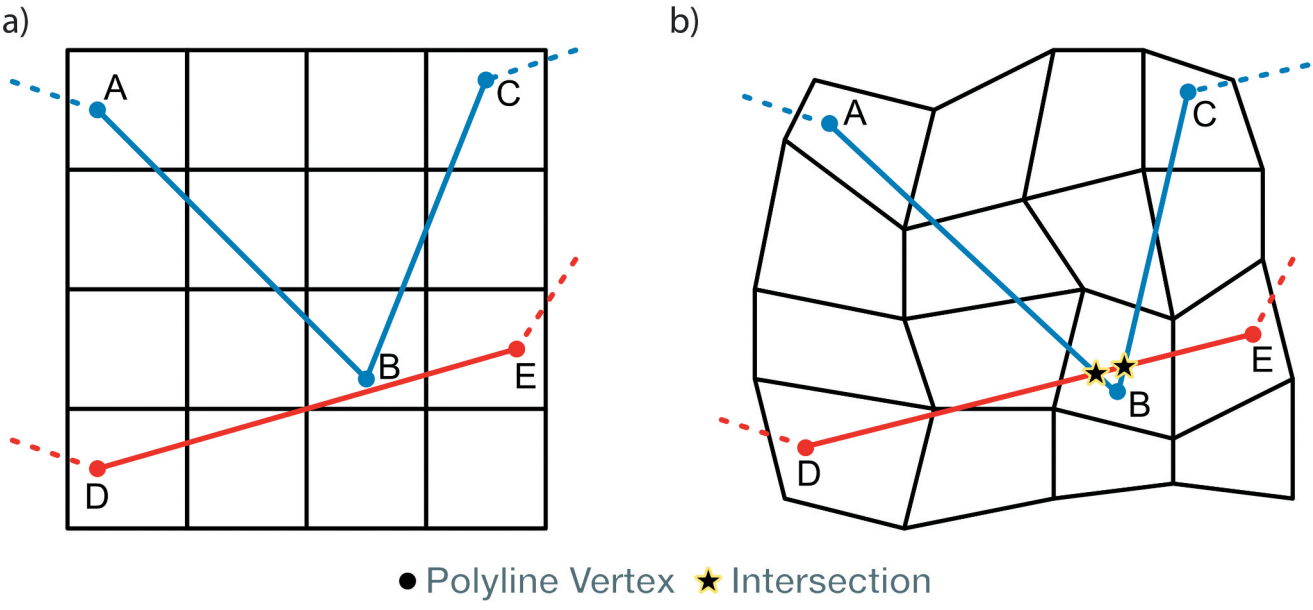


Figure 3. Illustration of a topology violation caused by projecting polylines. The figure depicts two polylines, one colored blue (ABC) and the other red (DE). Panel a) illustrates the original geometry, while panel b) shows the situation after transformation via a density-equalizing map projection. The transformed black grid lines indicate the projection. The projected positions of the polyline vertices A to E are interpolated on the basis of the quadrilaterals formed by the projected grid lines in panel b). Although the blue and red polylines do not intersect before the transformation, they cross each other afterward, resulting in an invalid topology.

vertices at regular intervals. However, this method does not optimize the number of insertions. On the one hand, if too few vertices are inserted, the densification might be too sparse to prevent intersections completely. On the other hand, excessive densification causes unnecessary computational and memory overhead as well as bloated image file sizes.

Dougenik et al. (1985) described a method guaranteeing non-overlapping regions when using their force-based cartogram algorithm. Their method proceeds by dividing the unprojected polygons into convex subpolygons. Yet, as pointed out by Sun (2020a), the complexity of this method has prevented its implementation in cartogram software. Instead, Sun (2013) suggested using quadtrees as auxiliary data structures and inserting additional vertices at intersections between polygons and quadtree edges. However, it is not self-evident how to interpolate projected coordinates such that the projection is continuous at the boundaries of adjacent quadtree cells, which may differ in size. Therefore, in this paper, we describe an algorithm that supplements the quadtree representation of the density-equalizing map projection with a Delaunay triangulation to guarantee topology preservation.

3.2. Cartogram evaluation

The evaluation of cartograms lacks a universally accepted set of metrics, with different studies adopting various approaches. Nusrat et al. (2016) proposed three key metrics: area error, deformation, and topological accuracy, the latter defined as the fraction of adjacent regions in the original map that are not adjacent in the cartogram. However, this metric is uninformative for contiguous cartograms because topological accuracy is always perfect. Instead, Sun (2020a) extended the evaluation framework by considering topological integrity, reporting the number of overlapping or self-intersecting regions as another measure of cartogram quality. Additionally, both Gastner et al. (2018) and Sun (2020a) used area error, deformation, and running time as primary metrics for evaluating cartograms. In this paper, we follow the evaluation strategies proposed in previous works by adopting topological integrity, area error, deformation, and running time as quality metrics.

For area error, the choice of metric varies between studies. Henriques et al. (2009) suggest traditional metrics such as the quadratic mean error, the weighted mean error, and the simple average error. Sun (2020a) utilized the area-weighted mean error, which prioritizes errors from larger regions. In contrast, Gastner et al. (2018) employed the maximum relative area error, focusing equally on all regions regardless of size. The maximum relative area error represents a stricter criterion, as the area-weighted mean error can be significantly reduced by only focusing on large regions. Furthermore, reducing the maximum relative area error not only involves reshaping large polygons but also requires modifying small polygons, a process that is computationally challenging and, thus, indicative of algorithmic robustness. Consequently, in this study we adopt the maximum relative area error as a metric.

For deformation analysis, Gastner et al. (2018) utilized Tissot's indicatrix. However, Sun (2020a) highlighted that Tissot's indicatrix is a local measure and unsuitable for measuring the global distortion of polygons. Consequently, Sun (2020a) proposed using the symmetric difference – previously applied by Cano et al. (2015)—along with the Fréchet and Hausdorff distances as distortion metrics. In contrast, Sagar (2014) quantified shape errors in terms of the shapiness index, which encapsulates overall morphological similarity. Because our primary concern is the displacement of individual polygon vertices, we adopt the deformation metrics introduced by Sun (2020a). Unlike their approach of averaging the Fréchet distance, Hausdorff

Algorithm 1: Proposed Densification

Input: Multipolygons representing geographic boundaries (possibly with holes), densities, and density-equalizing map projection.

Output: Densified multipolygons guaranteed to remain free of intersections under the density-equalizing map projection.

(1) **Quadtree Construction:**
Recursively split leaf nodes with the largest density differences until a predetermined number of leaf nodes is reached. Then grade the quadtree to ensure that the depth difference between neighboring leaf nodes is ≤ 1.

(2) **Delaunay Triangulation:**
Triangulate the quadtree corners and select the triangulation configuration that maximizes the minimum angles of the projected triangles.

(3) **Line Densification:**
Insert points at the intersections of polygon edges with Delaunay triangle edges.

distance, and symmetric difference into a single composite measure, we evaluate these metrics separately to provide a more detailed and granular evaluation.

4. Proposed algorithm

Algorithm 1 provides a concise summary of our densification procedure. In the following sections, we detail each step of the algorithm.

4.1. Quadtree construction

We begin by constructing a quadtree from the rasterized input density. Following Gastner and Newman (2004), any raster cell falling outside the map's polygons is assigned the mean density, denoted by $\bar{\rho}$ in Equation (1). We initialize the quadtree with its root as the only leaf node – a square that encloses the entire map. We then iteratively split the leaf node containing the highest difference between maximum and minimum density until a predetermined number of leaf nodes is reached (Figure 4). In our implementation, we set this target to $2^{-8} \approx 0.004$ times the number of raster cells, typically resulting in 200 to 17,000 leaf nodes, depending on the resolution of the density grid provided as input. This threshold was chosen to optimize average running times across a test set of 32 real-world maps; we report detailed measurements in the online supplement (Table S6 and Figure S1).

After reaching the target leaf count, we perform additional splits to ensure that the depth difference between neighboring quadtree nodes does not exceed 1, a process known as grading. The role of grading for our algorithm is discussed in Section 4.3.

By subdividing quadtree nodes where density variations are greatest and bounding the total number of leaves, this construction concentrates computational effort on region requiring the strongest deformation while ensuring efficient execution.

4.2. Delaunay triangulation

Once the quadtree is constructed, we generate a constrained Delaunay triangulation – denoted T—of the graded quadtree cell corners, as illustrated in Figure 5. This triangulation is a variant of the standard Delaunay triangulation that enforces the inclusion of a predefined set of edges. Initially, we create a temporary constrained Delaunay triangulation, T_{temp}, using the projected quadtree cell corners and adding the projected quadtree cell edges as constraints. By constructing the Delaunay triangulation in the projected space, we maximize the minimum angle in the triangle, thereby ensuring numerically robust interpolation by preventing excessively skinny triangles. Next, we unproject each edge in T_{temp}. If no other quadtree cell corner lies along an unprojected edge, we incorporate it as a constraint in T. After all constraints have been added, we perform a Delaunay triangulation on the unprojected quadtree vertices to construct T. Our implementation leverages CGAL's Delaunay triangulation module (Devillers et al., 2024).

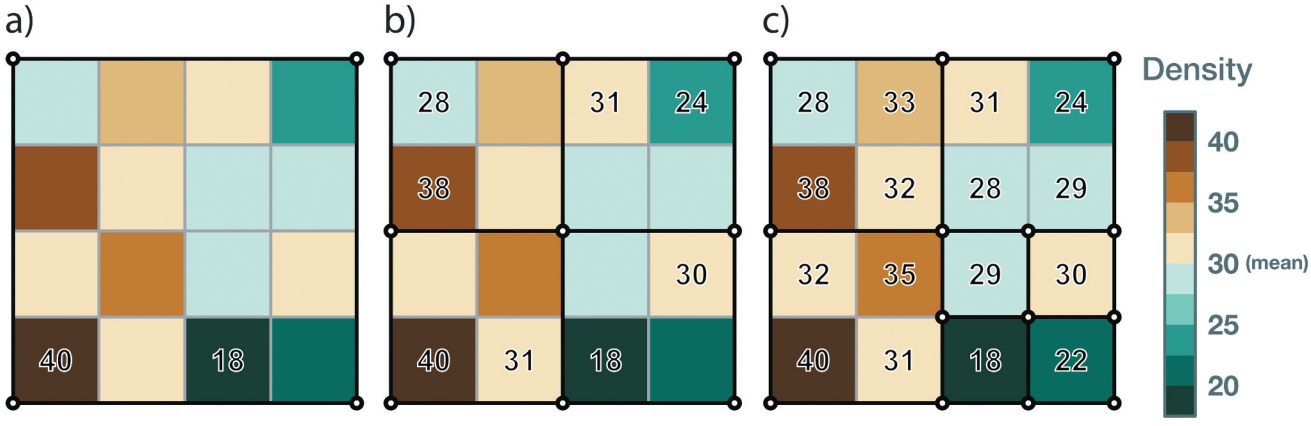


Figure 4. Illustration of the quadtree splitting criteria. The squares that are delineated by thin gray lines indicate a rasterized density grid, while those outlined in black denote quadtree leaf nodes. The quadtree is constructed by recursively splitting an $n \times n$ cell containing the largest difference between maximum and minimum density into four $(n/2) \times (n/2)$ cells until a predetermined number of leaf nodes (7 in this example) is reached. The squares in the raster grid that contain the maximum and minimum density values are annotated with those values. Panel a) depicts the initial 4×4 cell, which is split due to having the largest internal density difference (22) and because the current number of leaf nodes (1) has not reached the target (7). Panel b) illustrates the resulting subdivision with 4 leaf nodes, where again the node with the largest density difference (12, bottom-right) is selected for further splitting, as the target number of leaf nodes has still not been met. Panel c) shows the final subdivision with 7 leaf nodes, at which point the desired number of leaf nodes has been reached, marking the completion of the quadtree.

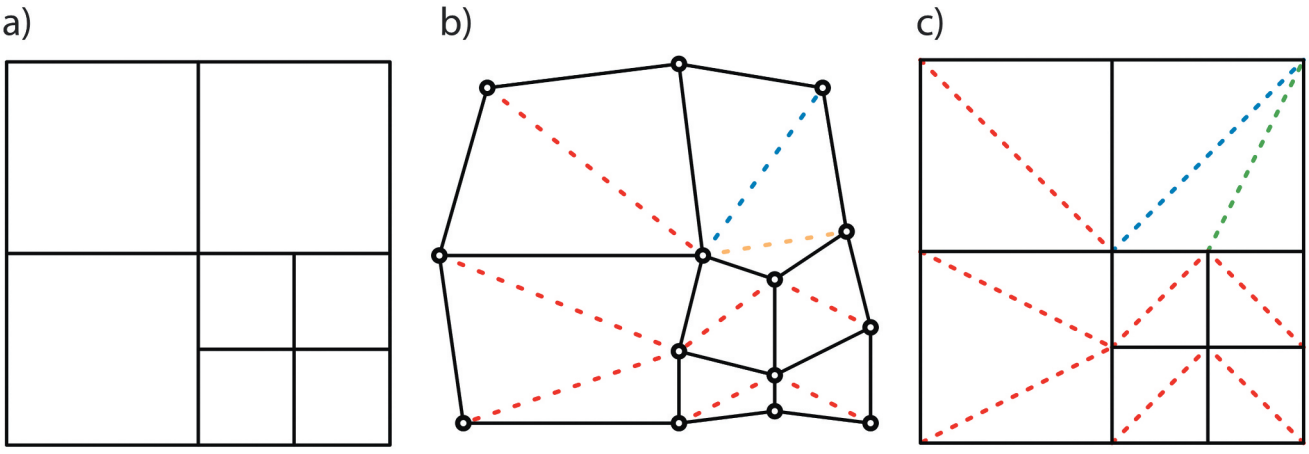


Figure 5. Illustration of the triangulation process. a) Original graded quadtree, as obtained in Figure 4c). b) Quadtree corners projected according to the density grid from Figure 4. The projected quadtree cell edges (black solid lines) serve as constraints for computing a Delaunay triangulation, T_{temp}, of the projected corners. The additional triangulation edges are represented by dashed lines. c) After unprojecting the edges, the final constrained Delaunay triangulation, T, comprises both the solid black lines, as well as the red, green and blue dashed lines. Although the yellow dashed line represents the Delaunay triangulation of the transformed quadtree, we do not insert it into the original quadtree because it overlaps with a boundary of a quadtree leaf node. Thus, we initially only add the blue dashed line as a constraint in the upper right leaf node. Subsequently, the green dashed line is added to complete the Delaunay triangulation under the specified constraints.

4.3. Line densification

After the constrained Delaunay triangulation T is created, we insert points at all intersections of polygon edges with any Delaunay triangle boundary (Figure 6a). The projected coordinates of the original and added points are then interpolated using the uniquely defined piecewise affine transformation that maps the unprojected Delaunay triangles to their respective projections, as shown in Figure 6b).

Because the quadtree is graded, the smallest angle in T is $\arctan(2) - \pi/4 \approx 18.4°$. Consequently, interpolating points within these triangles is numerically more

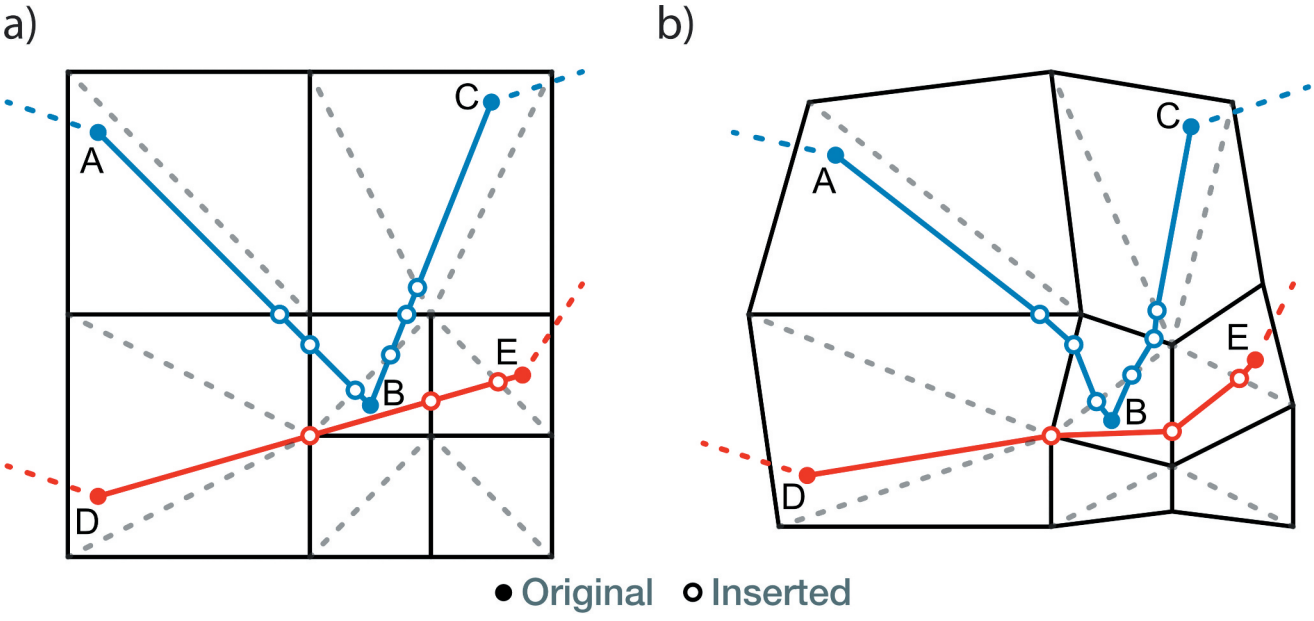


Figure 6. Illustration of our algorithm designed to prevent polyline intersections. a) The positions of the polyline vertices are identical to those in Figure 3a); the quadtree and the Delaunay triangulation are those shown in Figure 5c). Additionally, polyline vertices, represented by open circles, are added at intersections with the Delaunay triangle boundaries. b) When the vertices added in a) are transformed using the same density-equalizing map projection as in Figure 3b), alongside vertices A to E, the red and blue polylines do not intersect. Thus, in contrast to the situation depicted in Figure 3b), the original topology is preserved.

stable than in the potentially strongly elongated triangles that can arise in the Delaunay triangulation of an ungraded quadtree, where angles can approach zero. Figure 7 illustrates the application of the line-densification algorithm to real-world data for Belgium.

Finally, once all points have been transformed, we perform topology-preserving simplification to ensure that the total number of vertices remains within bounds. In our implementation, we use CGAL's simplification module (Fabri, 2026) and set the target number of

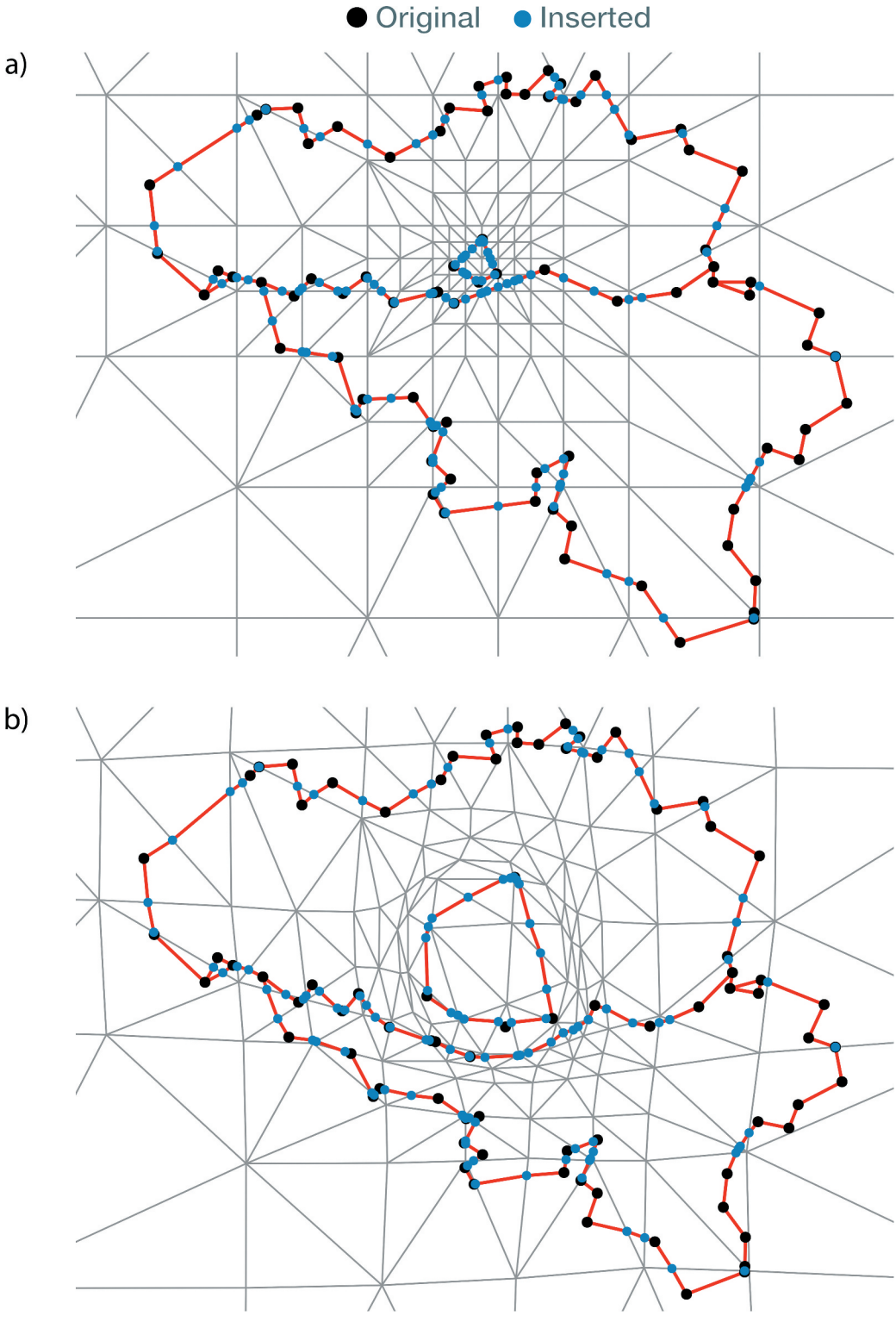


Figure 7. Illustration of Belgium's constrained Delaunay triangulation. Among the various possible triangulations of the quadtree corners in panel a), the selected triangulation maximizes the minimum angle of the projected triangles in panel b).

vertices to either 10,000 or 15 times the number of polygons, whichever is greater, providing a compromise between detail and efficiency.

4.4. Guaranteed topology preservation

To establish that 5FCarto guarantees topologically valid projected polygons, we first recall that density-equalizing map projections are invertible, as explained in Section 2. Assuming that the Jacobian is sufficiently smooth to preserve orientation in the projected triangulation, the affine mappings on adjacent Delaunay triangles align continuously, preventing any intersections along their shared edges. Furthermore, line segments inside the same triangle will not intersect after the projection because the affine transformation of the triangle has a nonzero determinant, establishing properties P1 and P2. Finally, because property P3 follows directly from the orientation-preserving nature of the triangulation, the projected polygons are topologically valid.

5. Evaluation

To evaluate the performance of 5FCarto, we compare it with two alternative approaches: Sun's (2020b) F4Carto software and the original BFB method by Gastner et al. (2018). Figure 8 shows an equal-area map of Singapore alongside population cartograms generated by the three methods. Cartograms are generated from a set of 32 maps using all three methods, and we assess the results using four performance metrics: area error, topological integrity, deformation, and running time. For the tests, we use real-world geometries and data, deliberately selecting a diverse range of examples representative

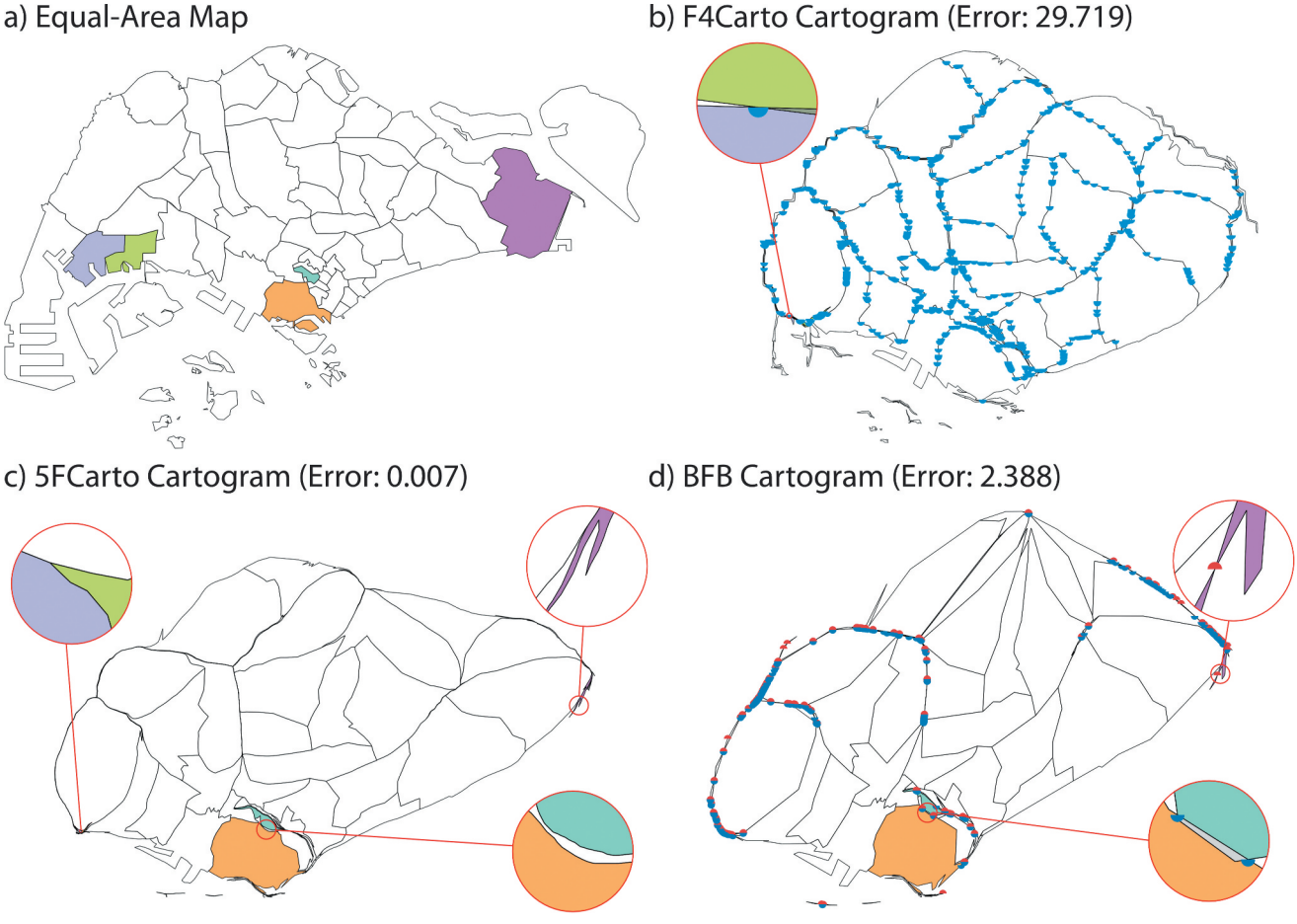


Figure 8. Illustration of Singapore's equal-area geometry, divided into planning areas, and population cartograms generated by F4Carto, 5FCarto, and BFB. The values in parentheses indicate the maximum relative area error for each cartogram. a) Equal-area map of Singapore, provided as input. b) Cartogram generated by F4Carto, with 483 overlapping intersections indicated by blue semicircles. This cartogram exhibits a high maximum relative area error, and it produces nearly rectilinear boundaries that deviate significantly from the true geographic contours. c) Cartogram generated by 5FCarto, with no intersections present. d) Cartogram generated by BFB, displaying 316 self-intersections (red upper semicircles) and 423 overlapping intersections (blue lower semicircles).

of common use cases, to evaluate the difficulty of cartogram generation comprehensively. These test cases include regions with numerous islands and holes, identical maps with different datasets, large and small geographic areas, as well as regions with complex borders.

Let A_i^{actual} denote the actual area of a region i in the input map and A_i^{target} the desired area in the cartogram. The density of the region is defined as

$$\rho_i = \frac{A_i^{\text{target}}}{A_i^{\text{actual}}} .$$

A density value below 1 indicates that a region is meant to shrink, while a value above 1 implies enlargement. Regions with densities significantly lower or higher than 1 require more substantial boundary adjustments, making the cartogram generation process more challenging.

To quantify the difficulty of generating a cartogram for a given dataset, we compute the density disparity, defined as the ratio of the maximum to the minimum density across all regions. A higher density disparity indicates a more challenging transformation. Based on this metric, we divide our 32 maps into four groups: Group 1 contains the easiest cases, while Group 4 comprises the most difficult ones. Table 1 summarizes these groups, while the supplemental Table S1 provides detailed information for all 32 maps.

An algorithm that performs well, particularly on the more challenging groups, can be considered more robust in handling significant deformations in region boundaries.

5.1. Performance metrics

The cartograms were evaluated using four performance metrics: area error, topological integrity, deformation, and running time. In the following sections, we provide theoretical details of each metric.

5.1.1. Area error

We use the maximum relative area error to measure how well the cartogram approximates the target areas. Let A_i^{cart} denote the actual area of region i in the generated cartogram and A_i^{target} the desired target area. The relative area error for each region is defined as the absolute difference between the ratio $A_i^{\text{cart}}/A_i^{\text{target}}$ and 1:

$$e_i = \left| \frac{A_i^{\text{cart}}}{A_i^{\text{target}}} - 1 \right| .$$

The maximum relative area error across all regions is then given by:

$$E_{\max} = \max_i \; e_i .$$

A value of $E_{\max} = 0$ indicates a perfect cartogram where every region exactly matches its target area. In practice, differences below 0.01 can be considered negligible because they are unlikely to be perceptible by the human eye.

5.1.2. Topological integrity

We assess the topological integrity of a cartogram by counting two types of intersections: self-intersections and overlapping intersections. Self-intersections occur when two edges of the same polygon cross, while overlapping intersections occur when edges from different polygons intersect. Ensuring that a cartogram is free of these intersections is particularly challenging, as the process of generating a cartogram involves moving points, and even slight imprecisions can lead to unwanted overlaps or self-intersections. Moreover, once a cartogram exhibits an intersection, it can be difficult to correct. Intersections not only complicate the cartogram's structure but also confuse viewers, who may struggle to determine the true boundaries of the affected regions, a problem impacting, for instance, the online animation by Riedmann and Stöhr (2021).

5.1.3. Deformation

To evaluate the deformation between the cartogram and the original map, we employ three metrics: the Fréchet distance, the Hausdorff distance, and the symmetric difference. Since the purpose of a cartogram is to adjust

Table 1. Summary of the 32 evaluation maps. They are grouped by density disparity, defined as the ratio of maximum to minimum density. Unless stated otherwise, target areas for the cartogram are proportional to population. Further details are available in the supplemental table S1.

Group	Maps	Data	Density disparity
Group 1	8	Austria, Belgium (high and low resolution), England, France (by région), Japan, Germany, Vietnam	<100
Group 2	8	Brazil, Croatia (population and COVID-19 cases), European Union (by country), Malaysia, Switzerland (population and GDP), Tunisia	100–999
Group 3	8	Australia, Bahamas, Egypt, France (by département), Indonesia, Singapore, USA (by state; once including Alaska and Hawaii, once without them)	1,000–9,999
Group 4	8	Algeria, China, European Union (by region), Russia, USA (by county, without Alaska and Hawaii), World (by country), World (by region, with and without Antarctica)	≥10,000

areas relative to its equal-area counterpart, calculating these metrics directly on the input and output regions would not yield meaningful results. Instead, we first normalize the regions by scaling them such that each region has an area equal to 1 and by centering their centroids at (0, 0). This normalization ensures that the metrics effectively quantify how well the cartogram preserves the geographic context of the original regions, enabling users to recognize and interpret them.

The Fréchet distance provides a measure of similarity between two curves by considering both the locations of the points and the order in which they appear along the curves. A common analogy is that of a person walking a dog on a leash: the person follows one curve while the dog follows another, with both allowed to move forward but not backward. The Fréchet distance is the minimum length of leash required for both to complete their respective paths (Buchin et al., 2017), thus capturing the general similarity in the shape and structure of the curves.

The Hausdorff distance measures the maximum deviation between two boundaries. Formally, it is defined as the maximum of the shortest distances from any point on one boundary to the nearest point on the other boundary. This metric is particularly useful for capturing the worst-case discrepancy between the two boundaries (Okabe & Miller, 1996).

The symmetric difference measures the area that is exclusive to either of the two regions. For regions R_1 and R_2, the symmetric difference is given by:

$$\text{SymDiff}(R_1, R_2) = \text{Area}(R_1 \backslash R_2) + \text{Area}(R_2 \backslash R_1).$$

This metric provides an intuitive measure of the divergence between the two shapes by quantifying the area that does not overlap.

To extend these metrics to an entire map or cartogram, which is composed of multiple regions, we compute each metric for every corresponding pair of regions between the original map and the cartogram. Each region may consist of multiple disjoint polygons with holes. When there is a one-to-one relationship between the polygon rings of the equal-area map (both outer boundaries and holes) and those of the cartogram, deformation metrics can be computed for each polygon. However, since F4Carto sometimes removes small polygons and polygons with holes, a one-to-one matching between the disjoint polygons becomes infeasible. Consequently, for the Fréchet and Hausdorff distances, we only consider the polygon with holes having the largest area for each region. We then use the outer ring from both the equal-area map and the cartogram for calculating these distances. In contrast, no such adjustment is necessary for the symmetric difference, allowing the use of all available polygons with holes.

The overall values for each deformation metric (Fréchet and Hausdorff distances, as well as the symmetric difference) are obtained by averaging the values across all regions. This aggregated approach provides a single representative value that captures the collective deformation between the original map and the cartogram. Together, these metrics enable a quantitative assessment of how well the cartogram preserves the geographic features of the original map.

5.1.4. Running time

To evaluate the efficiency of cartogram generation, we consider the total running time of the algorithm as a key metric. Generating a cartogram involves iterative adjustments to the polygonal regions, a process that can become computationally intensive, especially for maps with a high density disparity or complex boundaries. Algorithms with high time complexity may struggle to scale effectively when processing large datasets. To ensure a fair assessment, it is crucial that each algorithm is run on the same machine and, if possible, under identical conditions. Additionally, if parallelization is available, it should be enabled consistently for all algorithms; otherwise, it should be disabled for all. By measuring and reporting the total running time required for cartogram generation, we provide a clear indicator of each method's computational feasibility and its suitability for practical applications.

5.2. Results

In this section, we present our findings for each performance metric across all three algorithms, grouped by difficulty into four levels – Group 1 (smallest density disparities; easiest) through Group 4 (largest density disparities; most challenging) – as specified in Table 1. Additionally, we describe the experimental design to provide context for our evaluation.

5.2.1. Area error

F4Carto's default stopping criterion is to terminate either when the error increases compared to the previous iteration or at iteration 4–whichever occurs first. However, we observed that the error at iteration 4 tends to remain high, and Sun (2020b) notes that F4Carto may sometimes overcorrect, resulting in an initial increase in area error before subsequently improving. To address this issue, we ran all three algorithms for 100 iterations and recorded the best (i.e. minimum) maximum relative area error achieved during those iterations. Note that for

some maps, F4Carto terminated before completing 100 iterations due to errors caused by invalid geometry, whereas both 5FCarto and BFB consistently ran the full course without premature termination. Table 2 provides a groupwise summary, while details on all maps can be found in the supplemental Table S2.

The results demonstrate that, across all groups, the average of the best-case maximum relative area errors, computed per map over all iterations, is consistently lower for 5FCarto than for F4Carto. While F4Carto performs well in the easiest scenarios (Groups 1 and 2), its performance degrades significantly in more challenging cases. In contrast, 5FCarto consistently reduces errors below 10^{-7} regardless of map difficulty, highlighting its robustness.

5.2.2. Topological integrity

To evaluate this metric, we ran all three algorithms using their default parameters, with one modification to the termination criteria for F4Carto: we increased its maximum number of iterations from 4 to 10. This change was made for two reasons. First, the area error often continued to decrease beyond the fourth iteration. Second, Sun's (2020b) example parameter files suggested to use 8 to 12 iterations. In some instances, F4Carto terminated before reaching 10 iterations, as it is configured to stop when the area error increases. By default, 5FCarto and BFB stop once the area of each region in the cartogram differs from its target value by less than 1%. After generating the cartograms, we leveraged functionality from the C++ Boost library (Schäling, 2014) to compute the number of intersections. See Table 3 for a groupwise summary and supplemental Table S3 for details on all maps.

The results indicate that, as the density disparity increases, F4Carto produces an increasing number of overlapping intersections, while BFB exhibits growing numbers of both self-intersections and overlapping intersections. In contrast, 5FCarto consistently achieves zero intersections across all groups.

Additionally, we compared 5FCarto's quadtree-based strategy to ggplot2's more straightforward coord_munch() approach (see Section 3.1), which inserts extra points along every polygon edge until the distance between consecutive vertices falls below a user-specified threshold. For the sample of 32 real-world datasets in Table 1, the coord_munch() strategy introduced intersections in 13 maps, despite increasing the vertex count by an average of 431% (median: 201%) across all maps. In contrast, 5FCarto generated all 32 cartograms free of intersections while increasing the number of vertices by only an average of 62% (median: 20%).

5.2.3. Deformation

For this metric, we ran all three algorithms using the stopping criteria as described in Section 5.2.2). To compute the distance metrics, we employed the C++ Boost library (Schäling, 2014). Table 4 presents a groupwise summary, while supplemental Table S4 provides details on all maps.

Table 2. Average of the best-case (i.e. minimum) maximum relative area error among the first 100 iterations for each algorithm, with maps partitioned into groups based on density disparity – an indicator of cartogram generation difficulty (Group 1 being the easiest and Group 4 the most challenging).

	Averaged best-case maximum relative area error Among first 100 iterations		
Group	F4Carto	5FCarto	BFB
Group 1	0.08	5.426×10^{-15}	$\mathbf{3.594} \times \mathbf{10^{-15}}$
Group 2	0.06795	2.595×10^{-13}	$\mathbf{3.560} \times \mathbf{10^{-14}}$
Group 3	0.9591	$\mathbf{7.827} \times \mathbf{10^{-14}}$	0.09684
Group 4	9.871	$\mathbf{8.967} \times \mathbf{10^{-9}}$	1.002

The best performance is highlighted in **bold**. Note that F4Carto occasionally terminates early due to invalid geometry errors.

Table 3. Average number of self-intersections and overlapping intersections for each algorithm, with maps partitioned into groups based on density disparity – an indicator of cartogram generation difficulty (Group 1 being the easiest and Group 4 the most challenging).

	Self-intersections			Overlap intersections		
Group	F4Carto	5FCarto	BFB	F4Carto	5FCarto	BFB
Group 1	**0.0**	**0.0**	0.5	37.9	**0.0**	0.9
Group 2	**0.0**	**0.0**	0.6	81.8	**0.0**	**0.0**
Group 3	**0.0**	**0.0**	55.8	202.6	**0.0**	61.9
Group 4	**0.0**	**0.0**	198.5	617.4	**0.0**	134.2

The best performance is highlighted in **bold**.

Table 4. Average Fréchet distance, Hausdorff distance, and symmetric difference for each algorithm, with maps partitioned into groups based on density disparity – an indicator of cartogram generation difficulty (Group 1 being the easiest and Group 4 the most challenging).

	Average fréchet distance			Average hausdorff distance			Average symmetric difference		
Group	F4Carto	5FCarto	BFB	F4Carto	5FCarto	BFB	F4Carto	5FCarto	BFB
Group 1	0.66	0.36	**0.34**	0.28	0.24	**0.23**	0.61	**0.47**	**0.47**
Group 2	0.66	**0.51**	**0.51**	**0.31**	0.32	0.32	0.68	**0.64**	0.65
Group 3	**1.01**	1.14	1.85	**0.37**	0.42	1.01	**0.70**	0.75	0.76
Group 4	1.16	**0.95**	1.04	**0.38**	0.46	0.46	**0.66**	0.83	0.85

The best performance is highlighted in **bold**.

The results indicate that the deformations produced by 5FCarto and BFB are comparable across all three deformation metrics. With the exception of the Fréchet distance, on average, F4Carto appears to perform better than both 5FCarto and BFB when density disparity is high.

However, it is important to note that deformation and area error are conflicting minimization objectives. For instance, an algorithm that functions essentially as an identity transformation would produce cartograms with zero deformation but a large area error, rendering the results practically useless. Although F4Carto is not an identity transformation, its area error upon termination remains orders of magnitude higher (5.07 for Group 3 and 39.75 for Group 4) than that of 5FCarto, which, for the purpose of this measurement, was stopped only once the area error dropped below 0.01. This observation suggests that F4Carto's cartograms undergo minimal boundary modifications and may not accurately meet the target areas, whereas 5FCarto's lower area error is achieved through more substantial boundary adjustments that may compromise the preservation of the original geographic features.

5.2.4. Running time

Comparing running times between algorithms is challenging when their stopping criteria differ. By default, both 5FCarto and BFB continue iterating until the area error falls below 0.01, which typically results in longer running times compared to F4Carto. To ensure a fair comparison, we allowed F4Carto to terminate using the stopping criterion described in Section 5.2.2–that is, when its area error first increases relative to the previous iteration or at iteration 10. Then, for both 5FCarto and BFB, we continued execution until their area errors dropped below the level reached by F4Carto or, in cases where they never dropped below this level, until 100 iterations were completed.

All three algorithms were executed on the same machine to maintain a consistent testing environment. F4Carto was run natively on Microsoft Windows 11 Home using a 12th Gen Intel (R) Core(TM) i7-12700 H with 16 GB of RAM. Since 5FCarto and BFB cannot run natively on Windows, they were executed within an Ubuntu container using Docker in Windows Subsystem for Linux on the same hardware. Due to the virtualization overhead, the running times for 5FCarto and BFB are expected to be slower than they would be on a native Linux system with identical hardware. Moreover, although all three algorithms support parallel processing, enabling parallelization in F4Carto often caused it to stop responding during cartogram generation. Therefore, parallelization was disabled for all algorithms to ensure a fair and stable running time comparison. Table 5 provides a groupwise summary, and the supplemental Table S5 contains details on all maps.

The results indicate that, at high density disparities (Group 4), 5FCarto achieves the shortest running times of the three methods. This observation suggests that the overhead of densification is more than offset by faster convergence to accurate areas.

6. Application: 2024 presidential election cartogram of the United States

As a real-world case study, we apply the 5FCarto algorithm proposed in this paper to data from the 2024 United States presidential election. These election results have formed the basis for cartograms published by various news outlets – for example, in articles by ABC News (2024), Levinson et al. (2024), and Breteau and Ferrer (2024)—because such maps eliminate visual overemphasis of land area relative to electoral weight. Figure 9 shows the equal-area reference map used as input to 5FCarto in panel a) and the resulting cartogram in panel b), where areas of all states are proportional to their number of electoral votes.

Because most states use winner-take-all rules, each state is shaded uniformly: red if the Republican ticket won the statewide popular vote, blue if the Democratic ticket won it. Maine and Nebraska are exceptions: they allocate one electoral vote to the winner of each congressional district

Table 5. Average running time and maximum relative area error at the time when F4Carto terminates, with maps partitioned into groups based on density disparity – an indicator of cartogram generation difficulty (Group 1 being the easiest and Group 4 the most challenging). Note that both 5FCarto and BFB can, in principle, achieve substantially smaller area errors, as indicated in table S2 in the supplemental material. However, BFB failed to achieve a lower area error than F4Carto for one map in Group 4 and ran the full 100 iterations, resulting in an unusually large average time.

	Average time (s)			Average maximum relative area error at termination		
Group	F4Carto	5FCarto	BFB	F4Carto	5FCarto	BFB
Group 1	0.296	0.087	**0.068**	0.385	**0.242**	0.244
Group 2	0.617	0.152	**0.082**	0.369	0.266	**0.260**
Group 3	1.219	0.210	**0.179**	5.072	**3.346**	3.740
Group 4	3.737	**0.621**	30.253	39.747	**35.662**	37.129

The best performance is highlighted in **bold**.

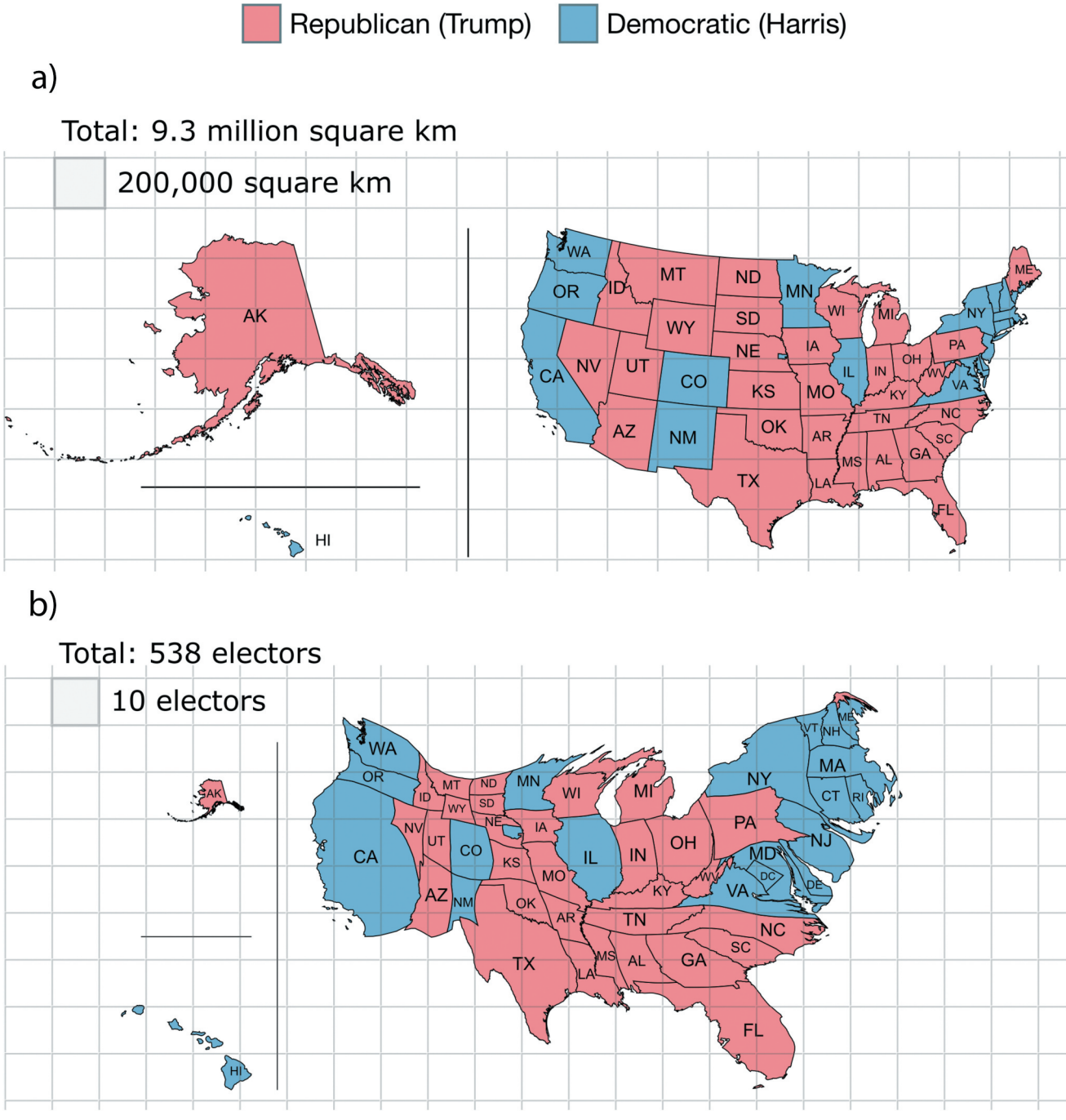


Figure 9. Maps of the United States. Colors represent the party that won the electoral votes in the 2024 presidential election—by state or, in Maine (ME) and Nebraska (NE), congressional district. a) Equal-area map. b) Cartogram generated by 5FCarto, with each region's area proportional to its electoral-vote count.

and two at-large votes to the statewide winner. Consequently, Figure 9 divides both states into two regions, aggregating the congressional districts won by each party; each region is then scaled in proportion to the number of electoral votes that party is awarded in that state.

It is standard cartographic practice to display Alaska and Hawaii – both noncontiguous with the rest of the country – as insets, allowing them to be shifted closer to the other 48 states. We adopted this convention for 5FCarto, depicting both Alaska and Hawaii as insets scaled in proportion to the conterminous United States. Because this scaling is merely an affine transformation, the computational difficulty of cartogram construction is determined almost entirely by the density disparity within the conterminous United States. The resulting value (1,662.45) places this example in Group 3 of Table 1, indicating a moderate computational challenge. Because F4Carto does not support insets, we excluded Alaska and Hawaii from its input and report statistics only for the conterminous United States across all algorithms.

As with all previously tested datasets, 5FCarto's output was free of polygon intersections. While F4Carto and BFB produced no self-intersections, they generated 168 and 2 overlapping intersections, respectively. Adopting the algorithms' default settings, 5FCarto was slightly faster than F4Carto (0.923 versus 0.926 seconds) but slower than BFB (0.829 seconds) due to the required densification and line simplification. Upon termination, 5FCarto reduced the maximum relative area error to 6.21×10^{-3}—the same order of magnitude as BFB (6.43×10^{-3}) but substantially better than F4Carto (0.744). All three algorithms yielded comparable polygon-similarity scores, measured by the Fréchet distance (5FCarto: 0.380; F4Carto: 0.391; BFB: 0.366), Hausdorff distance (0.276, 0.271, 0.264), and symmetric difference (0.512, 0.539, 0.517).

This example corroborates our earlier finding that, among the three algorithms tested, only 5FCarto perfectly preserves the topology. Consistent with previous results, 5FCarto accomplishes this task without substantially increasing running time, area error, or deformation.

7. Conclusion

In this paper, we addressed the issue of topology violations during the transformation of polylines in the construction of contiguous cartograms using density-equalizing map projections. Despite theoretical guarantees, practical implementations often result in intersections due to the approximation of region boundaries with a finite set of points connected by straight lines. We reviewed existing solutions, identified their limitations, and introduced a topology-preserving line-densification algorithm, which leverages the Delaunay triangulation of a graded quadtree to ensure that polylines transform without intersecting.

In our comparative analysis, our line-densification algorithm, 5FCarto, consistently generated cartograms with perfect topological integrity. While F4Carto and BFB may yield acceptable results for simpler maps, their performance degrades under more challenging conditions, with increasing area errors and topological violations. In contrast, 5FCarto can achieve low area errors (below 10^{-7} in all investigated cases; see supplemental Table S2) while still exhibiting running times comparable to all tested alternatives (see supplemental Table S5).

Although our algorithm performs well on challenging datasets, ultimately, as a computer algorithm, it relies on floating-point arithmetic to solve the underlying equations. Because floating-point representations have limited precision, convoluted boundaries or maps with extreme density disparities may still produce overlaps or invalid topologies due to numerical errors. Existing methods may be adapted to correct these floating-point artifacts, such as C++ CGAL's polygon repair module (Ohori, 2026). Adopting higher-precision numeric types may further reduce such errors, albeit at the cost of increased computational expense.

Future research should explore opportunities for further optimization and allowing spherical input geometries alongside Cartesian ones (Li & Aryana, 2018; Lyu et al., 2024). Nonetheless, even in its current implementation, 5FCarto's line densification represents a significant contribution to robust and efficient cartogram generation and may also be applicable to other continuous deformation visualizations in computer-generated imagery.

Acknowledgments

We would like to thank Philemon Heng for initial code development and Edward Liau for curating the test data. We acknowledge the use of ChatGPT for language editing in the final stages of manuscript preparation. We are grateful to the anonymous reviewers whose comments improved this paper.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This project is supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 2 (EP2) programme [MOE-T2EP20221-0007]. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the Ministry.

ORCID

Nihal Z. Miaji http://orcid.org/0000-0003-3459-6677
Adi Singhania http://orcid.org/0000-0001-5193-3343
Callista Le http://orcid.org/0009-0009-6689-9542
Atima Tharatipyakul http://orcid.org/0000-0002-2513-6372
Michael T. Gastner http://orcid.org/0000-0002-1097-8833

Data availability statement

The data that support the findings of this study are openly available in figshare at https://doi.org/10.6084/m9.figshare.28478408.

References

ABC News. (2024). *US election 2024 live results: Donald Trump defeats Kamala Harris, count continues in Arizona and Nevada*. Australian Broadcasting Corporation. Retrieved September 2, 2025, from https://www.abc.net.au/news/2024-11-06/us-election-2024-live-results-electoral-map-who-has-won/104514296

Breteau, P., & Ferrer, M. (2024). *2024 US presidential election: Map of the results, state by state*. Le Monde. Retrieved September 2, 2025, from https://www.lemonde.fr/en/les-decodeurs/article/2024/11/05/2024-presidential-election-map-of-the-results-state-by-state_6731694_8.html

Buchin, K., Buchin, M., Meulemans, W., & Mulzer, W. (2017). Four Soviets walk the dog: Improved bounds for computing the Fréchet distance. *Discrete and Computational Geometry*, *58*(1), 180–216. https://doi.org/10.1007/s00454-017-9878-7

Cano, R. G., Buchin, K., Castermans, T., Pieterse, A., Sonke, W., & Speckmann, B. (2015). Mosaic drawings and cartograms. *Computer Graphics Forum*, *34*(3), 361–370. https://doi.org/10.1111/cgf.12648

Devillers, O., Hornus, S., & Jamin, C. (2024). dD triangulations. In *CGAL user and reference manual* (6.0.1 ed.). CGAL Editorial Board. https://doc.cgal.org/6.0.1/Manual/packages.html#PkgTriangulations

Döll, P. (2017). Cartograms facilitate communication of climate change risks and responsibilities. *Earth's Future*, *5*(12), 1182–1195. https://doi.org/10.1002/2017EF000677

Dorling, D. (1996). Area cartograms: Their use and creation. In Quantitative Methods Study Group of the Institute of British Geographers (Ed.), *Concepts and techniques in modern geography*. Environmental Publications, University of East Anglia.

Dougenik, J. A., Chrisman, N. R., & Niemeyer, D. R. (1985). An algorithm to construct continuous area cartograms. *The Professional Geographer*, *37*(1), 75–81. https://doi.org/10.1111/j.0033-0124.1985.00075.x

Fabri, A. (2026). 2D polyline simplification. In *CGAL user and reference manual* (6.1.1 ed.). CGAL User and Reference Manual. https://doc.cgal.org/6.1.1/Manual/packages.html#PkgPolylineSimplification2

Gastner, M. T., Miaji, N. Z., & Singhania, A. (2022). Smooth pycnophylactic interpolation produced by density-equalising map projections. *Kartografija i Geoinformacije*, *21*(37), 61–69. https://doi.org/10.32909/kg.21.37.3

Gastner, M. T., & Newman, M. E. (2004). Diffusion-based method for producing density-equalizing maps. *Proceedings of the National Academy of Sciences*, *101*(20), 7499–7504. https://doi.org/10.1073/pnas.0400280101

Gastner, M. T., Seguy, V., & More, P. (2018). Fast flow-based algorithm for creating density-equalizing map projections. *Proceedings of the National Academy of Sciences*, *115*(10), E2156–E2164. https://doi.org/10.1073/pnas.1712674115

Gastner, M. T., Shalizi, C. R., & Newman, M. E. (2005). Maps and cartograms of the 2004 US presidential election results. *Advances in Complex Systems*, *8*(1), 117–123. https://doi.org/10.1142/S0219525905000397

Henriques, R., Bacao, F., & Lobo, V. (2009). Carto-SOM: Cartogram creation using self-organizing maps. *International Journal of Geographical Information Science*, *23*(4), 483–511. https://doi.org/10.1080/13658810801958885

House, D. H., & Kocmoud, C. J. (1998). Continuous cartogram construction. In D. Ebert, H. Hagen, H. E. Rushmeier (Eds.), *Proceedings visualization'98 (cat. no. 98cb36276)* (pp. 197–204). ACM Press. https://doi.org/10.1109/VISUAL.1998.745303

Levinson, A., Ketineni, L., & Friedman, D. (2024). *Presidential election 2024: Live results*. The Wall Street Journal. Retrieved September 2, 2025, from https://www.wsj.com/election/2024/general

Li, Z., & Aryana, S. (2018). Diffusion-based cartogram on spheres. *Cartography and Geographic Information Science*, *45*(5), 464–475. https://doi.org/10.1080/15230406.2017.1408033

Lyu, Z., Lui, L. M., & Choi, G. P. T. (2024). Spherical density-equalizing map for genus-0 closed surfaces. *SIAM Journal on Imaging Sciences*, *17*(4), 2110–2141. https://doi.org/10.1137/24M1633911

Nusrat, S., Alam, M. J., & Kobourov, S. (2016). Evaluating cartogram effectiveness. *IEEE Transactions on Visualization and Computer Graphics*, *24*(2), 1077–1090. https://doi.org/10.1109/TVCG.2016.2642109

Ohkita, S., & Tsukamoto, M. (2024). Simple proof of the global inverse function theorem via the hopf–rinow theorem. *Proceedings of the Japan Academy, Series A, Mathematical Sciences*, *100*(3), 17–19. https://doi.org/10.3792/pjaa.100.004

Ohori, K. A. (2026). 2D polygon repair. In *CGAL user and reference manual* (6.1.1 ed.). CGAL Editorial Board. https://doc.cgal.org/6.0.1/Manual/packages.html#PkgPolygonRepair

Okabe, A., & Miller, H. J. (1996). Exact computational methods for calculating distances between objects in a cartographic database. *Cartography and Geographic Information Systems*, *23*(4), 180–195. https://doi.org/10.1559/152304096782438846

Riedmann, B., & Stöhr, M. (2021). *Die Erde in Karten: So haben Sie die Welt noch nicht gesehen*. Der Spiegel. Retrieved February 24, 2025, from https://www.spiegel.de/ausland/die-erde-in-karten-so-haben-sie-die-welt-noch-nicht-gesehen-a-4b03cf99-672c-41f1-a846-1cee641215dd

Sagar, B. S. D. (2014). Cartograms via mathematical morphology. *Information Visualization*, *13*(1), 42–58. https://doi.org/10.1177/1473871613480061

Schäling, B. (2014). *The Boost C++ libraries* (2nd ed.). XML Press.

Selvin, S., Merrill, D., Schulman, J., Sacks, S., Bedell, L., & Wong, L. (1988). Transformations of maps to investigate clusters of disease. *Social Science & Medicine*, *26* (2), 215–221. https://doi.org/10.1016/0277-9536(88)90242-0

Strimas-Mackey, M. (2023). *Smoothr: Smooth and tidy spatial features*. R package version. Retrieved May 30, 2024, from https://CRAN.R-project.org/package=smoothr

Sun, S. (2013). A fast, free-form rubber-sheet algorithm for contiguous area cartograms. *International Journal of Geographical Information Science*, *27*(3), 567–593. https://doi.org/10.1080/13658816.2012.709247

Sun, S. (2020a). Applying forces to generate cartograms: A fast and flexible transformation framework. *Cartography and Geographic Information Science*, *47*(5), 381–399. https://doi.org/10.1080/15230406.2020.1745092

Sun, S. (2020b). *Software and example data for F4Carto: A fast and flexible force-based transformation framework for cartograms*. Harvard Dataverse. Retrieved February 27, 2025 https://doi.org/10.7910/DVN/6ST8NQ

Tobler, W. (2004). Thirty five years of computer cartograms. *Annals of the Association of American Geographers*, *94*(1), 58–73. https://doi.org/10.1111/j.1467-8306.2004.09401004.x

Tobler, W. (2017). Cartograms as map projections. In M. Lapaine & E. L. Usery (Eds.), *Choosing a map projection* (pp. 149–159). Springer International Publishing. https://doi.org/10.1007/978-3-319-51835-0_5

Tobler, W. R. (1963). Geographic area and map projections. *Geographical Review*, *53*(1), 59–78. https://doi.org/10.2307/212809

Wickham, H. (2016). *Ggplot2: Elegant graphics for data analysis*. Springer-Verlag. https://doi.org/10.1007/978-3-319-24277-4