# Multicriteria Optimization for Dynamic Demers Cartograms

Soeren Nickel, Max Sondag, Wouter Meulemans, Stephen Kobourov, Jaakko Peltonen, and Martin Nöllenburg

**Abstract**—Cartograms are popular for visualizing numerical data for administrative regions in thematic maps. When there are multiple data values per region (over time or from different datasets) shown as animated or juxtaposed cartograms, preserving the viewer's mental map in terms of stability between multiple cartograms is another important criterion alongside traditional cartogram criteria such as maintaining adjacencies. We present a method to compute stable stable Demers cartograms, where each region is shown as a square scaled proportionally to the given numerical data and similar data yield similar cartograms. We enforce orthogonal separation constraints using linear programming, and measure quality in terms of keeping adjacent regions close (cartogram quality) and using similar positions for a region between the different data values (stability). Our method guarantees the ability to connect most lost adjacencies with minimal-length planar orthogonal polylines. Experiments show that our method yields good quality and stability on multiple quality criteria.

**Index Terms**—Time-varying data, cartograms, mental map preservation

## 1 Introduction

Many datasets are georeferenced and relate to places or regions. A natural way to visualize such spatial data is to use cartographic maps. One prominent tool is the choropleth map, which colors each region in a map based on its data value. While choropleth maps work well for data that correlates to region sizes, when the data is not correlated it has the drawback that the visual salience of large and small regions is unequal. Moreover, it is difficult to compare colors to each other, and colors are not the most effective encoding for numeric data [1], requiring a legend to facilitate relative assessment.

One way to overcome these drawbacks is with cartograms, which reduce spatial precision in favor of clearer encoding of data values: the map is deformed such that each region's visual size is proportional to its data value. The visual salience of a region then correspond to its data value, and comparison of magnitudes becomes a task of estimating area – which is a more effective encoding for numeric data [1]. Additionally, the freed up color channel can be used to visualize secondary data. Cartogram quality is assessed by multiple criteria [2] including *1. Spatial deformation:* regions should be placed close to their geographic position; *2. Shape deformation:* regions should resemble their geographic shape; *3. Preservation of relative directions:* spatial relations such as north-south and east-west should be maintained. *4. Topological accuracy:* geographically adjacent regions should be adjacent in the cartogram, and vice versa. *5. Cartographic error:* relative region sizes should be close to the data values. Criteria 1-4 describe geographical accuracy of the region arrangement, criterion 5 (also called statistical error) captures how well data values are represented. Many techniques aim at (near-)zero cartographic error, often at the expense of other criteria.

These criteria evaluate a single cartogram, but multiple cartograms of the same regions can be used to visualize for example dynamic data such as yearly census data, or different demographic variables that we want to explore, compare and relate, yielding a vector or set of values for each region. Example visualizations for multiple cartograms include animations (especially for time series; see [3] for an early example), small multiples showing a matrix of cartograms, or letting a user interactively switch the mapped value in one cartogram. In these cases, we want the cartograms to be *stable*: cartograms of the same regions for different data values should have as similar of a layout as the data values allow. A small change in the data values should result in a small change in the layout. This helps the viewer to retain their mental map [4], supporting linking and tracking across cartograms. Thus, we obtain an additional important criterion with multivariate or time-varying data. *6.*


- *Soeren Nickel and Martin Nöllenburg are with TU Wien, 1040 Vienna, Austria. E-mail: soeren.nickel@tuwien.ac.at, noellenburg@ac.tuwien.ac.at.*
- *Max Sondag is with Swansea University, SA2 8PP Swansea, Wales, U.K. E-mail: m.f.m.sondag@swansea.ac.uk.*
- *Wouter Meulemans is with TU Eindhoven, 5612, AZ, Eindhoven, The Netherlands. E-mail: w.meulemans@tue.nl.*
- *Stephen Kobourov is with the University of Arizona, Tucson, AZ 85721 USA. E-mail: kobourov@cs.arizona.edu.*
- *Jaakko Peltonen is with Tampere University, 33100 Tampere, Finland. E-mail: jaakko.peltonen@tuni.fi.*


*Manuscript received 19 July 2021; revised 14 Jan. 2022; accepted 27 Jan. 2022. Date of publication 14 Feb. 2022; date of current version 2 May 2022.*
*The work of Max Sondag was supported in part by The Netherlands Organisation for Scientific Research (NWO) under Grants 639.023.20. The work of Stephen Kobourov was supported in part by NSF under Grants CCF-1740858, CCF-1712119, and DMS-1839274. The work of Martin Nöllenburg was supported in part by FWF under Grant P 31119. The work of Jaakko Peltonen was supported in part by Academy of Finland decision under Grant 327352.*
*(Corresponding author: Soeren Nickel.)*
*Recommended for acceptance by J. Fekete.*
*Digital Object Identifier no. 10.1109/TVCG.2022.3151227*

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

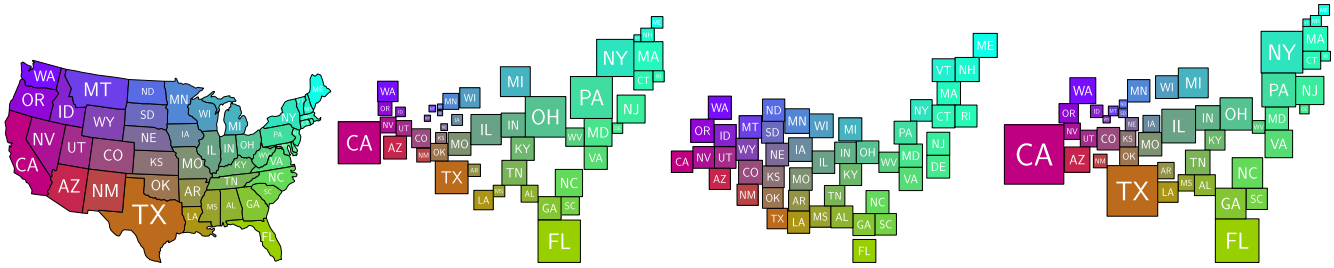


Fig. 1. Map of the contiguous US and three Demers cartograms: drug poisoning mortality, election turnout and population in 2016. The layout minimizes distance between adjacent regions. A two-dimensional color-gradient is used to facilitate correspondences.

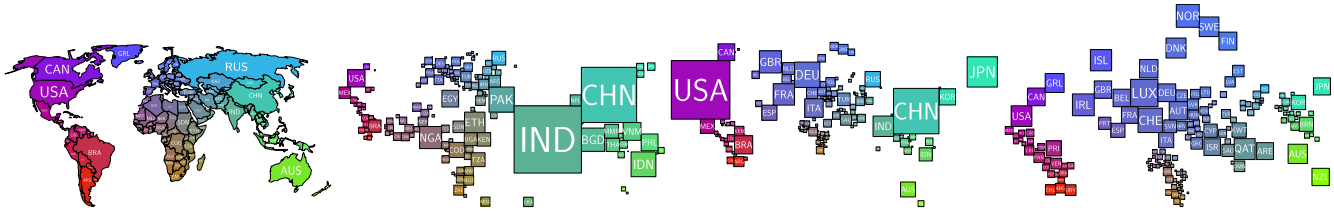


Fig. 2. Map of the world and three Demers cartograms: forest area, total GDP and GDP per capita in 2016. The layout minimizes distance between adjacent regions. A two-dimensional color-gradient is used to facilitate correspondences.

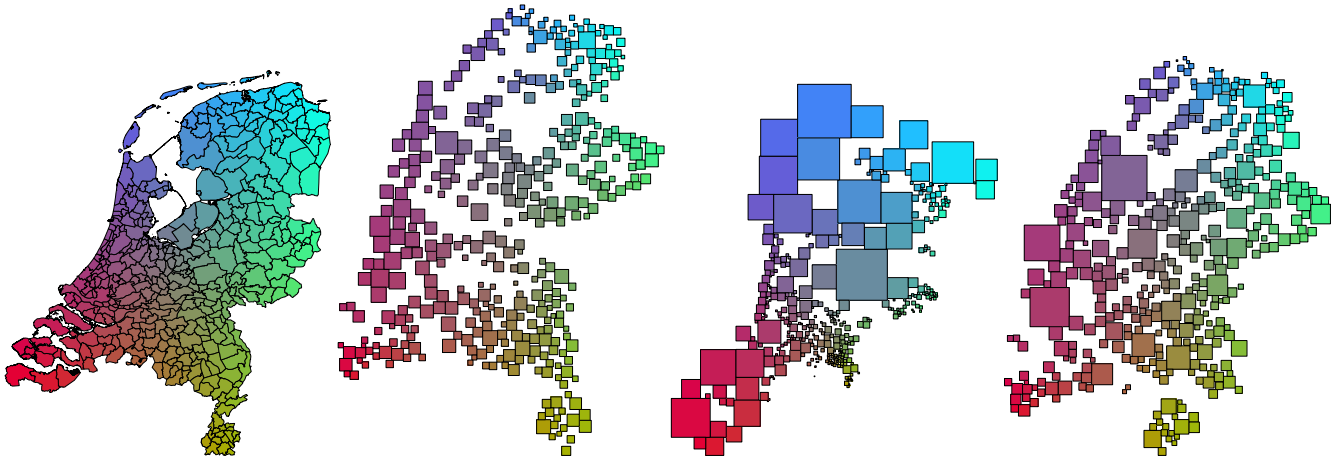


Fig. 3. Map of the Netherlands and three Demers cartograms: house density, percentage of water surface and population in 2017. The layout minimizes distance between adjacent regions. A two-dimensional color-gradient is used to facilitate correspondences. Region labels are omitted.

*Stability*: cartograms for the same regions with similar data values should have similar layouts.

The relative importance of criteria depends on the tasks [2] to be facilitated, and trade-offs between criteria are often required. Different cartogram types are more suitable for different tasks. We focus on *Demers cartograms* (DC) [5] which represent each region by a square whose area exactly matches the data value (criterion 5), similar to Dorling cartograms [6]. By using squares, DCs facilitate easy comparison of data values as aspect ratio is no longer a factor, unlike, e.g., for rectangular cartograms [7]. Moreover, a DC has no overlap and data values are thus not obfuscated. As abstract squares incur shape deformation (2), in spatial recognition tasks the cartogram layout as a whole must be informative instead. Therefore, the layout must optimize the other geographic criteria: *spatial deformation* (1), *preservation of relative directions* (3) and *topological accuracy* (4).

*Contributions.* Our primary contribution is a linear programming (LP) algorithm to compute high-quality stable DCs for dynamic data. To the best of our knowledge, this is the first fully automated approach for generating DCs that guarantees non-overlapping squares. Our DCs have no cartographic error (5), satisfy given constraints on spatial relations (3), and allow trade-offs between the topological error (4) and stability (6). Linear interpolation between different DCs generated by our LP yield an overlap-free transformation suitable for animation. Lost adjacencies can be shown as minimal-length planar orthogonal polylines called *leaders* under a mild assumption, connecting two regions that are adjacent in the map. Our experiments compare settings of our LP to each other and to a force-directed layout we introduce (also novel for DCs) to compare different settings for the LP. The results show that our LP efficiently computes stable DCs; see Figs. 1, 2, and 3 for example DCs computed by our method using the recommended default setting.

*Organization.* After a brief discussion of related work below, in Section 2 we discuss the problem formalization. Subsequently, we first solve the problem optimally for a single cartogram in Section 3, before we consider multiple weight functions and include stability into our method in Section 4. In Sections 5 and 6, we discuss the setup and results of our experiments. In Section 7 we briefly consider

how our results may be used in a visualization system and the possibility of leaders. We close in Section 8 with a brief reflection on our results and future work.

*Related Work.* Cartogram-like representations date back to the 1800s. By the 1900s, most standard cartogram types we use now were defined, including early rectangular value-by-area cartograms [8]. The first automatically generated cartograms were continuous deformation cartograms from the 1970s [9] which have been [10], [11], and are still being improved upon [12], [13]. Dorling [6] and Demers cartograms [5] exemplify the non-contiguous type, where the regions are not necessarily connected. Layouts representing regions by rectangles or rectilinear polygons have received much attention in the algorithmic literature [14], [15], [16], typically focussing on aspect ratio, topological error and region complexity. Area-universal rectangular layouts [16] accommodate arbitrary areas without changing their combinatorial structure including the topology. Such layouts are largely stable, but not every map admits such a layout. Compared to DCs, rectilinear variants can better retain shapes, but have higher visual complexity and assessing areas becomes more difficult; Mosaic Cartograms [17] aim to overcome such issues by using tiles to make sizes countable, but this results in a higher visual complexity again. Spatial deformation and cartographic error are (usually) in direct conflict and cannot both be achieved perfectly.

DCs also relate to drawing contact representations of graphs, where adjacencies between neighboring regions are encoded as touching shapes. The focus in the graph drawing literature is on recognizing which graphs can be perfectly represented. However, Even for the unit-disk case this is NP-hard [18] to determine, but efficient algorithms exist for some restricted graph classes [19]. Klemz *et al.* [20] for example considered a vertex-weighted variant using varying sized disks. Various other techniques are similar to DCs, using squares or rectangles for geospatial information. Examples include algorithms and computational experiments for grid maps [21], [22], [23]. Of particular relevance is the work of Inoue for circle [24] and simple-shaped [25] area cartograms, which they construct using a *non-linear* constraint programs. Meulemans [26] recently introduced a *linear* constraint program to compute optimal solutions under orthogonal order constraints for diamond-shaped symbols. We use a similar technique to Meulemans, but defer a discussion of the differences to the next section, after we have formally defined our problem.

Several measures for evaluating the quality of cartogram types and algorithms have been proposed [27], [28], but there is little work on evaluating or computing stable cartograms for time-varying or multivariate data. Yet they are used in such manner, e.g., as a sequence of contiguous cartograms showing the evolution of the Internet [29]. There has however been recent attention on stability in algorithms and visualization for both spatial [30], [31], [32] and nonspatial data [33], [34] that can be used as a basis.

## 2 Problem Definition

The problem of computing a DC can be formally defined as follows. We are given an input graph $G = (R, T)$ representing the topology of the underlying map. Each *region* $r \in R$ has a centroid in $\mathbb{R}^2$ and a weight $w(r)$, which corresponds to the side length of the square in the output. A pair of regions $(r, r')$ has an edge in $T$ if and only if the regions are adjacent on the original map. The output – a placement of a square $s$ for each region $r$ – is stored as a point $P : R \to \mathbb{R}^2$ for each region, encoding the center of its square. All squares must be pairwise disjoint.

For the multivariate or time-varying case, we assume that each region $r$ has a different weight $w_i(r)$ for each dimension or time step $i$. The centroid is assumed to not change[1], but regions do not need to be present in each dimension or time step – this is particularly relevant for time-varying data as the administrative units that determine the data partitioning (e.g., municipalities) may change over time. The output is then position $P_i(r)$ per dimension or time step in which the region is present.

This gives us the basic setup for DCs. Below, we discuss quality criteria, additional constraints, and the relation of our setup to Dorling Cartograms and overlap removal.

### 2.1 Quality Criteria

We restrict our attention here to a high-level discussion of the quality criteria mentioned in Section 1. How these are specifically measured varies slightly between algorithms and experiments which we will formalize in Section 5.

Following many existing techniques for cartograms, we focus on topological accuracy as the primary objective, i.e., we want adjacent regions in the input to geometrically touch in the output and vice versa. Instead of considering such a binary constraint, we consider the distance between regions that should be adjacent. As we shall see, this makes a huge difference in problem complexity, and has the further advantage to be more nuanced in considering the general neighborhood of a region in the cartogram – that is, the size of gaps between non-adjacent regions that should be adjacent matters. We consider the spatial deformation and preservation of relative directions as secondary quality criteria in our algorithms. Cartographic error is zero as all shapes are squares of the appropriate size.

Finally, for the multivariate or time-varying case, we want our solution to be stable. We quantify this primarily through the distance (or movement) between the position of a region in cartograms for the different weight functions. However, for our experiments we shall also consider stability in terms of relative directions between regions.

### 2.2 Separation Constraints

While the definition above is sufficient to define a DC, to get high quality DCs we add the notion of separation constraints to the input specification, which represent the "relative directions" between regions, i.e., North, East, South, West. To that end, we are additionally given two sets $H$, $V$ of ordered region pairs. A pair of regions $(r, r')$ is in $H$, if $r$ should be horizontally separated from $r'$ such that there exists a vertical line $\ell$ with the square of $r$ being left of $\ell$ and $r'$ to its right. Analogously, $V$ encodes the vertical separation requirements. If $r$ and $r'$ are adjacent, then $(r, r')$ is either in $H$ or in $V$ (but not in both) and they *should* touch $\ell$,

1. Algorithmically, this could easily be supported though.

otherwise we require a strict separation gap to avoid false adjacencies; we are given a minimum gap $\varepsilon > 0$ to ensure that this non-adjacency can be visually recognized.[2] The sets $H$ and $V$ model the relative direction criterion for DCs and any two regions are paired in at least one of those sets. To ensure a DC exists that satisfies the separation constraints, the directed graphs $D_H = (R, H)$ and $D_V = (R, V)$ must be *acyclic* (DAGs). We consider these relations transitively: if $(r, r') \in H$ and $(r', r'') \in H$, then this enforces a vertical line separating $(r, r'')$ in any DC and thus $(r, r'')$ is in $H$.

Our output now has to ensure that these separation constraints are maintained when placing the squares. A placement $P$ is *valid*, if it satisfies the separation constraints of $H$ and $V$. This implies that all squares are pairwise interior disjoint and fully disjoint for nonadjacent regions.

*Deriving Separation Constraints.* The region weights are given and adjacencies and centroids are easily derived from a map, but separation constraints $H$ and $V$ are not. Various models can be used to determine good directions or separation constraints [35], we use the following simple model; it is symmetric and ensures that constraints form a DAG.

For two regions $(r, r')$ represented by their centroids, we check whether their horizontal or vertical distance is larger. In the former case, we add $(r, r')$ to $H$ if $r$ is left of $r'$ and $(r', r)$ to $H$ otherwise. In the latter case, we add the pair similarly to $V$. We call this the *weak setting*. Constraints added in this setting are *primary separation constraints*.

In the *strong setting*, we add extra constraints for nonadjacent region pairs whose bounding boxes admit both horizontal and vertical separating lines: if a pair has a primary separation constraint in $H$ or $V$, we add a *secondary* separation constraint to $V$ or $H$ respectively.

The lemma below matches an observation from [26] that carries over to our setting. It implies that DCs for different weight functions but with the same constraints have a smooth and simple transition between any such DCs helping to retain the user's mental map.

**Lemma 1 ([26]).** *Let $R$ be a set of regions with separation constraints $H$ and $V$. Let $A$ and $B$ be two DCs for $R$, both satisfying $H$ and $V$. Then, any linear interpolation between $A$ to $B$ also satisfies $H$ and $V$ and is thus overlap-free.*

### 2.3 Comparison to Related Problems

*Overlap Removal.* In [26], a technique is presented to remove overlap between squares, based on maintaining structure between the square centers and the resulting implied separation constraints, while minimally displacing these squares from their original location. It hence differs in objective from our case, as we optimize for topology and relative positions. In terms of feasible placements, we observe that the algorithm of [26] uses a different set of constraints compared to our strong setting, though its "weak order constraints" coincides with our weak setting; see Fig. 4.

Extensions in [26] can be applied in our scenario, e.g., reducing actively considered separation constraints by removing transitive relations ("dominance" in [26]). Time-varying data is briefly considered in [26], but they only conceptualize a trade-off between displacement and stability for artificial data; we discuss several optimization criteria, also focusing on adjacencies which are not considered in [26], use real-world data in our experiments, and compare to a baseline DC implementation to move beyond the limits of linear programming.

2. In our implementation $\varepsilon$ is the minimum of the smallest side length and 5% of the diagonal of the bounding box of the input regions.

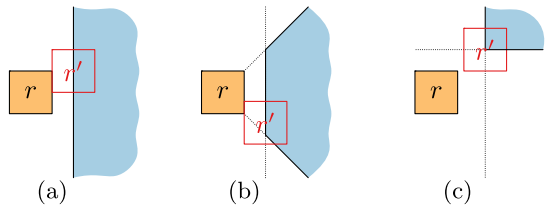



Fig. 4. Feasibility area (in blue) where $r'$ may be placed w.r.t $r$, when $r'$ is primarily to the right of $r$. (a) Our weak setting and the weak order constraints in [26] coincide. (b) The ("orthogonal") order constraint in [26]. (c) Feasibility in our strong setting.

*Dorling Cartograms.* We observe that a similar problem definition can be used for Dorling Cartograms, using $w(r)$ as the radius of the disk representing $r$ in the cartogram. However, Dorling Cartograms and DCs are in fact two different representations, and neither can perfectly represent all graphs that the other can. Take a graph $G$, with eight regions connected to a central region $r_1$ with no other adjacencies and no separation constraints. Assign a weight of 1 to $r_1$ and a weight of $1 - \epsilon$ to all other regions with $\epsilon$ approaching zero. A DC can perfectly represent this by positioning two regions on each side of $r_1$. In a Dorling cartogram however, we cannot do better than six circles that touch $r_1$. Conversely, if there are five regions connected to $r_1$ and the weight of these regions is $1 + \epsilon$, a Dorling cartogram can perfectly represent it, but a DC can maintain at most four adjacencies, as there cannot be two squares with side length $1 + \epsilon$ adjacent on the same side.

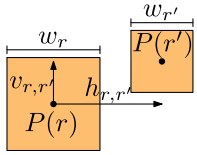
We further observe that our linear program given in the next sections results in optimal solutions, following natural separation constraints, that generally allow the resulting squares to touch to represent adjacencies. As mentioned in [26], these techniques can also be applied to other shapes such as circles, if separation constraints are chosen suitably. However, for the circles of a Dorling cartogram, such separation constraints are somewhat artificial, not allowing two circles to touch (unless perfectly positioned) and thus generally showing gaps where none would be expected.

## 3 Computing a Single DC

First, we consider a DC for a single weight function. We compute a layout realizing the weights with disjoint squares that may touch only if adjacent, such that the separation constraints are maintained. We quantify the layout quality via the distances between any two squares representing adjacent regions. We show that the problem, under appropriate distance measures, can be solved optimally via a linear program (LP) in polynomial time.

*Linear Program.* For each $r \in R$, we introduce variables $x_r$ and $y_r$ for the center $P(r) = (x_r, y_r)$ of the square $s(r)$. For any originally adjacent regions $\{r, r'\} \in T$ we introduce variables $h_{r,r'}$ and $v_{r,r'}$ for the respectively (non-negative)

horizontal and vertical distance between the two squares $r$ and $r'$. For any two regions $r, r'$, we let $w_{r,r'} = \frac{(w(r)+w(r'))}{2}$. This is illustrated on the right. Finally, we let $gap_{r,r'} = 0$ if $\{r, r'\} \in T$ and $gap_{r,r'} = \varepsilon$ otherwise. We use the following constrained optimization objective




$$\min \sum_{\{r,r'\} \in T} (h_{r,r'} + v_{r,r'}) \quad (1)$$

$$x_{r'} - x_r \geq w_{r,r'} + gap_{r,r'} \qquad \forall_{r,r'} \in H \quad (2)$$

$$y_{r'} - y_r \geq w_{r,r'} + gap_{r,r'} \qquad \forall_{r,r'} \in V \quad (3)$$

$$h_{r,r'}, v_{r,r'} \geq 0 \qquad \forall_{r,r'} \in T \quad (4)$$

$$h_{r,r'} \geq \max\{x_r - x_{r'}, x_{r'} - x_r\} - w_{r,r'} \qquad \forall_{r,r'} \in T \quad (5)$$

$$v_{r,r'} \geq \max\{y_r - y_{r'}, y_{r'} - y_r\} - w_{r,r'} \qquad \forall_{r,r'} \in T \quad (6)$$

The objective (1) minimizes the sum of distances between adjacent regions in the $L_1$ metric. Constraints (2) and (3) ensure the separation requirements by forcing the centers of relevant pairs of squares far enough apart. For nonadjacent regions, the *gap* function assures a recognizable gap of width $\varepsilon$ between resulting squares. Constraints (4)–(6) bind distance variables $h, v$ with positional variables $x, y$. Here, (5) and (6) encode two linear constraints per line, one for each term in the 'max' function. As (1) minimizes the distances, it suffices to enforce lower bounds, hence the '$\geq$' in the constraints. In an optimal solution, either one of the two versions, or the non-negativity constraint (4) will be satisfied with equality.

*Improving Adjacencies.* The above model has two minor flaws. First, two squares 'touch' even if they only do so at corners; we resolve this by adding $\delta_{r,r'} = 0.25 \cdot \min(w(r), w(r'))$ to the right-hand side of (5) and (6), to promote overlapping sides. Specifically, this change allows $h_{r,r'} = 0$ ($v_{r,r'} = 0$), when squares share a segment at least $\delta$ long. Second, in the strong setting the above LP asks for a minimum gap of size $gap_{r,r'} = \epsilon$ along both axes for two non-adjacent regions $r, r'$. This is not needed for visual separation, thus when using the strong setting we set $gap_(r, r') = 0$ instead of $\epsilon$ for the secondary separation constraint (which can be either in $H$ or $V$).

*Fine-Tuning the Optimization Criteria.* The above LP minimizes the sum of distances between adjacent regions. The cartogram literature however emphasizes *counting* lost adjacencies between regions in a binary way. We prefer our measure since: (1) there is a big difference if two neighboring regions are set apart by a small or large gap; (2) while the LP can be turned into an *integer* linear program to count lost adjacencies, it greatly increases computational complexity—optimizing for adjacencies is typically NP-hard, e.g., for disks [18], [36] or segments [37].

Our LP generally admits several optimal solutions, due to translation invariance and as touching squares may slide freely along each other as long as they touch. We can introduce a *secondary* term to the objective to nuance selection of preferred layouts, multiplied by $c_{r,r'} = c \cdot a_{r,r'}$, where $c$ is a small constant and $a_{r,r'} = 1$ if $r$ and $r'$ are adjacent and $a_{r,r'} = 0.1$ otherwise, to not interfere with the original (primary) objective. Our secondary term optimizes preservation of relative directions between squares within the freedom of the optimal solution.

Consider two regions $r$ and $r'$. We assume, without loss of generality, that $(r, r') \in H$ is the primary separation constraint; the case $(r, r') \in V$ is handled analogously. We compute a directional deviation $d_{r,r'} = |(y_r + \alpha(x_{r'} - x_r)) - y_{r'}|$, where $\alpha$ is the (finite) slope of the ray from $r$ to $r'$ in the input graph $G$. Similar to (5), the objective function will minimize $d_{r,r'}$; we emphasize this term with a higher weight for adjacent regions via $c_{r,r'}$. We thus turn the above formula into two linear inequalities. Below is the full LP; note that the first three constraints are identical to the constraints presented earlier

$$\min \sum_{\{r,r'\} \in T} (h_{r,r'} + v_{r,r'}) + \sum_{\{r,r'\} \in R^2} (c_{r,r'} \cdot d_{r,r'}) \quad (7)$$

$$x_{r'} - x_r \geq w_{r,r'} + gap_{r,r'} \qquad \forall_{r,r'} \in H \quad (2)$$

$$y_{r'} - y_r \geq w_{r,r'} + gap_{r,r'} \qquad \forall_{r,r'} \in V \quad (3)$$

$$h_{r,r'}, v_{r,r'} \geq 0 \qquad \forall_{r,r'} \in T \quad (4)$$

$$h_{r,r'} \geq \max\{x_r - x_{r'}, x_{r'} - x_r\} - w_{r,r'} + \delta_{r,r'} \qquad \forall_{r,r'} \in T \quad (8)$$

$$v_{r,r'} \geq \max\{y_r - y_{r'}, y_{r'} - y_r\} - w_{r,r'} + \delta_{r,r'} \qquad \forall_{r,r'} \in T \quad (9)$$

$$d_{r,r'} \geq (y_r + \alpha(x_{r'} - x_r)) - y_{r'} \qquad \forall_{r,r'} \in H \quad (10)$$

$$d_{r,r'} \geq y_{r'} - (y_r + \alpha(x_{r'} - x_r)) \qquad \forall_{r,r'} \in H \quad (11)$$

$$d_{r,r'} \geq (x_r + 1/\alpha(y_{r'} - y_r)) - x_{r'} \qquad \forall_{r,r'} \in V \quad (12)$$

$$d_{r,r'} \geq x_{r'} - (x_r + 1/\alpha(y_{r'} - y_r)) \qquad \forall_{r,r'} \in V \quad (13)$$

Alternatives exist for the primary optimization criterion: displacement from the original location helps find layouts maintaining many adjacencies for grid maps of equal-size squares [21], [22]. For each region we measure $L_1$ displacement from its *origin* (centroid of the original region in the geographic map) to the square center $P(r)$.

## 4 Computing Stable DCs

Our method can be extended for regions having multiple weights. In this setting, we are given a set of weight functions $W = \{w_1, \ldots, w_k\}$. We aim to compute a DC for each $w_i \in W$, i.e., positions $P_i(r)$ for each $r \in R$ and $w_i \in W$. If each weight function represents the same data semantic, say population size, we consider $W$ ordered by the $k$ time steps; we call this setting *time series*. If each weight function represents measurements of different data semantics, say population and gross domestic product, we treat $W$ as an unordered set; we call this setting *weight vectors*.

*Stability Models.* Before we discuss how to extend our LP to include multiple weights, we first consider how we may actually achieve stability. Roughly speaking, we identify two options: (1) an iterative setting in which we compute a layout for one weight function $w_i$, and afterwards compute a layout for a next weight function $w_{i+1}$, including stability to the now-fixed layout $P_i$; (2) a concurrent setting in which we combine the computation of the layouts for two or more weight functions while also incorporating their stability. Though concurrent computation seems beneficial for overall quality, this leads to higher and sometimes excessive computational complexity as $k$ grows.

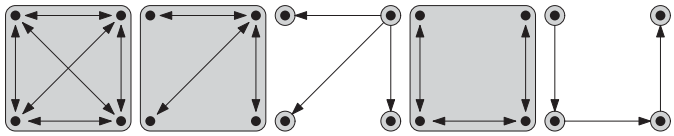


Fig. 5. Graphs defined on four weight functions, showcasing, from left to right, the C, St, StIt, Su and SuIt stability models. The gray boxes indicate an independent LP, solving the included weight functions.

We generally capture this idea through a *stability model*: a directed graph $S = ([k], I)$ on the index set $[k] = \{1, \ldots, k\}$ with $I \subseteq [k]^2$. We interpret $S$ as follows: an edge $(i, j)$ means we want to consider the stability when going from $P_i$ to $P_j$. If $i$ cannot be reached from $j$, this means that the layout $P_i$ can be computed and fixed, before computing the layout $P_j$: an iterative dependency. However, if both are reachable from each other, then the weight functions are to be dealt with concurrently. We may identify strongly connected components of $S$ in linear time [38]. We identify five models that we study experimentally; see also Fig. 5.

*(C)* $S$ is the **c**omplete graph. The C (**c**omplete) model aims to capture that all pairwise stabilities are important. This is generally useful for weight vectors but also time series in a small-multiples setting or when nonconsecutive time steps are to be compared. However, this model will lead to a high complexity.

*(St)* $S$ is a *st*ar with a single central weight function with bidirectional (St) or outgoing (StIt: *st*ar *it*erative) edges to all other functions. The St model mimics a similar principle as the C model, but with a single central weight function, reducing the model complexity: if all other weight functions are stable with respect to the central function, they cannot be too dissimilar pairwise – any path in $S$ has length at most two. For StIt the other LPs can be run in parallel after computing the layout for the central weight function.

*(Su)* $S$ is a single path of *su*ccessive weight functions, either bidirectional (Su) or directed (SuIt: *su*ccessive *it*erative). Su models are mostly natural for time-series data in animation, where the main concern is stability between consecutive time steps. The layout between time steps further apart may show larger differences as the paths in $S$ get longer. This is useful if the data exhibits drift in overall values (e.g., annual population over a century), such that a good layout for later time steps does not hinder a good layout for earlier time steps. For SuIt, while we can not compute the layouts in parallel, every single layout is computed on its own, depending only on the previous layout, which reduces the complexity of computing these layouts.

*Extending the LP.* Let us now turn to extending the LP to include multiple weight functions. This LP is set up for every strongly connected component of $S$. LPs belonging to weakly connected components are solved in the order of a topological sorting on the strongly connected components. Disconnected components could be computed in parallel. Here we assume that $S = ([k], I)$ is one such component. Our goal is to compute the centers $P_i(r)$ for each region $r$ and weight function $w_i \in W$ while incorporating stability between the layout for each function. To do so, we add constraints and optimization objectives for stability while re-using our previously presented LP. We extend the notation of Section 3 with superscript $i$ to denote the respective variables for weight function $w_i \in W$. We change objective (1) and add constraints to minimize displacement between centers of the same region for different weight functions as prescribed by stability model $S$. We treat $I$ as undirected[3] and extend the LP as follows:

$$\min \sum_{i=1}^{k} \sum_{\{r,r'\} \in T} (h^i_{r,r'} + v^i_{r,r'}) + \sum_{\{i,j\} \in I} \sum_{r \in R} (a^{i,j}_r + b^{i,j}_r) \tag{14}$$

$$a^{i,j} \geq \max\{(x^i_r - x^j_r), (x^j_r - x^i_r)\} \quad \forall_r \in R, \forall_{i,j} \in I \tag{15}$$

$$b^{i,j} \geq \max\{(y^i_r - y^j_r), (y^j_r - y^i_r)\} \quad \forall_r \in R, \forall_{i,j} \in I \tag{16}$$

For each $r \in R$, variables $a^{i,j}$ and $b^{i,j}_r$ respectively measure the horizontal and vertical displacement between $P_i(r)$ and $P_j(r)$ due to (15) and (16). In other words, we measure stability as the $L_1$ (Manhattan) distance between the centers of the same region in different layouts as specified by $I$. Any variant of the LP can be extended in this manner, by changing their objective function, e.g., if relative directions should be included, we can adapt 7.

## 5 Experimental Setup

*Linear Programs.* We categorize our method according to three properties: optimization term, method of deriving constraints, and the stability model. We implemented our LP with four different optimization terms:

*TOP*: distance between topologically adjacent regions;

*TOP**: similar to TOP, complemented by the secondary constraint of maintaining relative directions, see Equations 7 and 10 to 13;

*ORG*: distance to the origin (region's centroid in the geographic map);

*CNT*: number of lost adjacencies.

These LPs are described in full in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2022.3151227. The CNT setting was used primarily for providing an upper bound for topological accuracy; the method is too slow to be considered a feasible solution to the problem – see the end of the section for running times.

Separation constraints are deduced from the input map in one of two ways, *S* and *W*, matching the strong and weak case respectively. For stability, we use the five models as described in the previous section, as well as *N* – no stability: no optimization between layouts is added and thus all layouts are completely independent.

We specify our methods by concatenating the three properties in order, for example, TOP-S-Su indicates the LP optimized for distances of topologically adjacent regions (TOP)

3. Directed and bidirectional edges within one strongly connected component are treated equally.

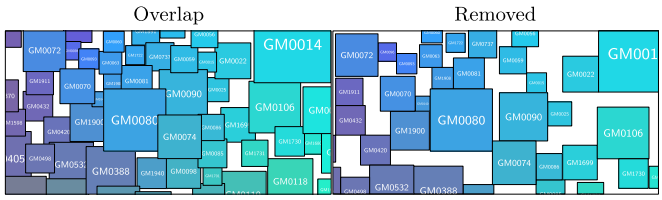



Fig. 6. Part of a force-directed layout before and after overlap removal.

with strong separation constraints (S) and with successive weight values linked (Su). We use an asterisk to indicate collections of implementations. For example, TOP-W-* refers to all TOP-W algorithms with different stability models.

*Force-Directed Method.* While being used in practice, DCs are hard to track down in the literature, especially regarding their automated construction. To our knowledge, there is no common baseline for computing a DC. Hence, we introduce a simple one that does not rely on separation constraint, such that we may also investigate their effect.

As Dorling cartograms and DCs are similar [5], and Dorling cartograms are commonly computed using a force-directed method; we implement one here, too: FRC (*Force*). For each region pair $(r, r')$ we define a disjointness force based on Chebyshev distance between their centers, which grows quadratically to push squares apart. We use the same desired distance $w_{r,r'} + gap_{r,r'}$ as in Section 3, at which this force becomes zero. We also add a force to increase cartogram quality, by pulling regions towards their original location (FRC-O) or between adjacent regions (FRC-T), analogously to the optimization terms of the LP. The specific forces and their relative weights are detailed in Appendix B, available in the online supplemental material.

Stability models are taken into account by initializing the starting positions of each region with the calculated position of the previous solution. Therefore only stability models which depend on a single previous weight value vector results are considered, specifically N, SuIt and StIt.

Forces towards the origin can clearly be computed in linear time. The same holds for forces between adjacent regions when the adjacency graph is planar. The number of computed disjointness forces is however quadratic. In order to increase performance, we overlay the map with a grid, such that every region is strictly smaller than a single grid cell. As a result, every region has a non-empty intersection with at most four grid cells. Two overlapping regions need to intersect the same cell at least once. We only calculate the disjointness forces between all pairs of regions, which intersect the same cell. This leads to a significant speed-up which let us run this algorithm on all instances.

We note two significant drawbacks to this approach. First, this force-directed solution does not always generate a valid DC as squares may overlap. In our experiments we compensated for this by considering the output of FRC-* as input for an ORG instance, i.e., an LP that includes constraints for overlap removal and at the same time minimizes the distance of each region to the position assigned by the FRC-* method, see Fig. 6. Second, separation constraints are not considered in this method and are therefore frequently violated in order to optimize the objectives. This is by design, to gain insight into the loss of quality due to imposition of separation constraints, but it also implies that the overall spatial quality (in terms of relative directions) may deteriorate, and that certain properties, such as overlap-free interpolation, are no longer applicable.

*Metrics: Cartogram Quality.* Our algorithms inherently yield zero cartographic error, and shape deformation is constant over all possible DCs and these metrics thus require no evaluation. In our experiments, we will evaluate the cartogram quality using four metrics.

*MADJ*: We measure topological accuracy as the number of lost adjacencies in all $k$ computed layouts.

*MADS*: We measure the average distance between regions in $T$, as a more fine-grained measure for topological accuracy.

*MREL*: We measure the preservation of relative directions with respect to the input map using the Relative Position Change Metric [33] which captures the preservation of the spatial mental model (orthogonal order) in a fine-grained way. Here, each rectangle defines eight zones by extending its sides to infinite lines. Between a pair of regions of the input map $(r, r')$ we consider fractions of the area of the bounding box of $r'$ that fall into the zones of the bounding box of $r$; if the bounding boxes of $r$ and $r'$ overlap, we scale values such that they sum to 1. We do the same between the corresponding squares in the cartogram layouts. The measure between two regions is half the sum over all absolute differences between fractions per zone; the value is in [0,1] but is not symmetric. Finally, we take the average over all pairs.

*MDIS*: For spatial deformation we measure distance to map origins, averaging the $L_1$ distances for each region $r$ in the DC to its origin (centroid of $r$ in the geographic map).

*Metrics: Stability.* To assess stability between the DCs, we use two quality metrics based on treemap stability metrics [33], interpreting DCs as special treemaps with added whitespace. The first metric (*SDIS*) is based on geometric distances between the layouts and measures the change in position of the squares. We use the layout distance change function as presented by Shneiderman and Wattenberg [39], measuring the average euclidean distance between all pairs of squares $(r, r')$. This strongly relates to our optimization term for quality when dealing with multiple weights (see Section 4). The second metric (*SREL*) is based on the change in relative directions between layouts; it is analogous to MREL, but compares two layouts instead.

*Postprocessing.* For comparability between the different experimental setups we included postprocessing steps. Optimality of the TOP, TOP*, CNT and the FRC-T approaches is invariant to translating all squares uniformly in each layout, and if these approaches are not using any stability implementation (*-N) they are even invariant to uniformly translating in a single layout. This can artificially worsen the MDIS and SDIS metrics, respectively. To remedy this, we move (in the first case) all layouts with a single translation vector, such that their average position is identical with the average position of regions in the input. In the

second case, we apply this translation to each layout individually.

*Datasets.* We run experiments on real-world datasets. We include both time-series datasets where we expect a gradual change and strong correlation between the different values, and weight-vectors datasets where we expect more erratic changes and less correlation. We use three maps with rather different geographic structures: the first (*World*) is a map of world countries, having mixed region (country) sizes in a rather unstructured manner. This world map was augmented with "sea regions": additional regions in the map that model connectivity between separate landmasses, but are not drawn in the final cartogram (see Appendix D, available in the online supplemental material). These sea regions have a size $s$ which is scaled relative to the largest region in a cartogram. In our LPs, these regions are allowed to take on any size in $[\frac{s}{2}, \frac{3s}{2}]$. The second (*US*) is a map of the 48 contiguous US states, having relatively high structure in sizes of its states, with large states in the middle and along the west coast and many smaller states along the east coast. The third (*NL*) is a map of the Dutch municipalities. The exact number of regions present in the map (and therefore the topology of the entire map) changes between time steps (between 388 and 483), due to municipalities merging and splitting. If such an event occurs, we consider all involved regions as unique entities, i.e., no stability connection persists. Note that a non-present region is different from a region with size 0, as such a region would still contribute to the overall topology.

For every map we collected four time series; see Appendix C, available in the online supplemental material for details. We construct a (single) weight-vectors dataset by taking the weight functions of a single time step for each of these four time series. Figs. 1, 2, and 3 illustrate excerpts of the layouts computed by TOP-W-St on these weight-vector datasets.

The values are interpreted as the areas of the squares for each region. The various datasets have different scales, and need to be projected into a reasonable square size to compute a DC. For this we scale the region sizes so that the sum of the region areas equals $\frac{A}{2}$, where $A$ is the area of the input bounding box. For a time-series dataset, we scale all time steps with the same factor (the maximum over all time steps). For a weight-vectors dataset, we scale the values for each weight function separately.

*Running Times.* All experiments were run on an Intel Xeon E5-2640 v4, 2.40GHz cpu, using Java 11, IBM ILOG CPLEX 12.8, and 8 GB (36GB for the NL map) of allocated memory to solve the (I)LP.

We observe the following running times. In general *-*-{N, SuIt, StIt} finished faster than *-*-{Co, Su, St}, smaller maps finished faster than larger maps and *-W-* finished faster than *-S-*. {TOP,ORG}-*-*It finished within seconds for all instances, except for the strong separation variant on the NL map, which finished within minutes on the time-series datasets. The non-iterative stability models TOP-*-* finished in seconds on the US map and minutes for everything else, but the NL map in the strong setting with the larger time series data sets, where completion took 1-3 hours for non-iterative stability models. ORG-*-* variants without iterative stability models finished on the USA map within seconds in all cases, on the World map within seconds (weight vector) or minutes (time series), on the NL map in the weak setting within seconds (weight vector) or minutes (time series) and on the NL map in the strong setting taking minutes (weight vector) or 1 to 2 hours (time series). The TOP*-*-* methods were only run without stability constraints on the NL map finishing in about 10 (weight-vector) or 60 hours (time series) and with iterative stability models on the WORLD map taking minutes (weight vector) or 1 to 2 hours (time series). On the USA map, this approach finished in seconds for the iterative models or minutes otherwise. The Force approach was not implemented with a focus on optimizing its running time, which are therefore only marginally informative. The FRC-ORG methods finished within seconds (US), minutes (World) or about half an hour (NL), while FRC-TOP took seconds (US), half an hour (World) and up to 4 hours (NL).

TABLE 1
Algorithm-Map Combinations

| Stab. M. | TOP | | TOP* | | ORG | | CNT | | FRC | |
|---|---|---|---|---|---|---|---|---|---|---|
| | S | W | S | W | S | W | S | W | -TOP | -ORG |
| C | ✓✓ | | US | | ✓✓ | | | | | |
| St | ✓✓ | | US | | ✓✓ | | | | | |
| StIt | ✓✓ | | US,Wo | | ✓✓ | | | | ✓ | ✓ |
| Su | ✓✓ | | US | | ✓✓ | | | | | |
| SuIt | ✓✓ | | US,Wo | | ✓✓ | | | | ✓ | ✓ |
| N | ✓✓ | | ✓✓ | | ✓✓ | | ✓✓ | | ✓ | ✓ |

*A ✓ indicates the setting was run on all three maps, otherwise the maps are indicated (Wo = World). If an algorithm was run for a map, it was run for each of its five datasets.*

In total we compare 36 variants of our LP (plus one ILP) and 6 variance of FRC with each other. All considered variants are shown in Table 1. Some TOP* variants were run only on a subset of the maps, indicated in the table. All datasets, the source code for the experiments, and a video of the results are openly available.[4]

## 6 Experimental Results

Here, we investigate our proposed LP to compute DCs, and the effect of choices within setting up the LP. Specifically, we are interested in the following questions:

1) How do results differ between strong and weak separation constraints?
2) How is the result affected by the model of stability?
3) How do optimization criteria affect the results?
4) Do the secondary direction constraints help?
5) How much quality and stability is lost due to enforcing weak separation constraints?

*Methodology.* To visualize our results, we apply the following visualization technique: a small-multiples matrix where each row matches one of our measures, each column matches an input, and each frame shows a horizontal bar chart for the various algorithms. Such a frame should allow us to compare performance between algorithms easily, drawing attention to the techniques that perform well. As most of our measures are not easily normalized in a meaningful way, we normalize to the best performing algorithm for a specific dataset: $s/b$ for a measure that is to be

4. https://github.com/loizuf/StableDemersLP

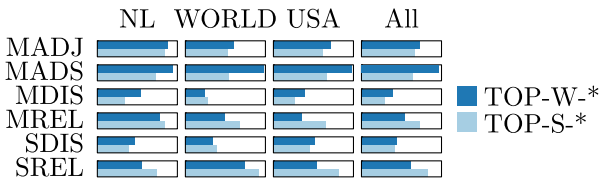



Fig. 7. Average relative scores for all TOP-W-* and TOP-S-* variants per map. Higher scores indicate better performance.

minimized and $b/s$ for a measure that is to be maximized, where $s$ and $b$ denote the score of the algorithm and the best obtained score for that dataset. Thus, the best algorithm has a fully filled bar, and all other algorithms have shorter bars accordingly – for example, an algorithm that performs twice as poorly as the best algorithm will get a bar of half length.

The full matrix can be found in Appendix E, available in the online supplemental material. With it, we established that *TOP-W-St* provides a good trade-off in terms of efficiency, cartogram quality and stability. We *recommend* using this as a default setting, and modifying the settings when the tasks require it. In the remainder, we discuss the results via aggregated excerpts to arrive at this conclusion. These excerpts aggregate the datasets according to the underlying geography, or to the type of weight functions used. Moreover, a selection of the algorithms is usually shown, possibly also combining different settings.

As the force-directed layouts violate the separation constraints, these are excluded from normalization in the visualization, unless explicitly indicated otherwise. Note that this is only relevant in answering our last question.

*1) Strong and Weak Separation.* To compare the weak and strong setting, that is, to investigate the influence of the secondary separation constraints, we compare TOP-W-* to TOP-S-*. Fig. 7 shows their performance for each map.

The TOP-S-* variants perform better on relative direction measures, both within a DC (MREL) and between DCs (SREL). This is to be expected, since the secondary separation constraints aim at prescribing relative directions more strongly. However, we see that it performs worse in all other measures: specifically for MADS, there is a considerable drop in performance, implying that gaps between regions that should be adjacent are significantly larger.

The only exception is that, for World datasets, TOP-S-* performs slightly better on keeping elements close to their geographic location (MDIS) and to other layouts (SDIS). This is likely caused due to the rather weak connection between continents by the sea regions. As such, the strong setting helps to better preserve the relations between these parts and position them appropriately.

A trade-off thus exists between preserving directions (TOP-S-*) and obtaining a compact DC (TOP-W-*). As the strong setting has considerably higher computation time, we select TOP-W-* as the recommended technique and compare it further.

*2) Stability Model.* We consider the influence of different stability models. We aggregate our datasets into time-varying and weight-vector datasets, and visualize the performance per stability model of TOP-W-* in Fig. 8.

We first observe that TOP-W-C and TOP-W-St perform almost identically. Considering the increased computational cost, TOP-W-St is the better choice. Comparing TOP-W-St to

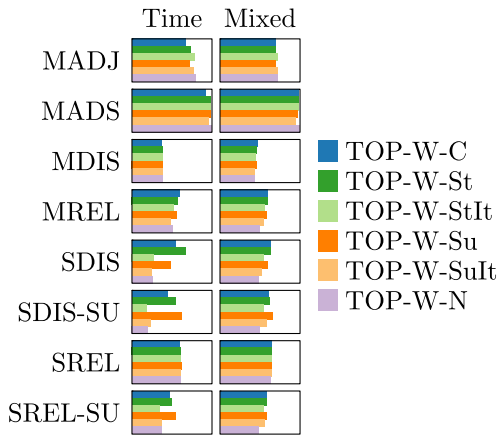



Fig. 8. Average relative scores for stability models of TOP-W-* variants per weight type. Higher scores indicate better performance.

TOP-W-Su, we see similar performance, though TOP-W-St is slightly better: interestingly, the largest difference is in SDIS for the time-series data. This is likely attributable to time-varying data increasing or decreasing over time.

When we compare iterative models to their non-iterative counterparts, we see that iterative models perform worse in stability measures – though minor improvement in cartogram quality can be observed for some measures such as MADJ. SDIS shows the largest effect: in the iterative model, regions move unnecessarily between different layouts. This effect is particularly strong for time-varying data.

However, these measures consider the stability between all pairs of layouts: this is useful in settings where arbitrary layouts need to be compared, but less relevant when, e.g., creating a video that shows the layouts in a specific sequence. Hence, we also measure these only on successive layouts (SDIS-SU and SREL-SU); for time series, this is the temporal order, for mixed weights this is an arbitrary order. The main effect is that TOP-W-Su gains in relative performance, but the iterative models remain weak even in this setting. The difference is not very large though, and thus we generally recommend the St model for stability.

If the algorithm does not consider stability at all (TOP-W-N), then we see a considerable drop in performance for stability, as to be expected. However, we see comparatively little gain in cartogram quality. Mostly, MADJ is slightly higher. Interestingly, MREL is even lower. This is likely explainable through stability: by requiring stability between layouts that can have extremely small or large weights, squares cannot be placed in extreme positions as this would incur a large cost of stability. When no stability is considered, they can be placed in extreme positions to the detriment of MREL.

*3) Optimization Criteria.* In Fig. 9 we compare TOP-W-*, ORG-W-* and CNT-W-* for each map. Generally, we see the same pattern between the maps, but the magnitude of the performance differences between these methods is influenced by the map; care needs to be taken when generalizing the results in terms of magnitude beyond these maps.

Comparing TOP and ORG variants, we see that TOP yields better results in adjacency-based measures (both MADJ and MADS), but worse in the other measures. This suggests that regions need to move from their original locations more to preserve neighborhoods well. This is in accordance with the

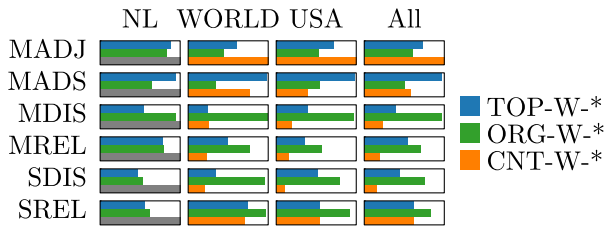



Fig. 9. Average relative scores for TOP-W-*, ORG-W-* and CNT-W-* variants. Higher scores indicate better performance. Computation for CNT-W-* on NL did not finish.

general observations in related problems [21], [22] that optimizing displacement from the original location performs fairly well. However, in the literature, cartograms are typically evaluated based on how well they preserve neighborhoods, rather than distances. As such, we prefer TOP variants over ORG nonetheless in for DCs; other use cases may rather use ORG variants. The main trade-off we see here is better neighborhood preservation versus reduced stability. We prefer neighborhood preservation here, as overlap-free animations can be achieved (Lemma 1), and needless displacement is prevented by our methods in any case. Also observe that the relative importance of stability could be increased in the objective function; investigating such effects are beyond the scope of this work.

Compared to optimizing the standard quality measure of counting the number of maintained adjacencies (MADJ), we see that CNT variants perform best as they explicitly optimize for it. However, the computational complexity makes this ILP impractical: as indicated earlier, it does not compute on the NL map and is slow on the others. Moreover, we see that, though it maintains many adjacencies, it is overall worse in keeping adjacent regions close together (MADS) and other measures. Nonetheless, it gives us some insight into how well TOP and ORG variants maintain adjacencies. For TOP, we see that it maintains about 75% of the optimum amount of adjacencies, but its total adjacency distance score improves by 67% with regards to CNT-W-*.

*4) Secondary Direction Constraint.* In Fig. 10 we compare TOP to TOP*, that is, we investigate the effect of adding a secondary optimization term that considers the relative direction between all pairs of regions. Because the LP gains a quadratic number of constraints, the NL map is excluded from comparison as it is no longer efficiently computable.

As we may expect, the relative position scores (MREL and SREL) improve slightly for TOP*, at the expense of slightly worse scores for the other metrics. Given the computational cost involved, we recommend using TOP rather than the more nuanced TOP*. Note that TOP* inherently has the ability to place disconnected parts (islands or different continents) in the correct relative position. One reason why little difference is achieved is due to sea regions being added to ensure a

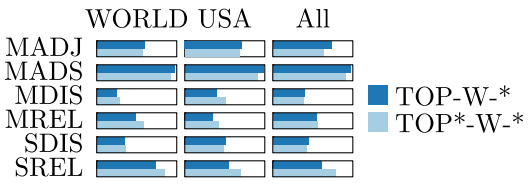



Fig. 10. Average relative scores for TOP-W-* and TOP*-W-* variants per map. Higher scores indicate better performance.

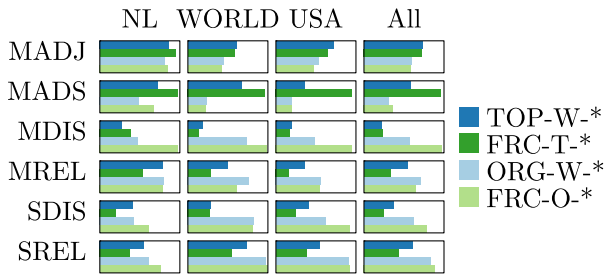



Fig. 11. Average relative scores comparing our solutions leveraging separation constraints (TOP-W-* and ORG-W-*) to force-based solutions which do not adhere to such constraints (FRC-T-* and FRC-O-*). Higher scores indicate better performance.

connected topology for the input. If no such action is taken, TOP* may become preferable.

*5) No Separation Constraints.* Finally, we consider the cost induced by our separation constraints. Though it allows for overlap-free animations using simple linear interpolation between cartograms (Lemma 1), it may affect cartogram quality. To investigate the degree to which this happens, we compare TOP-W-* and ORG-W-* variants to FRC-T-* and FRC-W-* respectively. That is, we use a heuristic approach[5] to optimize for the same objective function without the separation constraints, and compare the resulting cartogram quality and stability. The results are shown in Fig. 11.

Comparing TOP-W-* to FRC-T-*, we see that they perform very similarly in MADJ and MDIS. In other words, the separation constraints have little influence on how many adjacencies are present in the cartogram and on how far regions are to move from their map location. However, we do see that adding such constraints tends to increase the distances between adjacent regions if these are not maintained. We also see that our separation constraints improve the relative positions between regions. Thus, to improve distances between neighboring regions, one must violate the separation constraints. In terms of stability (SDIS and SREL), we see that our LP-based methods perform better than the force-based methods. However, this may be attributable to our FRC methods using only iterative or no stability models: FRC methods do not incorporate stability forces. As such, FRC methods are primarily focused on cartogram quality.

We see similar patterns comparing ORG-W-* to FRC-O-*. Notably though, there is less advantage for MREL for ORG-W-* with respect to FRC-O-*, and, for SDIS and SREL, FRC-O-* even performs better than ORG-W-*; but consistent with earlier considerations, displacement-based algorithms such as these perform generally well in terms of stability.

## 7 Discussion

Our evaluation focused on cartogram quality and stability, both measure using multiple criteria, and demonstrated that our LP-based methods are an effective way of solving the problem. Here, we briefly reflect on how our results may be used in an eventual system.

Consider a system that shows a cartogram based on interactively selectable weight functions. By Lemma 1, the

5. It would be straightforward to define a Mixed Integer Program that computes the optimal result; however, this did not result in reasonably solvable programs in our experience due to our map sizes.

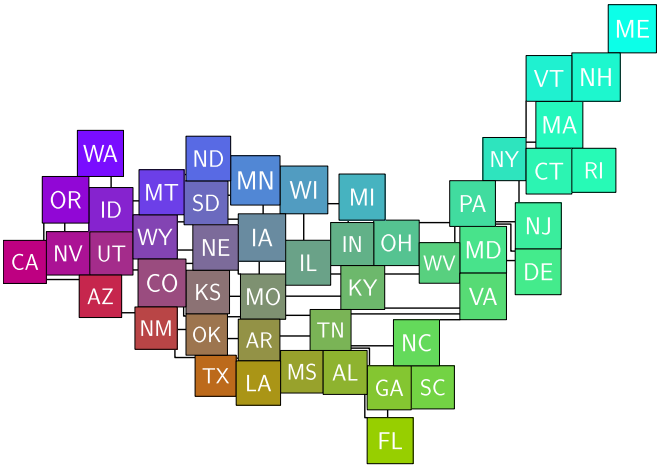



Fig. 12. Example with manually drawn leader lines.

linear interpolation is free of overlap and can thus be easily animated when switching between weights to make it easier to track the items. Based on our findings, we recommend TOP-W-St as the algorithm to be used for such a setting.

However, regardless of the method, adjacencies may be lost: not all planar graphs can be represented using touching squares. A real-world example is Luxembourg having three pairwise neighbors; the input graph $G$ is a $K_4$, which has no touching-squares representation. However, in an interactive system we might want to visualize these lost adjacencies to better show the user which regions are adjacent in reality.

One way to show this would be to use *leaders* between regions that lost adjacencies: non-crossing orthogonal polylines connecting the two squares through the gaps in the DC . We want these leaders to be short and have few bends, which we can guarantee under mild assumptions: 1) leaders can coincide with square boundaries; 2) regions to be connected are *realizable*, i.e., a valid DC (with possibly different weights) exists for each pair of regions such that they are adjacent. Even when these assumptions are not met, we expect the leaders to have these properties in practice.

Let us first consider leader length. Let $L_1^B(r_1, r_2)$ denote the $L_1$ distance between squares of regions $r_1$ and $r_2$ in DC $B$. The lemma below states that a leader of minimal length can always be drawn between two adjacent regions in DCs that satisfy separation constraints, and thus by extension in DCs computed by our LP. Note, that since we minimize the $L_1$ distance between lost adjacencies in our LP (TOP-variants), our optimization criteria corresponds directly to the total length of all leaders: leaders are as short as possible.

**Lemma 2.** *Consider DCs with separation constraints $H, V$, and two regions $\{r_1, r_2\} \in T$. Let $(r_1, r_2)$ be a pair in $H$ or $V$ that is adjacent in the input map. Then, in any DC $B$, there is a monotone geodesic leader $\ell$ between $r_1$ and $r_2$ with length $L_1^B(r_1, r_2)$.*

The proof of this lemma (found in Appendix F, available in the online supplemental material ) readily gives us an algorithm to compute these minimal leaders. Aside from minimal length, we also want leaders to have few bends (low complexity). With strong separation constraints, we can indeed guarantee this. This is captured by the lemma below; its proof can be found Appendix F, available in the online supplemental material.

**Lemma 3.** *Let $\{r_1, r_2\} \in T$ and assume a DC $A$ exists with $r_1$ and $r_2$ adjacent, from which $H$ and $V$ are derived in the strong setting. Then, for any DC $B$ satisfying $H$ and $V$, a leader $\ell$ exists between $r_1$ and $r_2$ with at most two bends.*

The proof of this lemma (found in Appendix F, available in the online supplemental material ) also gives us an algorithm to compute these leaders. For effective use in a system, we may however want to slightly modify these leaders to maintain a small visible gap between the leaders and the edges of other squares, similar to the gap maintained for non-adjacent regions, for increased clarity.

Of course, all leaders can be drawn (see Fig. 12), but we expect that this is particularly effective in an interactive manner. For example, leaders are drawn or highlighted for a selected region, to allow seeing the topological adjacencies immediately in the cartogram. Of course, such interactions can be complemented by juxtaposing it with a linked view that shows the geographic map in which the same regions can be highlighted, which was shown to be effective even for novice users at least for contiguous area cartograms [40].

## 8 Conclusion and Future Work

We described a linear program to compute stable Demers cartograms for dynamic data based on separation constraints and minimizing distance between adjacent regions. It allows overlap-free transitions between weight functions, and the connecting of lost adjacencies with short and low-complexity leaders. Experiments show it offers a good trade-off between topological error and other criteria. The LP outperforms basic force-directed layouts, though there is no unique variant that does so, suggesting an interplay between separation constraints, optimization and quality metrics.

In future work we may consider stability in other cartogram styles, and perform human-centered comparisons in addition to computational ones, with methods implemented in interactive systems; such systems can, e.g., emphasize adjacent regions by drawing leaders (at all or more clearly) or link regions back to the geographic map. We focused on Demers cartograms, but there are many different styles of cartograms. Future work may also investigate stable variants of such other cartogram styles.

## Acknowledgments

This research was initiated at NII Shonan Meeting 127 “Reimagining the Mental Map and Drawing Stability”. The authors would like to thank Markus Chimani for fruitful discussions on the topic of this paper.

## References

[1] T. Munzner, *Visualization Analysis and Design*. Wellesley, MA, USA/Boca Raton, FL, USA: AK Peters/CRC Press, 2014.
[2] S. Nusrat and S. Kobourov, “The state of the art in cartograms,” *Comput. Graph. Forum*, vol. 35, no. 3, pp. 619–642, 2016.
[3] D. Dorling, “Stretching space and splicing time: From cartographic animation to interactive visualization,” *Cartogr. Geographic Informat. Syst.*, vol. 19, no. 4, pp. 215–227, 1992.
[4] K. Misue, P. Eades, W. Lai, and K. Sugiyama, “Layout adjustment and the mental map,” *J. Vis. Lang. Comput.*, vol. 6, no. 2, pp. 183–210, 1995.
[5] I. Bortins, S. Demers, and K. Clarke, “Cartogram types,” 2002. [Online]. Available: http://www.ncgia.ucsb.edu/projects/Cartogram_Central/types.html

[6] D. Dorling, *Area cartograms: Their use and creation*. Norwich, U.K.: Univ. of East Anglia, Environmental Publications, 1996.
[7] M. van Kreveld and B. Speckmann, "On rectangular cartograms," *Comput. Geometry*, vol. 37, no. 3, pp. 175–187, 2007.
[8] E. Raisz, "The rectangular statistical cartogram," *Geographical Rev.*, vol. 24, no. 2, pp. 292–296, 1934.
[9] W. R. Tobler, "A continuous transformation useful for districting," *Ann. New York Acad. Sci.*, vol. 219, pp. 215–220, 1973.
[10] D. H. House and C. J. Kocmoud, "Continuous cartogram construction," in *Proc. IEEE Conf. Visualization*, 1998, pp. 197–204.
[11] M. Gastner and M. Newman, "Diffusion-based method for producing density-equalizing maps," *Proc. Nat. Acad. Sci. USA*, vol. 101, pp. 7499–7504, 2004.
[12] M. T. Gastner, V. Seguy, and P. More, "Fast flow-based algorithm for creating density-equalizing map projections," *Proc. Nat. Acad. Sci. USA*, vol. 115, no. 10, pp. 2156–2164, 2018.
[13] S. Sun, "Applying forces to generate cartograms: A fast and flexible transformation framework," *Cartogr. Geographic Informat. Sci.*, vol. 47, no. 5, pp. 381–399, 2020.
[14] M. J. Alam, T. Biedl, S. Felsner, M. Kaufmann, S. G. Kobourov, and T. Ueckerdt, "Computing cartograms with optimal complexity," *Discrete Comput. Geometry*, vol. 50, no. 3, pp. 784–810, 2013.
[15] K. Buchin, B. Speckmann, and S. Verdonschot, "Evolution strategies for optimizing rectangular cartograms," in *Proc. Int. Conf. Geographic Informat. Sci.*, 2012, pp. 29–42.
[16] D. Eppstein, E. Mumford, B. Speckmann, and K. Verbeek, "Area-universal rectangular layouts," in *Proc. Symp. Comput. Geometry*, 2009, pp. 267–276.
[17] R. G. Cano, K. Buchin, T. Castermans, A. Pieterse, W. Sonke, and B. Speckmann, "Mosaic drawings and cartograms," *Comput. Graph. Forum*, vol. 34, no. 3, pp. 361–370, 2015.
[18] H. Breu and D. G. Kirkpatrick, "Unit disk graph recognition is NP-hard," *Comput. Geometry*, vol. 9, no. 1/2, pp. 3–24, 1998.
[19] E. Di Giacomo *et al.*, "Low ply graph drawing," in *Proc. 6th Int. Conf. Informat., Intell., Syst. Appl.*, 2015, pp. 1–6.
[20] B. Klemz, M. Nöllenburg, and R. Prutkin, "Recognizing weighted disk contact graphs," in *Proc. 23rd Int. Symp. Graph Drawing Netw. Visualization*, 2015, pp. 433–446.
[21] D. Eppstein, M. van Kreveld, B. Speckmann, and F. Staals, "Improved grid map layout by point set matching," *Int. J. Comput. Geometry Appl.*, vol. 25, no. 2, pp. 101–122, 2015.
[22] W. Meulemans, J. Dykes, A. Slingsby, C. Turkay, and J. Wood, "Small multiples with gaps," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 1, pp. 381–390, Jan. 2017.
[23] W. Meulemans, M. Sondag, and B. Speckmann, "A simple pipeline for coherent grid maps," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 1236–1246, Feb. 2021.
[24] R. Inoue, "A new construction method for circle cartograms," *Cartogr. Geographic Informat. Sci.*, vol. 38, no. 2, pp. 146–152, 2011.
[25] R. Inoue, "Generalized approach to construction of simple-shaped non-contiguous area cartograms," in *Proc. 25th Int. Cartographic Conf.*, 2011, pp. 3–8.
[26] W. Meulemans, "Efficient optimal overlap removal: Algorithms and experiments," *Comput. Graph. Forum*, vol. 38, no. 3, pp. 713–723, 2019.
[27] M. J. Alam, S. G. Kobourov, and S. Veeramoni, "Quantitative measures for cartogram generation techniques," *Comput. Graph. Forum*, vol. 34, no. 3, pp. 351–360, 2015.
[28] D. A. Keim, S. C. North, and C. Panse, "CartoDraw: A fast algorithm for generating contiguous cartograms," *IEEE Trans. Vis. Comput. Graphics*, vol. 10, no. 1, pp. 95–110, Jan./Feb. 2004.
[29] T. Johnson, C. Acedo, S. Kobourov, and S. Nusrat, "Analyzing the evolution of the Internet," in *Proc. Eurograph. Conf. Visualization*, 2015, pp. 43–47.
[30] W. Meulemans, B. Speckmann, K. Verbeek, and J. Wulms, "A framework for algorithm stability and its application to kinetic Euclidean MSTs," in *Proc. 13th Latin Amer. Symp. Theor. Inform.*, 2018, pp. 805–819.
[31] I. van der Hoog, M. J. van Kreveld, W. Meulemans, K. Verbeek, and J. Wulms, "Topological stability of kinetic k-centers," in *Proc. 13th Int. Conf. Algorithmcs Comput.*, 2019, pp. 43–55.
[32] J. Wulms, J. Buchmüller, W. Meulemans, K. Verbeek, and B. Speckmann, "Stable visual summaries for trajectory collections," in *Proc. 14th IEEE Pacific Vis. Symp.*, 2021, pp. 61–70.
[33] M. Sondag, B. Speckmann, and K. Verbeek, "Stable treemaps via local moves," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 729–738, Jan. 2018.
[34] E. F. Vernier, M. Sondag, J. Comba, B. Speckmann, A. Telea, and K. Verbeek, "Quantitative comparison of time-dependent treemaps," *Comput. Graph. Forum*, vol. 39, no. 3, pp. 393–404, 2020.
[35] K. Buchin, V. Kusters, B. Speckmann, F. Staals, and B. Vasilescu, "A splitting line model for directional relations," in *Proc. 19th ACM SIGSPATIAL Int. Conf. Adv. Geographic Informat. Syst.*, 2011, pp. 142–151.
[36] C. Bowen, S. Durocher, M. Löffler, A. Rounds, A. Schulz, and C. D. Tóth, "Realization of simply connected polygonal linkages and recognition of unit disk contact trees," in *Proc. Graph Drawing Netw. Visualization*, 2015, pp. 447–459.
[37] P. Hliněný, "Contact graphs of line segments are NP-complete," *Discrete Math.*, vol. 235, no. 1–3, pp. 95–106, 2001.
[38] R. E. Tarjan, "Depth-first search and linear graph algorithms," *SIAM J. Comput.*, vol. 1, no. 2, pp. 146–160, 1972.
[39] B. Shneiderman and M. Wattenberg, "Ordered treemap layouts," in *Proc. IEEE Symp. Informat. Visualization*, 2001, pp. 73–78.
[40] I. K. Duncan, S. Tingsheng, S. T. Perrault, and M. T. Gastner, "Task-based effectiveness of interactive contiguous area cartograms," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 3, pp. 2136–2152, Mar. 2021.
[41] K. Buchin, R. G. Cano, P. J. de Rezende, C. C. de Souza, and B. Speckmann, "Sea regions for rectangular cartograms," in *Proc. Abstr. 30th Eur. Workshop Comput. Geometry*, 2014, pp. 47:1–47:4.

**Soeren Nickel** received the master's degree in 2019 from TU Wien, where he is currently working toward the PhD degree. His research interests include map schematization and complexity of and algorithms for geometric problems.

**Max Sondag** received the PhD degree from TU Eindhoven in 2020. He is currently a postdoc with Swansea University. His research interests include geometric algorithms in both theory and practice, and (stable) visualizations.

**Wouter Meulemans** is currently an assistant professor of algorithms, geometry and applications cluster with TU Eindhoven. His research interests include geometric algorithms, graph drawing, (geo)visualization, and automated cartography.

**Stephen Kobourov** received the BS degree in mathematics and computer science from Dartmouth College, and the MS and PhD degrees from Johns Hopkins University. He is currently a Professor with the Department of Computer Science, University of Arizona. His research interests include information visualisation, graph theory, and geometric algorithms.

**Jaakko Peltonen** received the DSc degree from the Helsinki University of Technology in 2004. He is currently a professor of statistics and data analysis with the Faculty of Information Technology and Communication Sciences, Tampere University. His research interests include statistical machine learning, dimensionality reduction, visual data analysis, exploratory search, and modeling text data.

**Martin Nöllenburg** received the PhD and habilitation degrees from the Karlsruhe Institute of Technology, Germany, in 2009 and 2015, respectively. He is currently a professor with Algorithms and Complexity Group, TU Wien, Vienna, Austria. His research interests include graph drawing algorithms, computational geometry, and information visualization.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.