Balanced Centroidal Power Diagrams for Redistricting

Vincent Cohen-Addad
CNRS, Sorbonne Université, Paris

Philip N. Klein*
Brown University

Neal E. Young†
University of California, Riverside

ABSTRACT

We consider the problem of political *redistricting*: given the locations of people in a geographical area (e.g. a US state), the goal is to decompose the area into subareas, called *districts*, so that the populations of the districts are as close as possible and the districts are "compact" and "contiguous," to use the terms referred to in most US state constitutions and/or US Supreme Court rulings.

We study a method that outputs a solution in which each district is the intersection of a convex polygon with the geographical area. The average number of sides per polygon is less than six. The polygons tend to be quite compact. Every two districts differ in population by at most one (so we call the solution *balanced*).

In fact, the solution is a *centroidal power diagram*: each polygon has an associated *center* in $\mathbb{R}^3$ such that

- the projection of the center onto the plane $z = 0$ is the centroid of the locations of people assigned to the polygon, and
- for each person assigned to that polygon, the polygon's center is closest among all centers. The polygons are convex because they are the intersections of 3D Voronoi cells with the plane.

The solution is, in a well-defined sense, a locally optimal solution to the problem of choosing centers in the plane and choosing an assignment of people to those 2-d centers so as to minimize the sum of squared distances subject to the assignment being balanced.

A practical problem with this approach is that, in real-world redistricting, exact locations of people are unknown. Instead, the input consists of polygons (*census blocks*) and associated populations. A real redistricting must not split census blocks. We therefore propose a *second phase* that perturbs the solution slightly so it does not split census blocks. In our experiments, the second phase achieves this while preserving perfect population balance. The district polygons are no longer convex at the fine scale because their boundaries must follow the boundaries of census blocks, but at a coarse scale they preserve the shape of the original polygons.

*Research supported by National Science Foundation Grants CCF-1409520 and CCF-1841954.
†Research supported by National Science Foundation Grant IIS-1619463.

SIGSPATIAL '18, November 6–9, 2018, Seattle, WA, USA

ACM ISBN 978-1-4503-5889-7/18/11…$15.00
https://doi.org/10.1145/3274895.3274979

CCS CONCEPTS

• **Theory of computation → Facility location and clustering**; *Discrete optimization*; *Algorithm design techniques*.

KEYWORDS

redistricting, computational geometry, optimization, graph algorithm

ACM Reference Format:
Vincent Cohen-Addad, Philip N. Klein, and Neal E. Young. 2018. Balanced Centroidal Power Diagrams for Redistricting. In *26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '18), November 6–9, 2018, Seattle, WA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3274895.3274979

1 INTRODUCTION

In the context of elections, *redistricting* refers to decomposing a geographical area into subareas called *districts*. The districts are supposed to satisfy three properties.

First, in order to honor the principle of equal representation, the districts are supposed to have equal population to the extent possible. Although the Supreme Court has declined to name a specific percentage limit on how much populations of districts can differ, "a 2002 Pennsylvania redistricting plan was struck down because one district had 19 more people … than another." [16, p. 499]

Second, districts are supposed to be *contiguous* to the extent that is possible. "…Forty-nine [out of fifty] states require at least one chamber's state legislative districts to be contiguous …the vast majority of congressional districts—perhaps every one in the 2010 cycle—will be drawn to be contiguous" [24]. Contiguous can reasonably be interpreted to mean *connected*.

Third, is *compactness*. "Thirty-seven states require their legislative districts to be reasonably compact; eighteen states require congressional districts to be compact as well. Few states define precisely what 'compactness' means, but a district in which people generally live near each other is usually more compact than one in which they do not." [24]

There are other criteria considered by the states. Some states require that district boundaries account in some way for existing political boundaries such as county or city lines, although there is flexibility in this rule.

In this paper, we focus on equal population, contiguity, and compactness. Of these, compactness is the one that is not easy to formalize. Some measures of compactness are based on boundaries; a district is preferred if its boundaries are simpler rather than contorted. Some measures are based on dispersion, "the degree to which the district spreads from a central core" [24]. Idaho, for example, directs its redistricting commision to "avoid drawing districts that are oddly shaped." Other states loosely address the meaning of compactness: "Arizona

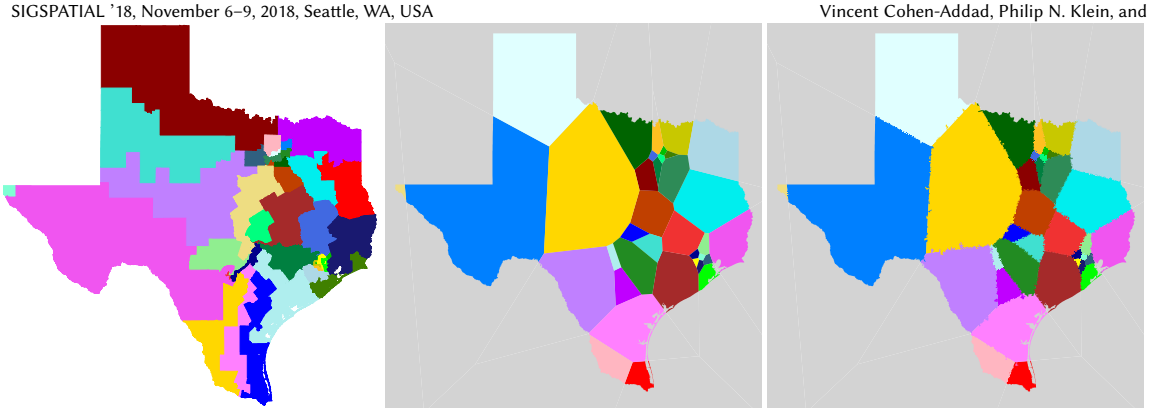

Figure 1: *(left)* The current Texas districting plan for the 114th Congress [25], designed to respect some county boundaries. *(middle)* A solution to the idealized redistricting problem for Texas. *(right)* A solution that respects census blocks. The middle and right figures are perfectly population-balanced.

and Colorado focus on contorted boundaries; California, Michigan, and Montana focus on dispersion; and Iowa embraces both" [24].

Fryer and Holden [16], two economists, state three properties that they argue every index of compactness should satisfy, and formulated a measure of compactness they call *relative proximity index* (RPI). They showed that any compactness index satisfying their three properties ranks districting plans identically to the RPI. The RPI is based on a quantity that for this paper we will call the *dispersion* of a districting plan: it is the sum, over all districts, of the sum, over all pairs of residents in a district, of the squared distance between the two residents. The RPI of a plan is the ratio of the plan's dispersion to the minimum dispersion achievable. Achieving the minimum dispersion is NP-hard. Fryer and Holden use a local-search heuristic to upper bound the optimum, so as to estimate the RPI of existing districting plans.

1.1 The idealized redistricting problem: assuming known locations of residents

Here is one natural way to formulate redistricting as a computational problem: the input specifies the desired number k of districts and a set of points (the locations of the residents). The output is a decomposition of the state into polygons. Assuming this formulation, a minimum-dispersion set of districts has the following desirable properties:

(P1) Each district is the intersection of the state with a convex polygon.
(P2) The average number of sides per polygon is less than six.
(P3) The populations of the districts differ by at most one (or zero if the total population is divisible by k).

Districts that are convex polygons with few sides on average are arguably not "oddly shaped" and have boundaries that are not "contorted" [24]. Section 2 discusses a practical method ("Phase One") that achieves properties P1–P3. The middle of Figure 1 shows the output of Phase One for Texas. For contrast, the left of the figure shows the actual current districts (for the 114th Congress).

It is worth emphasizing Property P3: populations of districts differ by at most one (zero if the total state population is divisible by k). We say a districting plan with this property is *(perfectly) balanced.*

1.2 A more practical redistricting problem: respecting census blocks

The problem formulation in Section 1.1 is idealized. In reality, the exact locations of residents are unknown. The input consists of polygons (*census blocks*) and associated populations. Because the locations of the residents within each census block are unknown, a real districting plan *must not split census blocks.* Unfortunately, ideal districts that satisfy properties P1–P3 will have polygonal boundaries that cut right through census blocks. To what extent can we preserve properties P1–P3 while also respecting (that is, not splitting) census blocks?

Adding this requirement to the three legal requirements—population balance, contiguity, and compactness—makes the problem considerably more difficult. As far as we know, no previously published redistricting method both respects census blocks and achieves contiguity and perfect population balance. Hess et al. [21] report population differences ranging between 6.8% and 9.5%.[1] Garfinkel and Nemhauser [17] tried to respect larger geographical units (counties) and had a tradeoff between compactness and population deviation; for one of their more difficult instances (Washington state), they achieved 3.8% population deviation with a solution that was slightly less compact than the existing redistricting plan. Helbig et al. [20] report a 2.84% difference between largest and smallest. Mehrotra et al. [28] report that their method generated a plan whose maximum population deviation from exact balance was 1.86% but that when they allowed greater deviation, they obtained a more compact solution (with a better objective value) with a deviation of 3.73% after postprocessing. Spann et al. [31] limit the population deviation to at most 2%.

[1] This is the percentage by which the greatest district population exceeds the least in a given plane.

Some of these methods respect (or try to respect) larger geographical units than census blocks (census tracts or counties), which further impedes achieving population balance.

Our main contribution is a method that in all our experiments succeeded in respecting census blocks while achieving contiguity and perfect population balance. The method consists of two phases:

- Phase One uses local search to find an idealized districting plan satisfying properties P1–P3. For this phase, the population of a census block is considered to be located at the centroid of the block, but (if the center is on a district boundary) this population may be *split and assigned to different districts*.
- Phase Two assigns some census blocks, those that are in a sense on the boundaries of the polygons, to nearby districts, then reassigns the entire population of each such block to its district. The assignment is guaranteed to respect census blocks and preserve connectivity. It is chosen to minimize the maximum difference in population between the resulting districts.

In each of our experiments, Phase Two achieved perfect population balance.

Furthermore, since Phase Two only affects the boundary census blocks, the resulting districts largely retain the virtues of the Phase-One solution. The Phase-Two districts are no longer convex polygons with few sides—the polygon boundaries become more complicated—but that complexity is in a sense due to the complexity of the census-block boundaries. At a large scale, the district boundaries still resemble those of the Phase-One polygons. Moreover, since only boundary census blocks are affected, Phase Two only slightly increases the value of the dispersion measure. The right of Figure 1 shows the output of Phase Two for Texas.

2 PHASE ONE: BALANCED CENTROIDAL POWER DIAGRAMS

Proposals to use optimization for redistricting date as far back as 1965 [17, 21]. See [2, 30] for additional references. In this section we focus specifically on *balanced centroidal power diagrams*, which is what the first phase of our algorithm computes. We start with definitions and relevant history.

Fix an input (P, k), where P, the *population*, is a set of m *residents* (points in a Euclidean space), and k is the desired number of districts. Given (P, k), Phase One outputs a pair (C, f), where C is a sequence of k *centers* (points in the Euclidean space) and $f : P \to C$ is an assignment of residents to centers. Let $d(y, x)$ denote the distance from resident $y \in P$ to a possible center x.

For given centers C and a weight $w_x \in \mathbb{R}$ for each center $x \in C$, the *power diagram* of (C, w), denoted $\mathcal{P}(C, w)$, is defined as follows. For any center $x \in C$, the *weighted squared distance* from any point y to x is $d^2(y, x) - w_x$.

The *power cell* C_x associated with $x \in C$ consists of all points whose weighted squared distance to x is no more than the weighted squared distance to any other center in C. The power diagram $\mathcal{P}(C, w)$ is the collection $\{C_x : x \in C\}$ of these power cells.[2] An assignment $f : P \to C$ is *consistent* with $\mathcal{P}(C, w)$ if every resident assigned to center x belongs to the corresponding cell C_x. (Residents in the interior of C_x are necessarily assigned to x.) $\mathcal{P}(C, w, f)$ denotes the power diagram $\mathcal{P}(C, w)$ augmented with a consistent assignment f.

Power diagrams are well-studied [4]. If the Euclidean space is $\mathbb{R}^2$, it is known that each power cell C_x is necessarily a (possibly unbounded) convex polygon.

If each weight w_x is zero, the power diagram is the well-known *Voronoi diagram*, and denoted $\mathcal{V}(C)$. Likewise $\mathcal{V}(C, f)$ denotes the Voronoi diagram extended with a consistent assignment f (which simply assigns each resident to a nearest center).

A *centroidal power diagram* is an augmented power diagram $\mathcal{P}(C, w, f)$ such that the assignment f is *centroidal*: each center $x \in C$ is the centroid (center of mass) of its assigned residents, $\{y \in P : x = f(y)\}$. Compared to general power diagrams, centroidal power diagrams are often preferred because their cells tend to be more compact.

Centroidal *Voronoi* diagrams in particular have many applications [14]. A canonical application from graphics is downsampling a given image, by partitioning the image into cells, then selecting a single pixel from each cell to represent the cell. *Lloyd's method* is a standard way to compute a centroidal Voronoi diagram $\mathcal{V}(C, f)$, given (P, k) [14, § 5.2]. Starting with a sequence C of k randomly chosen centers, the method repeats the following two steps just until Step (2) does not change C:

(1) Given C, let f be any assignment assigning each resident to a nearest center in C.
(2) Move each center $x \in C$ to the centroid of the residents that f assigns to x.

The *cost* is $\sum_{y \in P} d^2(y, f(y))$. Step 1 chooses an f of minimum cost, given C. Step 2 moves the centers to minimize the cost, given the assignment f. Each iteration except the last produces a lower-cost assignment, so the algorithm must terminate. At termination, $\mathcal{V}(C, f)$ is, as desired, a centroidal Voronoi diagram, because Step 1 computes f that is consistent with $\mathcal{V}(C)$, and (in the last iteration) Step 2 does not change C.

At termination, (C, f) is a *local minimum* with respect to the cost in the following sense: by just moving the centers in C, or just changing f to any other assignment, it is not possible to reduce the cost. [3]

Miller [29] and Svec et al. [32] and Kleiner et al. [23] explore the use of centroidal Voronoi diagrams specifically for *redistricting*. The resulting districts (cells) are convex polygons, and tend to be compact, but their populations can be far from balanced.

[2] Subtracting the same number from all weights does not change the power cells. By subtracting an appropriate number, one can ensure that all weights are nonpositive. Interpret each weight as the negative of the square of the z-coordinate of the center; then the weighted squared distance from a point y to the center becomes the squared distance in 3-d from the point y (now lying in the plane $z = 0$) to the center's 3-d location. According to this interpretation, each power cell is the intersection with the plane $z = 0$ of the 3-d Voronoi cell of the center.

[3] Indeed, this condition is necessary and sufficient for $\mathcal{V}(C, f)$ to be centroidal.

A *balanced* power diagram is an augmented power diagram $\mathcal{P}(C, w, f)$ (not necessarily centroidal) such that the assignment f is perfectly balanced as defined in the introduction—for any two cells of $\mathcal{P}(C, w, f)$, the sizes of their populations are the same, or differ by at most 1 if the total population is not divisible by k.

Aurenhammer et al. [5, Theorem 1] give an algorithm that, given (P, k) *and* C, computes a weight vector w and an assignment f such that $\mathcal{P}(C, w, f)$ is a balanced power diagram, and f has minimum cost among all balanced assignments of P to C.[4] Their algorithm assumes a Euclidean metric and computes a specific f. As observed by Spann et al. [31], a slightly stronger result is possible: for any metric, given any (P, k, C) one can compute a weight-vector w such that $\mathcal{P}(C, w, f)$ is a balanced power diagram for *every* minimum-cost balanced assignment f.

Phase One: balanced centroidal power diagrams. Given (P, k), Phase One of our algorithm computes a *balanced centroidal power diagram*, an augmented power diagram $\mathcal{P}(C, w, f)$ such that f is both balanced *and* centroidal. It does so using the following capacitated variant of Lloyd's method:

Starting with a sequence C of k randomly chosen centers, repeat the following steps until Step (2) doesn't change C:

(1) Given C, compute a minimum-cost *balanced* assignment $f : P \to C$. (Recall that the cost is $\sum_{y \in P} d^2(y, f(y))$.)
(2) Move each center $x \in C$ to the centroid of the residents that f assigns to x.

As in the analysis of Lloyd's method, each iteration except the last produces a lower-cost assignment, so the algorithm must terminate. Furthermore, at termination the solution is a local minimum with respect to cost: by just moving the centers in C, or just changing f to any other balanced assignment, it is not possible to reduce the cost.

As Fryer and Holden observe [16], truly minimizing the cost is essentially equivalent to minimizing the quantity we called *dispersion* in Section 1.

The subproblem in Step (1) can be solved via Aurenhammer et al.'s algorithm [5]. Instead, our implementation solves it by reducing it to minimum-cost flow (see Section 4). This yields both the stipulated f and (via the dual variables) weights w such that $\mathcal{P}(C, w, f)$ is a balanced power diagram. In the final iteration Step (2) does not change C, so f is also centroidal, and $\mathcal{P}(C, w, f)$ is the desired balanced centroidal power diagram.

At termination, the pair (C, f) is a local minimum in the following sense: by just moving the centers in C, or just changing f (while respecting the balance constraint), it is not possible to reduce the cost. [5]

Given (P, k), Phase One of our algorithm computes a balanced centroidal power diagram $\mathcal{P}(C, w, f)$ using the above method. The corresponding districts are the power cells of $\mathcal{P}(C, w, f)$, clipped to the state boundaries.

Because $\mathcal{P}(C, w, f)$ is a power diagram, Property P1 holds: each district is the intersection the state with a convex polygon. Because the dual graph of these cells is planar, the average number of sides per polygon is less than six; Property P2 holds. Because $\mathcal{P}(C, w, f)$ is a balanced power diagram, Property P3 holds: the populations of the districts are balanced. Because $\mathcal{P}(C, w, f)$ is centroidal, the districts tend to be compact.

Related algorithms sacrificing balance to respect census blocks. Spann et al. [31] propose a variant of the algorithm in which, in Step (1), the assignment f is constrained to fully respect census blocks: for each census block, f must assign all residents within that block to the same center. As discussed earlier, this is desirable but can make the balance condition harder (or even impossible) to achieve. Spann et al. state that they relax the perfect-balance requirement, initially allowing a 20% deviation, and then reduce the allowed deviation with each iteration. In principle, reducing the allowed deviation can increase the minimum assignment cost, so some care is needed to guarantee termination. The paper does not describe precisely how this is done, or precisely how Step (1) is done. It states that the algorithm terminates when the deviation is within 2% of balanced. So, the resulting districts respect census blocks but are not perfectly balanced, deviating by as much as 2%.

Hess et al. [21] (forty years before Spann et al.) propose another variant that respects census blocks (then called "census enumeration districts"). They also implemented their Step (1) by solving a transportation (min-cost assignment), but with some (apparently manual) adjustment to the assignment to maintain connectivity of each district. They do not compute power-diagram weight vectors nor otherwise mention power diagrams. They report population differences of 6.8–9.5% between the resulting districts.

Helbig et al. [20] propose another variant that respects census blocks (then called "population units"). But they use a different cost function—the sum of distances, not the sum of squared distances — so do not achieve a power diagram. The resulting districts are not generally convex. Even noncontiguous districts can result, although this was not observed in practice. Their mathematical program constrains population balance using an indirect heuristic, which interferes with convergence, and they allow their algorithm to stop before reaching a true local minimum. They report population differences of 2.84% between districts.

Other related algorithms. Balzer et al. [6, 7] propose a variant of the algorithm we describe above, differing in that it uses a local-exchange heuristic (updating f by swapping pairs of residents) to carry out Step (1). For some inputs, local-exchange with pairwise swaps is not sufficient to reach a minimum-cost assignment f. Consequently, for some inputs, their algorithm outputs an assignment f that is not actually consistent with any balanced power diagram.

Many other papers on balanced centroidal power diagrams address applications (e.g. in graphics) that have very large instances, and for which it is not crucial that the power diagrams be exactly centroidal or exactly balanced [6, 7, 12, 26, 33]. This class of algorithms prioritize speed, and none are guaranteed to find an assignment f so that (C, f) is a local minimum as described above, or has a balanced centroidal power diagram.

[4] They address the more general problem in which a target population is specified for each center.

[5] Indeed, this is a necessary and sufficient condition on (C, f) for (C, f) to admit a balanced centroidal power diagram $\mathcal{P}(C, w, f)$ (for some w).

3 PHASE TWO: REASSIGNING BOUNDARY BLOCKS

Given input (P, k), Phase One solves the idealized redistricting problem (without census blocks), computing a perfectly balanced centroidal diagram $\mathcal{P}(C, w, f)$. The resulting power cells form idealized, relatively compact districts with the desired properties P1–P3 as described above.

Phase Two, described next, modifies the assignment f so as to meet the practical requirement that districts must respect (not split) census blocks while preserving connectivity. Phase Two itself consists of two parts. The first part establishes which blocks can be assigned to which districts, with some associated constraints we call *dependencies*. The second part finds an assignment that obeys these constraints and minimizes the maximum population difference between districts.

3.1 Computing candidate districts and dependencies

The first part of the Phase Two algorithm operates independently on each power cell. Fix some power cell C. First the algorithm constructs the *census-block graph*, where each census block that intersects C is a vertex, and two vertices are adjacent if their census blocks share a boundary within C. Next, the algorithm Next it identifies *interior* census blocks. —census blocks b that lie strictly within C. The interior census blocks induce a subgraph of the census-block graph. The algorithm computes the largest connected connected component of that subgraph; the census blocks comprising this connected component are unconditionally assigned to the district corresponding to the power cell; we refer to the set of these blocks as the *core* of the district. Generally they comprise the vast majority of blocks intersecting C.

The algorithm then computes a breadth-first-search forest in the census-block graph from the core to the noncore blocks in the graph. For each noncore block B, the *candidate districts* for B with respect to C are all the districts whose cells intersect some ancestor of B in the breadth-first-search forest. The *dependee* of B with respect to C is the parent of B in the breadth-first-search forest if that parent is a noncore block.

The meaning of these notions is that B is allowed to be assigned to any of its candidate districts, as long as its dependee is assigned to the same district. The latter constraint is called a *dependency*.

The above procedure is applied to each power cell.

Lemma 3.1. *Consider the blocks not in any core. Fix a dependency-respecting assignment of noncore blocks to candidate districts. For every block B, if C is the power cell corresponding to the district to which B is assigned then there is a sequence $B=B_0, B_1, \ldots, B_k$ of blocks, all assigned to the same district, such that B_k is in the core of that district's power cell and, for $i = 1, \ldots, k$. B_{i-1} shares a boundary with B_i.*

Because each power cell's core is itself connected, it follows as a corollary that all blocks assigned to a district are connected.

3.2 Finding an assignment with integer linear programming

Part two of Phase Two is responsible for finding an dependency-respecting assignment of noncore census blocks to candidate districts so as to minimize the maximum interdistrict population difference.

In our experiments, we formulated this optimization problem as an integer linear program (ILP) in a standard way, then solved the ILP using the commercial ILP solver, Gurobi. The ILP's are not small. For example, the ILP for California has about 15k constraints, 19k binary variables, and 32k non-zeros in the constraint matrix. Nonetheless, Gurobi solved all the ILPs. For example, California took about thirty seconds.

We speculate that the ILP has a relatively easy combinatorial structure, perhaps having to do with the structure described in the next section, which describes an alternative, dynamic-programming, approach to solving the problem.

3.3 Introduction to the dynamic program

We will now show that the Phase Two problem has special structure that enables it to be solved by a dynamic program. For for the sake of simplicity, we will ignore dependency relationships. We will also assume that the intersection of each census block with each power-cell boundary segment is a single subsegment. It is not hard to adapt the dynamic program to take into account dependency relationships and handle census blocks with more complicated intersections, as long as these are in a sense well-behaved.

Under these simplifying assumptions, the running time of the dynamic program is $n^{O(d)}m$ where n is the per-district target population, m is the number of census blocks, and d is the breadth-first-search depth of the planar dual of the power diagram. We explain the depth d in greater detail below. For now, suffice it to note that it tends to be small in the context of power diagrams for redistricting. For our most complicated state, California, the depth is three.

3.4 Dynamic program and sphere-cut-like decomposition

We will now show that the Phase Two problem has special structure that enables it to be solved by a dynamic program. For the sake of simplicity, we will ignore dependency relationships. We will also assume that the intersection of each census block with each power-cell boundary segment is a single subsegment. The *power-diagram graph* G_D is a graph in which each edge represents a maximal line segment that bounds a power cell, see Figure 2. Consider one such line segment L. Let $L_1, \ldots, L_p$ be the subsegments that are the intersections of census blocks with L, in the order in which they appear on L, not including intersections that include the endpoints of L. Subdivide the edge e corresponding to L, replacing it with a path $v_0e_0v_1e_1v_2 \ldots v_pe_{p+1}v_{p+1}$ where v_0 and v_{p+1} are the original endpoints of e. Each vertex v_i (for $i = 1, \ldots, p$ and possibly also for $i = 0$ and/or $i = p + 1$) corresponds to a subsegment of L, and thus to a census block that intersects L. This is called the *subdivided power-diagram graph*. We denote

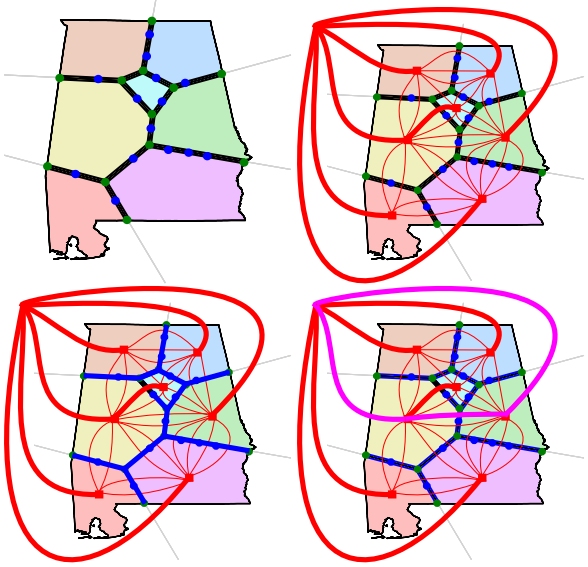

Figure 2: Top left: In bold black the edges of the subdivided power diagram graph. The green vertices are the vertices of the power diagram graph. The blue vertices result from the subdivision of the edges of G_D. The blue vertices (and likely the green vertices) correspond to boundary census blocks. Green and blue vertices together with the bold lines form $\widetilde{G_D}$.
Top Right: Red lines together with red squares respectively define edges and vertices of the dual of $\widetilde{G_D}$. Bold red lines show a breadth-first-search tree T of this graph rooted at the infinite face.
Bottom Left: Blue lines show the interdigitating tree of $\widetilde{G_D}$ associated with T.
Bottom Right: Pink lines show a fundamental cycle in the dual of $\widetilde{G_D}$. This cycle is associated with the edge of T^* that it intersects.

it by $\widetilde{G_D}$ and give an example in the top left of Figure 2. Note that it is a planar graph.

Next, consider the planar dual of $\widetilde{G_D}$. The planar dual has a vertex for each face of $\widetilde{G_D}$ (including the infinite face), and, for each edge e of $\widetilde{G_D}$, a dual edge that crosses e at approximately a right angle. The subdivisions in $\widetilde{G_D}$ give rise to parallel edges in the dual, as shown in the top right of Figure 2.

Let T be any spanning tree of the dual of $\widetilde{G_D}$. Each edge xy of the dual that is not part of T defines a simple cycle in the dual; the cycle consists of xy together with the simple x-to-y path in T. This cycle is called the *fundamental cycle* of xy with respect to T. It passes through some vertices, of which all but at most one (the infinite face) correspond to power cells and thus to districts.

In particular, let T be a breadth-first search tree of the dual. We define the *breadth-first search depth* d to be the maximum number of edges on any root-to-leaf path. Then any fundamental cycle with respect to T passes through at most $2d$ vertices that correspond to districts.

Let T^* be the set of edges of $\widetilde{G_D}$ that do not correspond to edges of T. A basic result in planar graph theory states that the edges of T^* form a spanning tree of $\widetilde{G_D}$. For each edge uv of T^*, let e be the corresponding edge of $\widetilde{G_D}$, and let C be the fundamental cycle of e with respect to T. Orient uv from the vertex enclosed by C towards the vertex not enclosed by C. In this way, T^* becomes a rooted spanning tree of $\widetilde{G_D}$, where the root is a vertex with only one incident edge of T^*. For each vertex u of T^*, let $T^*(u)$ be the set of descendants of u in T^*. For each nonroot vertex u, the parent edge uv corresponds to a cycle $C(u)$ in the dual such that $C(u)$ separates the vertices of $T^*(u)$ from the vertices of $\widetilde{G_D}$ not in $T^*(u)$. Let $D(u)$ denote the set of districts (power cells) corresponding to the vertices on $C(u)$.

Thus we obtain a recursive binary decomposition of the vertices of $\widetilde{G_D}$.[6]

3.5 Dynamic program

Fix a threshold λ. The dynamic program will determine whether there is an assignment of census blocks to districts so that the maximum discrepancy is at most λ.

Let u be a nonroot vertex of T^*. Consider an assignment of the census blocks in $T^*(u)$ to power cells they intersect. Making this assignment (without otherwise changing the population assigned to power cells) induces a change $\Delta(c_i)$ in the population assigned to power cell c_i. We say that this assignment is *feasible* with respect to u if, for every power cell not in $D(u)$, the absolute value of the change is no more than λ. Each feasible assignment induces a labeling of the power cells in $D(u)$; namely, cell c_i is labeled with $\Delta(c_i)$. We call this labeling a *feasible labeling* with respect to u. The number of feasible labelings with respect to u is at most $n^{|D(u)|}$, which is in turn at most n^{2d}.

The dynamic program finds, for each nonroot vertex u, the set of all feasible labelings with respect to u. To do this, the dynamic program works up T^* from leaves to root. For each vertex u, the feasible labelings of u can be derived by considering the feasible labelings of each of its children and the different cells that u itself can be assigned to. For each vertex u, the time to compute the feasible labelings with respect to u is proportional to the product of the number of feasible labelings of each of its children. Since each vertex has at most two children in T^* and each child has at most n^{2d} feasible labelings, the time for each vertex is $n^{O(d)}$. Similarly, from the sets of feasible labelings of the root's children, it can be determined whether there is any assignment of census blocks to power cells for which the maximum discrepancy is at most λ. The total time is $n^{O(d)}m$.

4 STEP (1) OF PHASE ONE

Aurenhammer et al. [5] provide an algorithm that, given the set P of locations of residents and the sequence C of centers, and given a target population for each center (where the targets sum to the total population), finds a minimum-cost assignment f of residents to centers subject to the constraint that the

[6] This decomposition resembles a structure called a *sphere-cut* decomposition [13] (see also the chapter on branchwidth in [22]).

number of residents assigned to each center equals the center's target population. Their algorithm also outputs weights w for the centers such that the assignment f is consistent with $\mathcal{P}(C, w)$. Their algorithm can be used to find a minimum-cost balanced assignment by using appropriate targets.

In the implementation here, we take a different approach to computing the minimum-cost balanced assignment: we use an algorithm for minimum-cost flow. Aurenhammer et al. [5] acknowledge that a minimum-cost flow algorithm can be used but argue that their method is more computationally efficient. As we observe below, the necessary weights w can be computed from the values of the variables of the linear-programming dual to minimum-cost flow.

The goal is to find a balanced assignment $f : P \to C$ of minimum cost, $\sum_{y \in P} d^2(y, p(y))$. Let $u_x \in \{\lfloor m/k \rfloor, \lceil m/k \rceil\}$ be the number of residents that f must assign to center $x \in C$.

Consider the following linear program and dual:

$$\begin{array}{lll} \text{minimize}_a \ \sum_{y \in P, x \in C} d^2(y,x)\, a_{yx} & & \\ \text{subject to} & \sum_{y \in P} a_{yx} = \mu_x & (x \in C) \\ & \sum_{x \in C} a_{yx} = 1 & (y \in P) \\ & a_{yx} \geq 0 & (x \in C, y \in P) \end{array}$$

$$\begin{array}{lll} \text{maximize}_{w,z} \ \sum_{x \in C} \mu_x\, w_x + \sum_{y \in P} z_y & & \\ \text{subject to} & z_y \leq d^2(y,x) - w_x & (x \in C, y \in P) \end{array}$$

This linear program models the standard *transshipment* problem. As the capacities μ_x are integers with $\sum_x \mu_x = |P|$, it is well-known that the basic feasible solutions to the linear program are 0/1 solutions ($a_{yx} \in \{0, 1\}$), and that the (optimal) solutions a correspond to the (minimum-cost) balanced assignments $f : C \to P$ such that $a_{yx} = 1$ if $f(y) = x$ and $a_{yx} = 0$ otherwise. The implementation here solves the linear program and dual by using Goldberg's minimum-cost flow solver [18] to obtain a minimum-cost balanced assignment f^* and an optimal dual solution (w^*, z^*). For any minimum-cost balanced assignment f (such as f^*) the resulting weight vector w^* gives a balanced power diagram $\mathcal{P}(C, w^*, f)$:

Lemma 4.1 (see also [31]). *Let (w^*, z^*) b any optimal solution to the dual linear program above. Let f be any balanced assignment. Then $\mathcal{P}(C, w^*, f)$ is a balanced power diagram if and only if f is a minimum-cost balanced assignment.*

5 EXPERIMENTS

we ran our algorithm on all forty-three US states having more than one congressional district. For each, the algorithm obtained a districting plan in which every two districts differed in population by at most one. For each of these states, we used the following data provided by the US Census Bureau [10]: for each census block, we used the geometric description of the census block, and the population from the 2010 census. In the very rare case where a census block consisted of several polygons, we made the assumption (usually but not always correct) that the census block's population was zero. For each state, for the number of districts we used the number of representatives the state sends to the US House of Representatives.

Examples of our results are depicted in Figure 1 and Figure 3. The reader is directed to http://district.cs.brown.edu for other examples and an interactive viewer. See also [11]. We note that in all cases the Phase One algorithm converged to a local optimum, and the Phase Two found a solution that preserved connectivity (aside from bodies of water) and perfect population balance.

The implementation is available at https://github.com/pnklein/district. It is written in `C++` and `Python3`. Our implementation makes use of a slightly adapted version of a min-cost flow implementation, `cs2` due to Andrew Goldberg and Boris Cherkassky and described in [18]. The copyright on `cs2` is owned by IG Systems, Inc., who grant permission to use for evaluation purposes provided that proper acknowledgments are given. For our experiments, the programs were compiled using `g++` version 7.

6 CONCLUDING REMARKS

In our implementation, we used the randomized method of [3] to initialize the centers. It would be interesting to explore the impact of the initial center locations on the resulting power diagram. Perhaps iterated use of the algorithm with different random starts would provide a suitable probability distribution of districting plans for comparison to existing plans, as in, e.g., [8, 27]. Alternatively, perhaps a careful choice of initial centers could achieve other goals, such as preserving similarity to an existing districting plan or increasing the likelihood of avoiding the splitting of a particular community into different districts.

We have focused in this paper on the Euclidean plane. This ensures that each district is the intersection of the geographical region (e.g. state) with a polygon. Moreover, there is potential for using the fact that the metric is Euclidean to speed up the Phase One (see [1]). However, in view of the fact that the method explored here might generate a district that includes residents separated by water, mountains, etc., one might want to consider a different metric, e.g. to take travel time into account (as is done in [15]). Suppose, for example, the metric is that of an undirected graph with edge-lengths. One can use essentially the same algorithm for finding a balanced centroidal power diagram. In Step 2, the algorithm must move each center to the location that minimizes the sum of squared distances from the assigned residents to the new center location. In a graph, we limit the candidate locations to the vertices and possibly locations along the edges. Under such a limit, it is not hard to compute the best locations.

One might want to incorporate the goal of avoiding splitting larger geographical units such as counties. It would be interesting to explore incorporating into the dynamic program or integer linear program costs for splitting such units.

Acknowledgements. Thanks to Warren D. Smith and Alexander Dubbs for informing us of some important references.

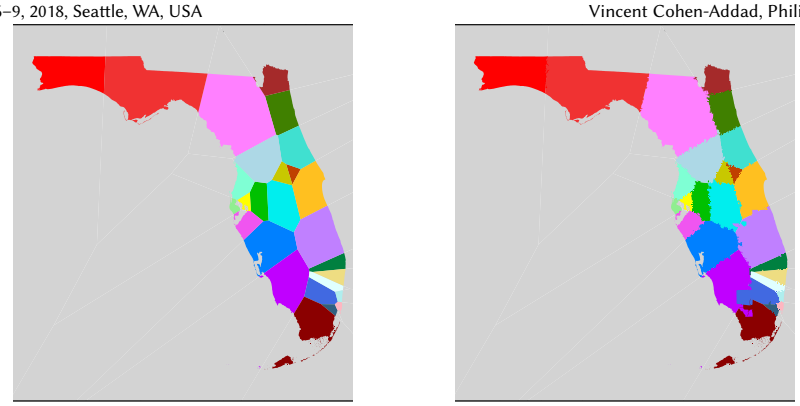

Figure 3: Florida, idealized districting plan (left) and plan that preserves census blocks (right)

REFERENCES

[1] Pankaj K. Agarwal, Kyle Fox, Debmalya Panigrahi, Kasturi R. Varadarajan, and Allen Xiao. Faster algorithms for the geometric transportation problem. In *Proceedings of the 33rd International Symposium on Computational Geometry*, pages 7:1–7:16, 2017.

[2] Micah Altman and Michael McDonald. The promise and perils of computers in redistricting. *Duke J. Const. L. & Pub. Pol'y*, 5:69, 2010.

[3] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

[4] Franz Aurenhammer. Power diagrams: Properties, algorithms and applications. *SIAM Journal on Computing*, 16(1):78–96, February 1987.

[5] Franz Aurenhammer, Friedrich Hoffmann, and Boris Aronov. Minkowski-type theorems and least-squares clustering. *Algorithmica*, 20(1):61–76, 1998.

[6] Michael Balzer and Daniel Heck. Capacity-constrained Voronoi diagrams in finite spaces. In *Proceedings of the 5th Annual International Symposium on Voronoi Diagrams in Science and Engineering*, Kiev, Ukraine, September 2008. 00014.

[7] Michael Balzer, Thomas Schlömer, and Oliver Deussen. Capacity-constrained point distributions: A variant of Lloyd's method. *Proceedings of ACM SIGGRAPH 2009*, 28(3), August 2009.

[8] Sachet Bangia, Christy Vaughn Graves, Gregory Herschlag, Han Sung Kang, Justin Luo, Jonathan C Mattingly, and Robert Ravier. Redistricting: Drawing the line. *arXiv preprint arXiv:1704.03360*, 2017.

[9] D. P. Bourne and S. M. Roper. Centroidal power diagrams, lloyd's algorithm, and applications to optimal location problems. *SIAM Journal on Numerical Analysis*, 53(6):2545–2569, January 2015.

[10] United States Census Bureau. Tiger/line with selected demographic and economic data; population % housing unit counts — blocks. https://www.census.gov/geo/maps-data/data/tiger-data.html. accessed September 2017.

[11] Vincent Cohen-Addad, Philip N. Klein, and Neal E. Young. Balanced power diagrams for redistricting. *CoRR*, abs/1710.03358, 2017.

[12] Fernando De Goes, Katherine Breeden, Victor Ostromoukhov, and Mathieu Desbrun. Blue noise through optimal transport. *ACM Transactions on Graphics (TOG)*, 31(6):171, 2012.

[13] Frederic Dorn, Eelko Penninkx, Hans L Bodlaender, and Fedor V Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010.

[14] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal Voronoi tessellations: Applications and algorithms. *SIAM review*, 41(4):637–676, 1999.

[15] David Eppstein, Michael T Goodrich, Doruk Korkmaz, and Nil Mamano. Defining equitable geographic districts in road networks via stable matching. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 52. ACM, 2017.

[16] Roland G Fryer Jr and Richard Holden. Measuring the compactness of political districting plans. *The Journal of Law and Economics*, 54(3):493–535, 2011.

[17] R. S. Garfinkel and G. L. Nemhauser. Optimal political districting by implicit enumeration techniques. *Management Science*, 16(8):B–495, April 1970.

[18] Andrew V. Goldberg. An efficient implementation of a scaling minimum-cost flow algorithm. *J. Algorithms*, 22(1):1–29, 1997.

[19] Andrew V Goldberg and Robert Kennedy. An efficient cost scaling algorithm for the assignment problem. *Mathematical Programming*, 71(2):153–177, 1995.

[20] Robert E Helbig, Patrick K Orr, and Robert R Roediger. Political redistricting by computer. *Communications of the ACM*, 15(8):735–741, 1972.

[21] S. W. Hess, J. B. Weaver, H. J. Siegfeldt, J. N. Whelan, and P. A. Zitlau. Nonpartisan political redistricting by computer. *Operations Research*, 13(6):998–1006, 1965.

[22] Philip N. Klein and Shay Mozes. Optimization Algorithms for Planar Graphs. http://planarity.org/. accessed June 2018.

[23] Evan Kleiner and Albert Schueller. A political redistricting tool for the rest of us — other approaches to redistricting. *Convergence (MAA)*, November 2013.

[24] Justin Levitt. All about redistricting: Professor Justin Levitt's guide to drawing the electoral lines. http://redistricting.lls.edu/. accessed September 2017.

[25] Jeffrey B. Lewis, Brandon DeVine, Lincoln Pitcher, and Kenneth C. Martis. Digital boundary definitions of United States Congressional districts, 1789-2012, 2013. Retrieved from http://cdmaps.polisci.ucla.edu on June 12, 2018.

[26] Hongwei Li, Diego Nehab, Li-Yi Wei, Pedro V. Sander, and Chi-Wing Fu. Fast capacity constrained Voronoi tessellation. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '10, pages 13:1–13:1, New York, NY, USA, 2010. ACM.

[27] Jonathan C Mattingly and Christy Vaughn. Redistricting and the will of the people. *arXiv preprint arXiv:1410.8796*, 2014.

[28] Anuj Mehrotra, Ellis L. Johnson, and George L. Nemhauser. An optimization based heuristic for political districting. *Management Science*, 44(8):1100–1114, August 1998.

[29] Stacy Miller. The problem of redistricting: The use of centroidal Voronoi diagrams to build unbiased congressional districts. *Senior project, Whitman College*, 2007.

[30] Brian Olson and Warren D. Smith. RangeVoting.org - Theoretical Issues in Districting Algorithms. http://rangevoting.org/TheorDistrict.html, accessed 2017-12-08.

[31] Andrew Spann, Daniel Kane, and Dan Gulotta. Electoral redistricting with moment of inertia and diminishing halves models. *The UMAP Journal*, 28(3):281–299, 2007.

[32] Lukas Svec, Sam Burden, and Aaron Dilley. Applying voronoi diagrams to the redistricting problem. *The UMAP Journal*, 28(3):313–329, 2007.

[33] Shi-Qing Xin, Bruno Lévy, Zhonggui Chen, Lei Chu, Yaohui Yu, Changhe Tu, and Wenping Wang. Centroidal power diagrams with capacity constraints: Computation, applications, and extension. *ACM Transactions on Graphics*, 35(6):1–12, November 2016.