PA

A New Approach for Developing Neutral Redistricting Plans

Daniel B. Magleby[1] and Daniel B. Mosesson[2]

[1] *Department of Political Science, Binghamton University, SUNY, Binghamton, NY 13902, USA.*
Email: dmagleby@bingamton.edu
[2] *Center on Democratic Performance, Binghamton University, SUNY, Binghamton, NY 13902, USA.*
Email: dmosess1@binghamton.edu

Abstract

Computers hold the potential to draw legislative districts in a neutral way. Existing approaches to automated redistricting may introduce bias and encounter difficulties when drawing districts of large and even medium-sized jurisdictions. We present a new algorithm that can neutrally generate legislative districts without indications of bias that are contiguous, balanced and relatively compact. The algorithm does not show the kinds of bias found in prior algorithms and is an advance over previously published algorithms for redistricting because it is computationally more efficient. We use the new algorithm to draw 10,000 maps of congressional districts in Mississippi, Virginia, and Texas. We find that it is unlikely that the number of majority-minority districts we observe in the Mississippi, Virginia, and Texas congressional maps of these states would happen through a neutral redistricting process.

Keywords: geographic information systems, graphs, simulation methods, Monte Carlo simulation, data analysis algorithms

1 Introduction

Geographically defined legislative districts present a temptation for politically motivated mapmakers. It has long been understood that those responsible for drawing districts may pervert mechanisms of representation by drawing the boundaries of districts in ways that advantage (or disadvantage) certain groups. A recent example illustrates this point. In 2010, Scott Walker won election as governor of Wisconsin. In addition to Walker's victory, Republicans running for seats in Wisconsin's State Senate and Assembly benefited from a wave of discontent with Democrats and established majorities in both chambers of Wisconsin's state legislature. The victory was well timed because following the census in 2010, Walker and legislative Republicans had the opportunity to redraw the state's state-legislative and congressional districts in advance of the 2012 election without meaningful Democratic opposition. The result was a set of legislative districts that ensured significant Republican majorities in Wisconsin's legislature and congressional delegation. When a group of Democratic voters challenged the map of state-legislative districts in federal court, an expert testifying on behalf of the Democrats argued that the Republican drawn districts potentially undermined a core principal of representative democracy, majority rule. The expert argued that it would take a "political earthquake" for Democrats to win a majority of seats in the legislature even though Wisconsin Democrats frequently won a majority of votes cast in legislative elections and in state-wide races (Johnson 2016a). Wisconsin Republicans countered that their advantage grew out of the concentration of Democrats in urban areas, and the map they drew simply reflected the more efficient distribution of Republicans in suburban and rural areas (Johnson 2016b). The question left for the panel of federal judges hearing the case was whether the Republicans had pushed their advantage too far.

In this article, we advocate for a method for evaluating just how perverse a set of legislative districts are. The method relies on the development of a counterfactual—a null distribution of possible redistricting outcomes. To have confidence that an observed map is extreme enough

Political Analysis (2018)
vol. 26:147–167
DOI: 10.1017/pan.2017.37

Corresponding author
Daniel B. Magleby

Edited by
R. Michael Alvarez

to warrant remedy, it is necessary to characterize a representative distribution of possible redistricting outcomes. In practice, this requires using a computer to draw a large number of alternative maps according to a set of neutral criteria.[1] In comparing an observed outcome to a large set of neutral alternatives, we may determine the degree to which observed maps differ from what we would expect a neutral process to yield.[2]

We are not the first to propose the principal upon which the method is based nor are we the first to propose an automated process for drawing legislative districts. In 1961, economist and future Nobel laureate William Vickrey posited that it was possible to eliminate "chicanery" from the way legislative districts are drawn. He suggested that a neutral algorithm could solve the *redistricting problem*, the process of dividing a set of geographic units into a smaller number of districts, in an "automatic and impersonal" way (110). In the years and decades that followed, computers made it possible for scholars to implement methods for developing optimal, neutral maps of legislative districts (see Weaver and Hess 1963, Nagel 1965, Garfinkel and Nemhauser 1970, Browdy 1990, Bozkaya, Erkut, and Laporte 2003, Chou and Li 2006 and Fryer Jr and Holden 2011). At the same time, analysts recognized that retrieving an unbiased counterfactual of a jurisdiction's legislative districts is useful in a variety of applications (see Engstrom and Wildgen 1977, O'Loughlin and Taylor 1982, Cirincione, Darling, and O'Rourke 2000, McCarty, Poole and Rosenthal 2009, Chen and Rodden 2013, Chen and Cottrell 2016, Fifield *et al.* 2015a, Tam Cho and Liu 2016). In spite of Vickrey's assertion that developing a neutral process to draw legislative districts would be "not at all difficult" (1961, 110), computer-automated redistricting turned out to be at best a challenging problem and at worst an intractable one in complex redistricting scenarios. There are at least two difficulties automated processes encounter; first, an algorithm designed to generate maps of districts may take a biased sample of all possible legislative maps (Altman *et al.* 2015); and second, many of the most interesting and important redistricting problems are so complex that automated methods may fail to efficiently produce a meaningful distribution of possible alternative maps.[3]

In this article, we present an algorithm that avoids many of the issues related to bias and efficiency exhibited by alternative approaches to automated redistricting. To demonstrate the advantages of our approach, first, we subject the algorithm to tests designed to reveal bias and find no evidence of bias in the way the algorithm draws districts. In one test, we apply the algorithm to a redistricting scenario designed to reveal whether the algorithm we propose produces districts of one variety more often than districts of another variety. We find that it does not. As another test for bias, we compare the sample of maps drawn by the algorithm to a known distribution of possible districts. We find that the algorithm retrieves an accurate approximation of the known distribution. Second, we show that the algorithm we propose is remarkably efficient. Using asymptotic analysis, a technique used by computer scientists to evaluate the complexity of algorithms, we compare our proposed algorithm to a commonly used alternative. We show that our algorithm is many orders of magnitude more efficient than the alternative approach. In short, our contribution is to present an algorithm that retrieves an approximation of the null distribution with no indication of bias in a fraction of the time it takes alternative approaches to produce maps of similar jurisdictions.

To address the redistricting problem, our approach relies on graph partitioning methods developed in computer science. The class of graph partitioning algorithms upon which we base our method are well understood by computer scientists (Kernighan and Lin 1970, Hendrickson

1 By neutral, we mean to say that the process of generating the counterfactual only considers constitutionally relevant characteristics of the geography it divides into legislative districts: contiguity and population parity. In comparing an observed outcome to a large set of neutral alternatives, we may determine the degree to which observed maps differ from what we would expect a neutral process to yield.

2 In practice, state-level maps of legislative districts are drawn by state legislatures or redistricting commissions.

3 One possible solution is to implement alternative algorithms that are less efficient on more powerful computers; however, to generate a meaningful number of maps some scholars have resorted to using supercomputers (Remmert 2016). Inefficient algorithms put automated redistricting out of reach for scholars with access to more typical computing resources.

and Leland 1995, Karypis and Kumar 1995, 1998); however, applying these processes to the political problem of redistricting is new. In computer science applications, graph partitioning algorithms are used by programmers to divide related computational tasks between a computer's processor. In our application to politics, we use the algorithm to assign adjacent geographic units to legislative districts.

To further demonstrate the algorithm's capabilities, we apply it to the substantive problem of congressional districts in a medium (Mississippi), large (Virginia), and very large (Texas) states. These states each have significant minority populations and a history of using the redistricting process to discriminate against those minority groups. We use the algorithm to draw 10,000 maps using census block-level geographic and demographic data. The simulated maps allow us to retrieve a null distribution of the number of majority-minority districts we would expect from a neutral redistricting process within each state. We find that each state's 2012 congressional map has more majority-minority districts than we would expect from a neutral redistricting process. The finding is significant because, since its decision in *Shaw v. Reno*,[4] the Supreme Court has been skeptical of redistricting plans that go to extreme lengths to generate majority-minority districts. Our findings suggest that absent proactive steps on the part of mapmakers, a minority population would be denied the ability to elect "the minority's preferred candidate"[5] in the states we analyze.

The article proceeds as follows. In Section 2, we present a method for generating a neutral counterfactual against which existing maps may be evaluated. In Section 3, we evaluate the algorithm for bias and consider the algorithm's efficiency. Section 4 applies the method to three cases: Mississippi, Virginia, and Texas. In Section 5, we describe how the algorithm could be adapted to account for constraints beyond constitutional requirements of contiguity and population parity. We summarize our conclusions in Section 6 and suggest some further applications of our proposed method of neutral redistricting.[6]

2 A Graph Partitioning Approach to Redistricting

In this section, we propose a method for estimating the distribution of possible maps using a graph partitioning approach. We propose a new algorithm for sampling possible redistricting plans. Finally, we demonstrate how the algorithm would work on a simple redistricting problem consisting of nineteen geographic units and two legislative districts.

2.1 Graph partitioning as a solution to the redistricting problem

Framing redistricting as a graph partitioning problem allows us to simplify the process of drawing maps. As a result, we can generate a set of neutral maps of legislative districts under the constitutional constraints that districts be contiguous and contain equal populations. We conceive of a map as a undirected vertex-weighted graph in which vertices are weighted according to population, but edges are assigned no meaningful weight.[7]

Our approach can be summed up as follows: First, we convert the geographic data into a graph where every geographic unit becomes a vertex weighted by the population of the geographic unit. The graph's edges represent shared borders between geographic units. We then partition the graph (a simpler version of the graph derived from the map) into the required number of districts

4 509 U.S. 630 (1993).
5 *Thornburg v. Gingles*, 478 U.S. 30 (1986).
6 Data for replication of all analyses described and summarized in this article are available through the Harvard Dataverse (Magleby and Mosesson 2017). An implementation of the algorithm we propose for developing neutral redistricting plans is available from the editors of *Political Analysis* as an executable binary for the purposes of replication of the analyses described in this article. Likewise, an implementation of the algorithm is available from the authors for purposes of replication and other scholarly applications.
7 We focus our discussion on vertex-weighted graphs, but the class of graph partitioning algorithms that we apply to the problem of redistricting can account for weights applied to edges between vertices. By incorporating additional information about the relationship between vertices (geographic units), we may be able to address the additional constraints that states place on the types of legislative maps they produce.

that meet the legal requirements of being both contiguous and containing equal population. Since it is often infeasible to have districts that are *exactly* balanced, the algorithm is satisfied by districts with populations that neither exceed nor fall below certain bounds.

2.2 An algorithm for drawing neutral districts

The algorithm we propose for achieving a contiguous and balanced partition has four steps.[8] First, the algorithm *coarsens* the graph, grouping together connected vertices of the graph into contiguous *multinodes*. Second, the algorithm partitions the multinodes of this simplified graph into the desired number of contiguous districts using a breadth first search. Third, it *uncoarsens* the graph, so that the graph is again made up of the original vertices, but all vertices are now associated with a particular district from the previous step. Finally, it refines the graph by moving vertices into and out of districts until the populations of the districts are balanced. Refinement continues until either the established tolerances are met, or the graph ceases to become more balanced.[9] We consider a weighted graph $G_0(A_0, E_0, w)$ where w is a function that maps A_0 onto a set of weights. We limit our discussion to the intuition behind the algorithm and provide a formal description of our proposed process in appendix A.

Step 1 ("Coarsening")

We transform G_0 into a series of smaller graphs $G_1, G_2, G_3, \ldots, G_m$ where vertices in G_t are consolidated into *multinodes* in G_{t+1}. These multinodes are an intermediate construct that allow the algorithm to perform a more efficient partition, but they do not represent a politically meaningful unit. The number of multinodes is not particularly important, and neither do they have to be particularly balanced. The important thing is that they are internally contiguous, and that they are not *too* uneven.[10]

In our case, since the graph has unweighted edges, the algorithm randomly selects an initial vertex.[11] The algorithm then selects vertices from among the neighbors of that vertex and joins those vertices into a multinode. The algorithm repeats this process until all vertices are assigned to one of several multinodes. The intention is to simplify the graph in order to simplify the problem of partitioning the graph. Critically, all the information of a multinode's constituent nodes, their connection to other vertices and their weight, is preserved when vertices are collected into multinodes. Thus one multinode is adjacent to another multinode if at least one constituent vertex in each are adjacent to one other. Likewise, the weight of a multinode is the sum of the weights of its constituent nodes.

Step 2 ("Partitioning")

Once the graph has been simplified, the algorithm generates partition P_m. It randomly selects a multinode with which to start. It then selects a neighboring multinode to add to the district and continues to add randomly selected neighbors until roughly $1/k$ of the available weight is in the district where k is the number of districts into which the vertices (multinodes) should be divided.[12] The algorithm discards discontiguous graphs that result when the process yields a set of multinodes that cannot be joined because they are separated by contiguous districts.

8 We weight vertices by population. If the districts produced by the algorithm are balanced (of equal weight), they also have equal population in this application of the algorithm.

9 In practice, the algorithm may produce imbalanced districts if the refinement process fails to produce a more balanced graph. In those instances, the algorithm fails to converge, and we simply discard that simulated map and start the process with a new random seed.

10 The algorithm will behave correctly if the multinodes are very uneven, but it will take longer to converge on a balanced graph in the last step of the algorithm, so this is a performance concern, but not a correctness issue.

11 It is possible to assign weight to the connections (edges) between nodes. For the purpose of drawing neutral maps of contiguous, equipopulous districts in this article, we do not take this information into account.

12 This means that the algorithm begins by randomly adding multinodes that are contiguous to the multinode initially selected by the algorithm. If it exhausts all options among the vertices that are contiguous with the first nodes, it only adds multinodes that are contiguous to multinodes that are contiguous to the first multinodes, and so on.

Step 3 ("Uncoarsening")

In series of steps, the algorithm reverses the aggregating process that yielded G_m. At the beginning of each step t, the constituent elements of multinodes in G_{t-1} preserve their assignment to a particular district in P_{t-1}.

Step 4 ("Refinement")

Generally speaking, the purpose of this step is to make each component of a partition P_t as balanced as possible. At each step of uncoarsening the algorithm reassigns elements of one district into an adjacent district if reassignment would bring the districts closer to weight parity. This requires the implementation of an additional balancing algorithm to determine which elements of a partition should be (or should not be) reassigned. In practice, there exist several algorithms that can refine the graph into a more balanced partition in the way we describe. We apply one of these, the Kernighan–Lin algorithm (KL) (Hendrickson and Leland 1995, Kernighan and Lin 1970), to refine the graph at each step of uncoarsening. KL is a widely used algorithm for balancing graphs in computer science since it accounts for the weights of vertices and edges as it seeks to produce a more balanced map.[13]

Step 5 ("Repeat")

This step is the easiest of all, as it is just a check to see if the partition meets the required balance (population equality) conditions. If the resulting partition is a set of contiguous and balanced districts, we record the partition for later analysis. If not, the flawed partition is discarded and the algorithm restarts.

2.3 An example

Consider a simple application of our method of drawing legislative districts to a set of nineteen contiguous geographic units represented in Figure 1(a). The goal is to partition this map into a plan with two contiguous districts in which the population of both districts is balanced. The set of geographic units represented in Figure 1(a) is globally contiguous—that is, every unit in the map could potentially be included in a contiguous district with any other unit in the map. The algorithm proceeds by, first, simplifying the map into a connected graph with nineteen vertices and thirty-nine edges; second, collecting vertices into an even simpler graph; third, partitioning the simpler graph; and finally, refining the graph to ensure that the resulting districts have balanced (equal) populations. In this example, the result of this process is a single "partition" of the map into a set of two districts with equal populations.

First, the algorithm begins by simplifying the map into a connected graph. Using the map in Figure 1(a), we may determine which geographic units share any portion of its border with any other geographic unit.[14] We represent the simplified graph of the map in Figure 1(b). Each geographic unit is represented as a circular node. An edge (line) connects the vertex representing a unit to the vertex representing other units with which it is contiguous. In this simplified version of the original map, there are nineteen vertices connected by thirty-nine edges. This representation of the map preserves all of the information necessary for generating contiguous districts.

Second, the algorithm randomly collects vertices into multinodes creating a simpler graph. In the example, the algorithm assigns groups of four or five vertices to one of four multinodes. We represent this process in Figure 1(c). Each of the vertices assigned to a multinode shares an edge with at least one vertex in the multinode making each multinode internally contiguous. We index

13 We assume the edges between vertices have no weight; however, application of the algorithm on graphs with weighted edges is a possible extension of the analysis we present in Section 4.

14 This variety of contiguity is sometimes called "queen contiguity" in reference to the directions players may move the queen in a game of chess. Queen contiguity contrasts with "rook contiguity" in that units need only share a point of contiguity in order to be considered contiguous.

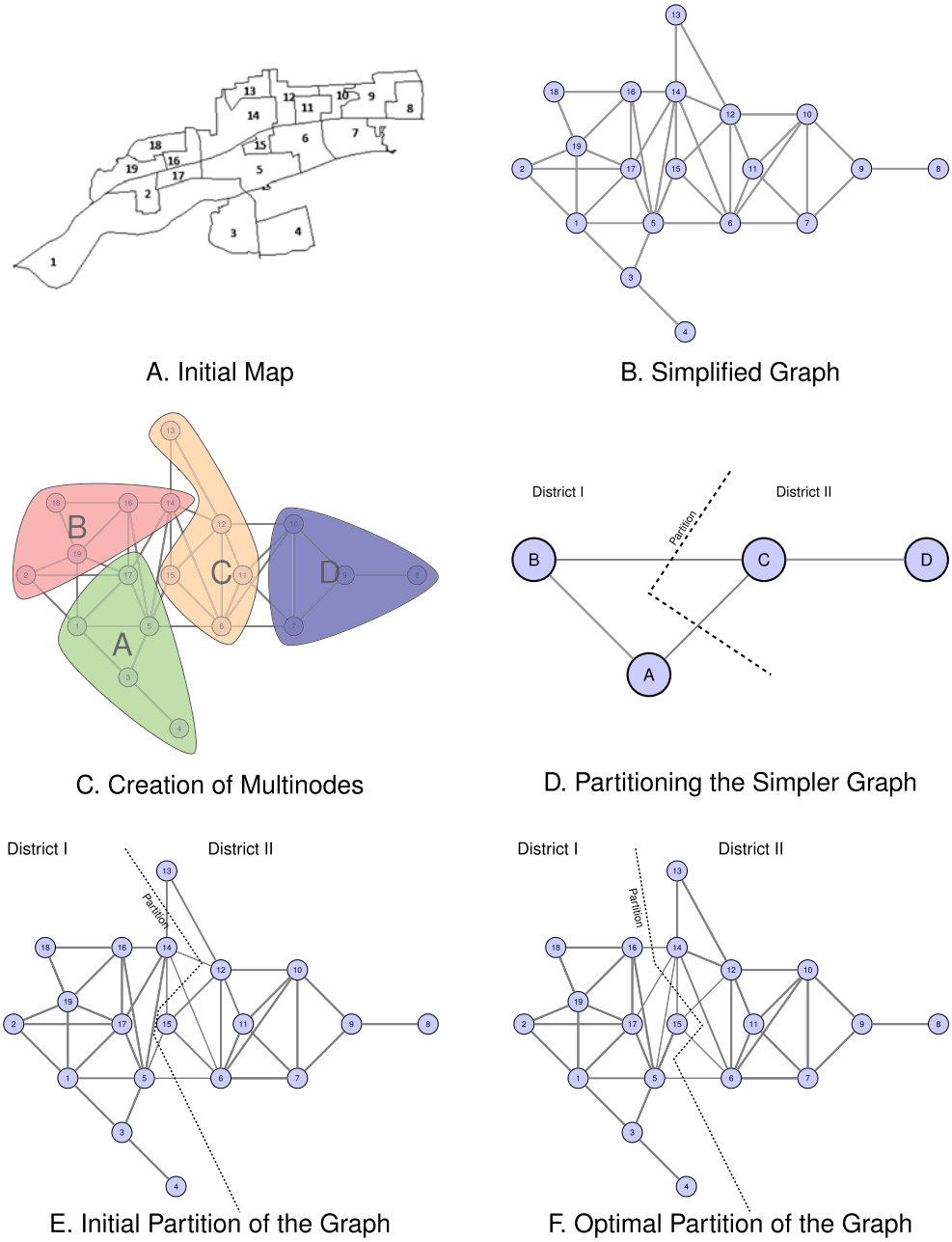

Figure 1. An example of how the algorithm partitions a map consisting of nineteen contiguous geographic units.

these multinodes as A, B, C, and D. The process of collecting these vertices into multinodes is random, so in any iteration of the algorithm, we can expect that two neighboring vertices will end up in different multinodes with positive probability. Observe that if two multinodes have constituent vertices that were contiguous in Figure 1(b), those multinodes are contiguous in Figure 1(c). That is, the multinodes preserve the contiguity of their constituent nodes.

Third, the algorithm partitions the simpler graph. It assigns each multinode (and the multinode's constituent nodes) to a district. This process is represented in Figure 1(d). Here, the algorithm generates the partition $\{A, B\}, \{C, D\}$. Note that not all combinations of these multinodes would be valid. While A, B, and C are all contiguous, the vertices in D are only contiguous with vertices C.

Finally, the algorithm refines the graph in order to ensure that the resulting districts have balanced (equal) population. The graph partitioning problem represented in Figure 1(d) is significantly simpler than the problem of finding a valid partition of the graph in Figure 1(b) with nineteen nodes, but this comes at the cost of balance. Since the multinodes were generated randomly, the partition represented in Figure 1(d) is not likely to be balanced. We represent the process of refining the partitioned graph in figures 1(e) and 1(f). Kernighan–Lin swaps randomly selected vertices across the border generating an alternative partition. If the alternative partition is closer to balanced than the initial partition, it keeps the alternative and throws out the initial partition. It stops once switching vertices does not generate a more balanced graph.

We limit our example to the generation of a single partition of the map in Figure 1(a); however, analysis of observed maps requires that we compare those maps to a neutral counterfactual. Since the process of creating multinodes began with the random selection of a vertex to which adjacent vertices were added, each time we repeat the algorithm in a large enough graph, we arrive at an alternative, balanced partition with positive probability. Since the algorithm only considers the contiguity and weight (population) of each node, the result is a distribution of neutrally drawn districts.[15] We can then compare the actual set of legislative districts to the distribution of districts we observe and characterize the probability that the actual partition created would have emerged from a neutral process.

3 Bias and Efficiency

The algorithm we describe in the previous section avoids many of the shortcomings of alternative algorithms. Specifically, we find no evidence of bias in the way the algorithm draws districts. We subject our proposed algorithm to two tests for bias. The first test is designed to reveal if the algorithm we propose produces districts of one variety more often than districts of another variety. We find that it does not. The second test is designed to evaluate the algorithm's ability to retrieve an approximation of the underlying distribution of possible districts. We compare a large sample of maps drawn by the algorithm to a known distribution of possible districts. We find that the distribution of maps the algorithm retrieves is very similar to the known distribution.

In this section, we also demonstrate that our proposed algorithm is remarkably efficient. Using asymptotic analysis, a technique used by computer scientists to evaluate the complexity of algorithms, we compare our algorithm to a commonly used alternative proposed by Cirincione, Darling, and O'Rourke (2000). We show that the algorithm we propose is many orders of magnitude more efficient than the alternative approach.

3.1 Bias

In order to evaluate the possibility that our proposed algorithm draws maps in a biased way, we apply the algorithm we propose for drawing districts to a problem for which we know the solution (Altman, Gill, and McDonald 2004). First, to show that our proposed algorithm does not produce districts of one variety more often than districts of another variety, we apply our algorithm to a square (100×100) geographic space with 10,000 units all of which contain the same number of people and evaluate the resulting maps to see if one type of district is more likely than others. We find that one type of district is not more likely than another. Second, to evaluate the algorithm's ability to approximate the underlying distribution of possible districts, we apply the algorithm to a much simpler space for which we know the true distribution of possible districts. We find that the algorithm approximates the true distribution.

15 We do not constrain the algorithm to generate maps with compact districts although the Court has expressed a preference for maps with this feature (See Shaw v. Reno, 509 U.S. 630 (1993); Miller v. Johnson, 515 U.S. 900 (1995)). As it turns out, districts generated by the process we present in this article seem to avoid the most extreme varieties of noncompactness in empirical applications.

3.1.1 *Bisecting a symmetrical space*

Demonstrating bias in a sample from a distribution as large and complex as the set of possible electoral maps is a complicated proposition. In practice, the geographic units that constitute the basis of jurisdictions' electoral maps have irregular populations and have varying numbers of neighbors to which they are adjacent. One way to approach the problem is to consider a case for which we know the characteristics of an unbiased sample. In this instance, we partition a hypothetical symmetrical set of geographic units with equal population. Since it is identical vertically and horizontally, we should observe that the number of maps that bisect the space horizontally should be close to equal to the number of maps that bisect the space vertically. Altman *et al.* (2015) propose a similar test using a smaller grid, and they find that several commonly used redistricting algorithms are more likely to produce one type of map than another. We implement the test on a large grid containing roughly the number of units in a medium-sized state's map of electoral precincts.[16]

Consider a square geographic space with 10,000 units that each contain the same number of residents. For simplicity, suppose that there are two possible outcomes of the process: districts that bisect the space horizontally and districts that bisect the space vertically. Observe that such a space is symmetrical vertically and horizontally. If the goal is to divide such a space into two districts with equal population, an unbiased process should be as likely to bisect the space horizontally as it is to bisect the space vertically. We apply our proposed algorithm to this simple problem. First, we partition a grid of 100×100 (10,000) units 1 million times (Magleby and Mosesson 2017). Next, we determine how many districts bisect the space horizontally, how many bisect the grid vertically.

In Figure 2 we plot the centroid of one district from 1,000 two district maps produced by the algorithm from the hypothetical grid. Examples A, B, C, and D in Figure 2 represent the types of bisections the algorithm produces. Each bisection has a darker and lighter "district", and we include arrows that point to the location of the darker district's centroid on the example grid. Above the plot, we include a histogram representing the density of observed centroids on the x-axis. To the right of the plot, we include a histogram of the density of observations on the y-axis. While not identical, these distributions are very similar for the 1,000 observations by visual examination.

Observe that all centroids fall between the 24th and 76th units on the x-axis and y-axis. To identify horizontally and vertically oriented plans, we divide the range of centroids on each dimension into three regions. We consider districts with centroids between the 37th and the 63rd unit on the x-axis to be horizontal. In like manner, we consider districts with centroids between the 37th and the 63rd unit on the y-axis to be vertical. If the algorithm produces maps in an unbiased way, then half the maps should have districts with centroids in the vertical region and half the maps should have districts with centroids in the horizontal region.

In 1 million trials, we find that 500,644 maps have a vertical orientation (a centroid in the region between the 37th and the 63rd unit on the y-axis) and 499,356 have a horizontal orientation (centroid in the region between the 37th and the 63rd unit on the x-axis). The resulting proportion of maps with vertically oriented districts (0.500644) is not statistically different from 0.5, our expected result from an unbiased process ($\chi^2 = 1.659, p = 0.1977$). Thus, we fail to find statistically significant evidence of bias in the orientation of the 1 million maps created by the algorithm. In addition, we only observe a small substantive difference (0.0006) from what we would expect from an unbiased mapmaking process.

16 Such a test has substantive implications. For example, consider jurisdictions with geographically segregated racial populations like Detroit, Michigan. Historically, neighborhoods north of a particular street (8 Mile Road) are predominantly white. For decades, neighborhoods to the south 8 Mile Road have been predominantly black. A sample taken by an algorithm that is more likely to partition Detroit and its suburbs vertically (north to south) or horizontally (east to west) is going to misrepresent the number of majority black districts in the underlying set of possible districts.

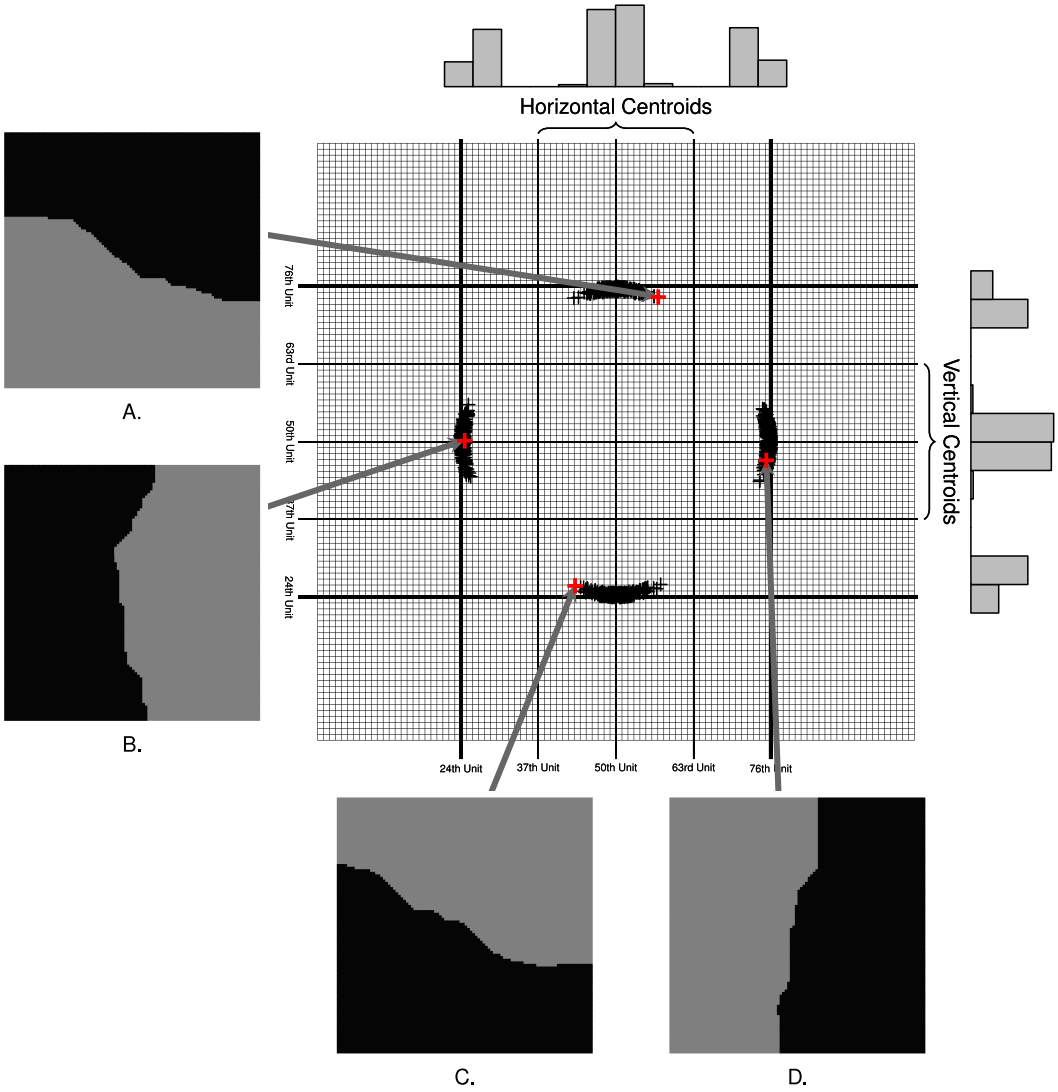

Figure 2. Plot of centroids for 10,000 two-way partitions of a 100 × 100 grid of geographic units with equal population. If the algorithm draws uniformly from the distribution of all possible districts, horizontally oriented districts (like examples *A*. and *C*.) should be as likely as vertically oriented districts (like examples *B*. and *D*.). We indicate the location of the darker district's centroid in the plot with a lighter colored + symbol.

3.1.2 *Sampling from a known distribution*

We apply our algorithm to a small jurisdiction consisting of twenty-five geographic units made available by Fifield *et al.* (2015b). We set the algorithm to partition the space into three districts with populations that deviate by no more than 10% from population parity (Magleby and Mosesson 2017). We then compare the distribution of maps produced by the algorithm we propose to the distribution of all possible maps of the jurisdiction provided by Fifield *et al.* (2015b). A Kolmogorov–Smirnov (KS) test indicates that the two distributions are not exactly the same;[17] however, we find that the distribution of maps retrieved by our process approximates the true distribution enumerated by Fifield *et al.* for maps balanced to 10% population parity. On the other hand, when we constrain our analysis to maps that meet more standard levels of population parity (±1.5%),

17 We note that the KS statistic may not be an appropriate measure for the comparisons between these distributions. For example, KS does not deal with ties, so we cannot be sure that the estimated *p*-values are exact. Similarly, the test presumes that the underlying distributions of possible maps are continuous. Given that Fifield *et al.* enumerate all of the possible maps, and the number of possible maps is relatively small (extremely small in the case of the more balanced maps), we know that the distributions do not satisfy the test's assumption of continuity.

Table 1. We summarize a comparison of the mean, median, standard deviation, and kurtosis of the true distribution of partisan skew of all possible maps containing districts that deviate by no more than 10% ($N = 927$) to the distribution of maps drawn by the algorithm we propose that in which the district population deviates by no more than 10% ($N = 40,000$). Below, we summarize the comparison between the mean, median, standard deviation, and kurtosis of all possible maps containing districts that deviate by no more than 1.5% ($N = 6$) to the distribution of maps drawn by the algorithm we propose in which the district population deviates by no more than 1.5% ($N = 1691$).

Maps balanced to ±10%	**True distribution**	**Proposed algorithm**
Mean	0.003	0.009
Median	0.009	0.011
Standard deviation	0.020	0.016
Kurtosis	3.665	3.598
N	927	40,000
Kolmogorov–Smirnov comparison of distributions	$D = 0.1587$	$p < 0.001$
Subset of maps balanced to ±1.5%	**True distribution**	**Proposed algorithm**
Mean	−0.004	0.006
Median	−0.010	0.009
Standard deviation	0.015	0.007
Kurtosis	1.727	6.378
N	6	1691
Kolmogorov–Smirnov comparison of distributions	$D = 0.5454$,	$p = 0.057$

a KS test does not permit us reject the null hypothesis of equivalence between the distribution of possible balanced maps and the distribution of balanced maps drawn by the algorithm we propose.

To summarize the maps' characteristics, we focus on the level of partisan skew in each map, specifically the skew in the distribution of Democratic votes across districts in a map.[18] We consider partisan skew in the set of possible maps, as reported by Fifield *et al.* (2015b), balanced to ±10% to a set of maps drawn using our proposed algorithm also balanced to ±10% we then repeat the comparison for the more realistic level of partisan parity, ±1.5%. In Table 1, we compare four characteristics of the true distribution of partisan skew along to the same four characteristics of the distribution of partisan skew of the maps drawn by the algorithm we propose. While characteristics of the distribution of maps produced by the algorithm approximate the characteristics of the distribution possible maps, the distributions are not exactly equivalent. In Table 1, we include the Kolmogorov–Smirnov (KS) statistic for the comparison between the distribution recovered by the algorithm we propose and the true distribution as reported by Fifield *et al.* (2015b). The KS test yields a test statistic of 0.1587. Thus, we can reject the null hypothesis that the distribution of maps produced by the algorithm is exactly the same as the distribution enumerated by Fifield *et al.* (2015b).

Important caveats apply to the comparisons summarized in Table 1. First, the KS test indicates that the two distributions are not identical; however, this finding should not be construed to mean that the sample of maps created by the algorithm does not approximate the distribution of all possible maps. The two distributions are very *similar*, but we cannot claim that they are *equivalent*. Second, the set of maps enumerated by Fifield *et al.* (2015b) is not particularly balanced, that is,

18 To find a map's skew, we calculate the proportion of votes cast for Democrats in each of the three districts. We then subtract the map's mean district-level Democratic vote proportion from the map's median district-level Democratic vote proportion. If the difference is negative (positive), Democratic (Republican) votes were overconcentrated in one or two districts in the map. If the difference is 0, the Democrats are neither advantaged or disadvantaged by the lines drawn by the algorithm. This is the measure proposed by McDonald and Best (2015) for detecting packing gerrymanders. Determining the level of partisan skew in a map is also one element of Wang's three pronged test for evaluating partisan gerrymandering (2016).

the population deviation between the largest and smallest districts in the set of possible maps is relatively large (±10%).[19] In the bottom panel of Table 1 we present the characteristics of the set of maps in Fifield *et al.*'s data that are balance to a more typical level of population deviation (±1.5%). We likewise present summary statistics from the subset of maps produced by the algorithm that conforms to the more typical population constraint (1691/40,000). The comparison is informative in that it shows how few of the possible maps in Fifield *et al.*'s data achieve standard levels of population parity (6/927). As with the larger sets of maps, the distributions of maps approximate one another. In contrast to the comparison between the larger sets of maps, the KS test does not show a significant difference between the set of possible maps with population parity and the set of subset of maps produced by the algorithm with population parity ($p = 0.057$). In sum, the KS test does not permit us reject the null hypothesis of equivalence between the distribution of possible balanced maps and the distribution of balanced maps drawn by the algorithm we propose.

3.2 Efficiency

In addition to showing no indication of bias, the algorithm we propose is extremely efficient. We demonstrate the algorithm's efficiency by considering its algorithmic complexity and show how that complexity scales with the complexity of the redistricting problem. Once we understand the algorithmic complexity of our process, it is possible to contrast the algorithm to the process proposed by Cirincione, Darling, and O'Rourke (2000), a commonly used alternative. While other automated redistricting processes exist, Cirincione, Darling, and O'Rourke's process is a useful comparison because it is acknowledged to be the "representative" redistricting algorithm (see Altman *et al.* 2015, Fifield *et al.* 2015a).[20] We show that, for practically all redistricting problems, the algorithm we propose should outperform the commonly used alternative.

An asymptotic analysis of the algorithm we propose characterizes its algorithmic complexity and demonstrates how that complexity scales with the complexity of the redistricting problem. We may conceive of the algorithm we describe as a function that maps some number of geographic units, n, into some number, D, of contiguous electoral districts. By analyzing the asymptotic behavior of the function that describes the algorithm, we determine the way that processing time will increase as the redistricting problem becomes more complex holding computational resources constant. The redistricting problem becomes more complex as either n or D or both increase. In Appendix B we show that the algorithm we propose scales according to the following function.

$$f(n, D) = n^2 \log nD \log D. \tag{1}$$

The contour plot in Figure 3(a) represents the way that (1) scales for different values n (x-axis) and D (y-axis). Each line represents the $\log_{10}$ of constant runtime. For example, we show in the graph that the algorithm should take the same amount of time to divide 100,000 units into 120 districts as it would take to divide 600,000 units into ten districts. The absolute value of the numbers associated with each contour line is less important than the relative increase. The graph indicates that, holding D constant, the algorithm's runtime increases in n. The same is true in reverse: holding n constant, the marginal impact on expected runtime increases with each additional district. Take the case of a state legislature with 100 members of its lower house. Our asymptotic analysis suggests that if it took 0.01 seconds to draw a map of 100 districts from roughly 100,000 geographic units, it would take about 0.1 second to draw a map of 100 districts from

19 This level of imbalance is close to violating constitutional constraints on population parity.

20 In their widely used BARD package for *R*, Altman and McDonald (2011) implemented Cirincione, Darling, and O'Rourke (2000) making it one of the most widely used algorithms for the purposes of automated redistricting.

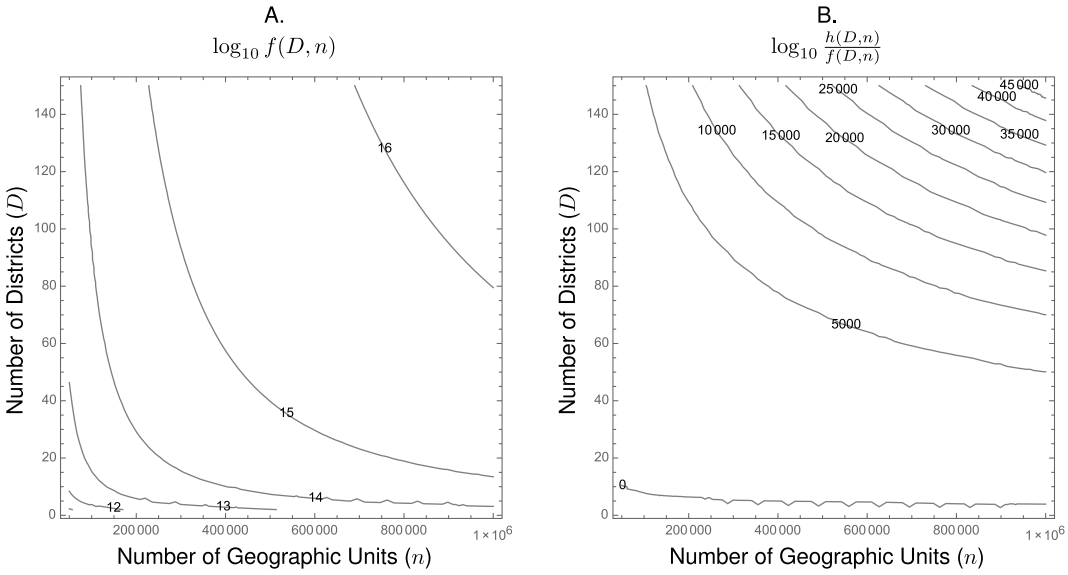

Figure 3. (a) The manner in which the algorithm we propose scales with the number of districts (D) and the number of geographic units (n), and (b) a comparison of algorithmic complexity of our proposed algorithm to the algorithm proposed by Cirincione, Darling, and O'Rourke (2000). The x-axis in both graphs represents the number of geographic units (n) that the algorithm takes as an input, and the y-axis represents the number of districts (D) that the algorithm creates from those geographic units. The lines represent orders of magnitude in difference in asymptotic runtime at corresponding values of D and n.

roughly 300,000 geographic units, and it would take 1.0 seconds to draw a map of 100 districts from 850,000 geographic units.

The computational efficiency of the algorithm we propose becomes particularly important when compared to other algorithms devised to address the same redistricting problem. For example, take a commonly used alternative redistricting algorithm proposed by Cirincione, Darling, and O'Rourke (2000). In Appendix B we show that their algorithm scales according to the following function.

$$h(n, D) = \frac{n}{\prod_1^{D-1} p(n, i)}, \tag{2}$$

where $p(n, i)$ is the probability that the next iteration will successfully yield a contiguous district.[21]

The differences between (1) and (2) may not be immediately apparent, so we represent ratio of expected runtimes for values of n and D in Figure 3(b). Here, the contour lines represent the difference in runtime (in orders of magnitude) between our proposed algorithm and this commonly used alternative.[22] The contour line labeled 0 are the values of n and D for which we expect no difference in runtime between the two algorithms. From Figure 3(b), it is clear that our proposed algorithm outperforms the alternative algorithm for all redistricting problems except those for which the n or D is very small.

21 Since the Cirincione, Darling, and O'Rourke (2000) algorithm draws districts iteratively, the function $h(n, D)$ also includes a probability function $p(n, i)$ that the next iteration will successfully yield a contiguous district. That probability decreases with each district the algorithm draws. For illustrative purposes we assume that the next district drawn by the algorithm will almost certainly be contiguous ($p(n, i) = 0.99999$).

22 We are interested in the asymptotic properties of the algorithm in this type of analysis, constants drop out of the assessment of complexity. Hence, if the algorithm discards 90% of its maps or 99% it has the same algorithmic complexity. By contrast, the rate of failure to find a valid map is proportional to the number of districts or vertices to which we might apply the Cirincione, Darling, and O'Rourke (2000) algorithm. In other words, the number of maps that we discard is constant in the algorithm we propose and is independent of the complexity of the redistricting problem, but the number of maps that would need to be discarded using Cirincione, Darling, and O'Rourke (2000) is a function of the complexity of the redistricting problem.

The difference between the two algorithms is stark when either is applied to jurisdictions with larger n and more D. For example, consider the redistricting problem for New York's State Assembly. There are 150 Assembly districts in New York ($D = 150$).[23] If an analyst were to use census blocks as the geographic unit from which he or she drew districts, then $n = 350,169$ census blocks in New York in the 2010 census. Thus, we expect that the runtime for our algorithm would be between 10,000 and 15,000 *orders of magnitude* faster than the processing time of the Cirincione, Darling, and O'Rourke (2000) algorithm when using census block data to draw state assembly districts in New York. Perhaps more concretely, suppose our proposed algorithm could draw a map of 150 districts from New Yorks 350,169 census blocks in 1.0 second. Asymptotic analysis indicates that Cirincione, Darling, and O'Rourke's algorithm would take longer than 317×10^{9990} years to draw one map of New York's 150 districts from its 350,169 census blocks.

4 Demonstration

Texas, Virginia, and Mississippi provide an excellent context in which to apply the algorithm we propose. All three states have significant minority populations, and all three states are located in the South. Prior to the Supreme Court ruling that invalidated Section 5 of the Voting Rights Act, all three states had to submit their redistricting plans to the Justice Department for preclearance.[24] Following the Court's decision in *Thornburg v. Gingles*,[25] which encouraged the creation of districts in which minority population made up more than half the population, all three states took steps to create majority-minority districts. We consider several types of majority-minority districts. For the purposes of our analysis, we consider majority-minority districts in Mississippi and Virginia to be districts in which the US Census data categorizes 50% or more of the residents as "Black or African American alone". In Texas, we consider districts in which the census categorizes 50% or more of the residents as "Hispanic or Latino" to be majority-minority districts.[26] Given the large minority populations in Texas (12.5% Black, 38.6% Hispanic), Virginia (19.7% Black, 8.9% Hispanic), and Mississippi (37.5% Black, 3% Hispanic) each state is a prime candidate for producing at least one majority-minority district.

4.1 Summary statistics

The goal of our analysis is to develop an approximate expectation of the distribution of minority residents between a state's districts if legislative mapmakers only consider contiguity and population. In practice, the geographic distribution of minority voters and the political and legal contexts in which the congressional districts are devised influences the number of majority-minority districts within a state. Since our proposed algorithm only considers contiguity of geographical units and the total population of those units, it is neutral with regards to the politics or race of the individuals living in a given geographical unit. Consequently, the algorithm is neutral with regards to the output of the process. Since the algorithm generates each map by a different neutral and random process, the probability that any two maps are exactly alike is extremely small. The distribution of maps represents the counterfactual against which we can compare the actual map generated by the states we analyze. The distribution also allows us to characterize the approximate probability by which the states' maps could have emerged from a neutral process. Of course, mapmakers in each state strove to comply with the Court's

23 In the United States, state legislatures pose the most complex redistricting problem because of the large number of districts and the large number of geographic units mapmakers could use to draw maps. State legislatures range in size from 40 (Alaska's lower chamber) to 203 (Pennsylvania's lower chamber). The number of census blocks, the smallest geographic unit upon which mapmakers can base districts, ranged from 24,114 in Delaware to 914,231 in Texas in the 2010 census.

24 *Shelby County v. Holder*, 570 U.S. 2 (2013).

25 478 U.S. 30 (1986).

26 Alternatively, majority-minority districts may be districts in which more than 50% of the population of a district is non-Anglo, any resident not categorized as "Not Hispanic or Latino: White alone". The analysis we conduct here yields similar results regardless of how we define majority-minority districts.

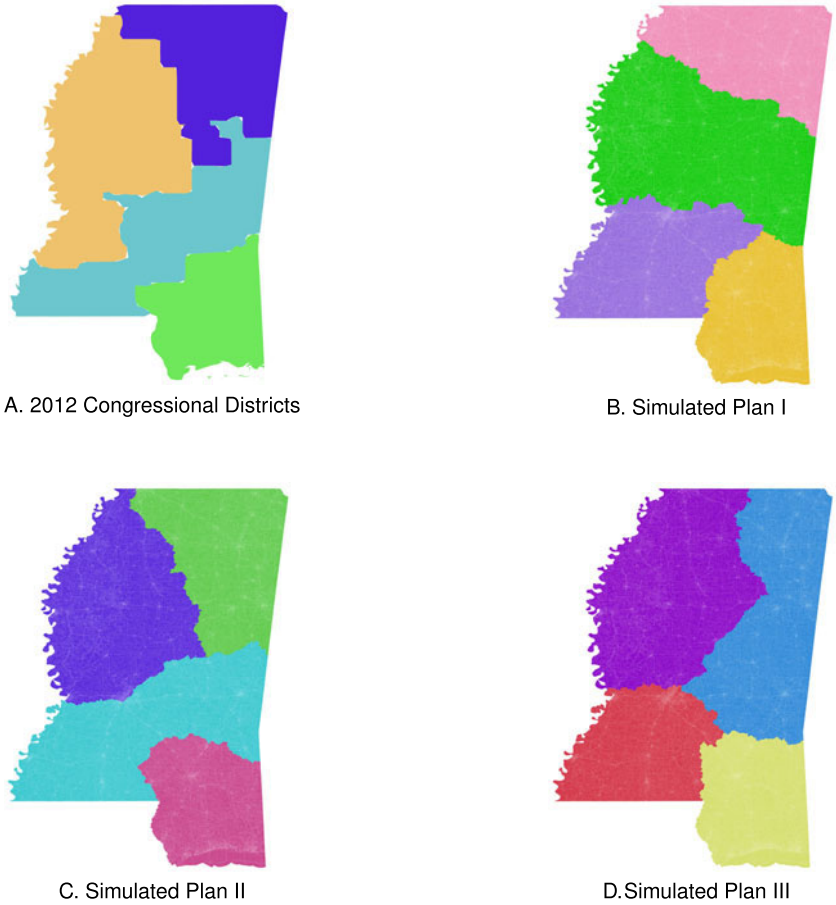

Figure 4. Three neutral maps of Mississippi exhibit the types of variation typical of our neutral mapping algorithm.

encouragement to generate majority-minority districts. Whether out of zeal to fulfill the courts instructions or out of some other motivation, each state in our analysis deviates in significant ways from a neutral process of drawing districts.

To draw our neutral maps, we use data collected by the United States Census and made available by the Minnesota Population Center's *National Historical Geographic Information System* (NHGIS) project. We use census block-level geospatial and demographic data. Census blocks are the smallest geographic unit for which we have information regarding population and reliable data regarding residents' demography. We generated 10,000 maps of four, eleven, and thirty-six congressional districts for Mississippi, Virginia, and Texas, respectively (Magleby and Mosesson 2017). Representations of the kinds of districts the algorithm produces are visible in Figures 4, 5, and 6. In Figures 4(a), 5(a), and 6(a), we include the actual map of districts that each state used for the 2012 congressional elections. As is clear from Figures 4, 5, and 6, the algorithm generated relatively compact districts. Each was balanced to within 1.5% of the ideal for that state. The examples of neutral partitions of Mississippi included in Figure 4 show how the maps generated by the algorithm diverge from the actual map used for the 2012 elections. Just as the actual plan does, the maps represented in Figures 4(b) and 4(c) divide Jackson, one of the only urban areas in Mississippi, across two districts. By contrast the map in Figure 4(d) keeps Jackson whole. Each example is a valid partition of Mississippi into four contiguous districts with roughly equal population.

The algorithm also generated realistic districts in the larger state Virginia that show considerable variance in the composition of districts. Just like the actual map, the maps in

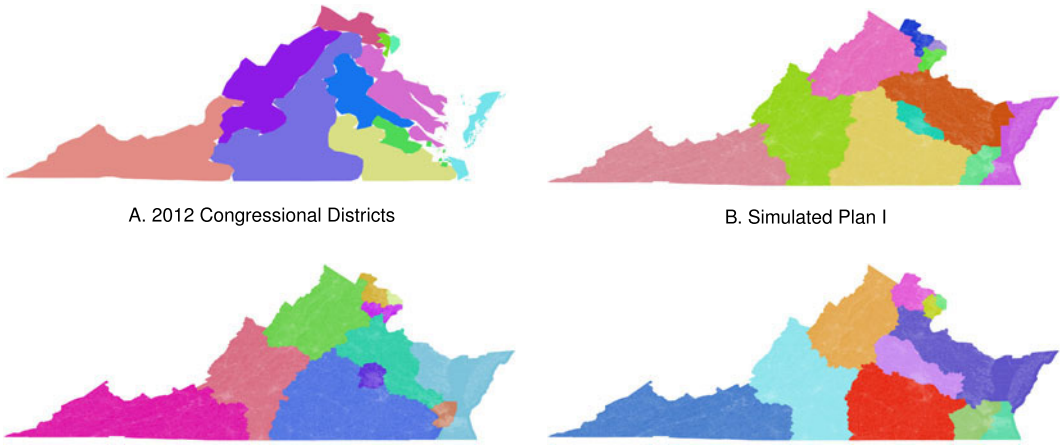

Figure 5. Three neutral maps of Virginia exhibit the types of variation typical of our neutral mapping algorithm.

Figures 5(b)–(d) show multiple districts in the densely populated areas around Washington, DC in Northern Virginia, but each of the example maps handled the other densely populated areas in and around Richmond and Hampton differently. Again, our proposed algorithm only considers population and contiguity, so these example maps represent realistic, balanced, and neutral partitions of Virginia into eleven congressional districts.

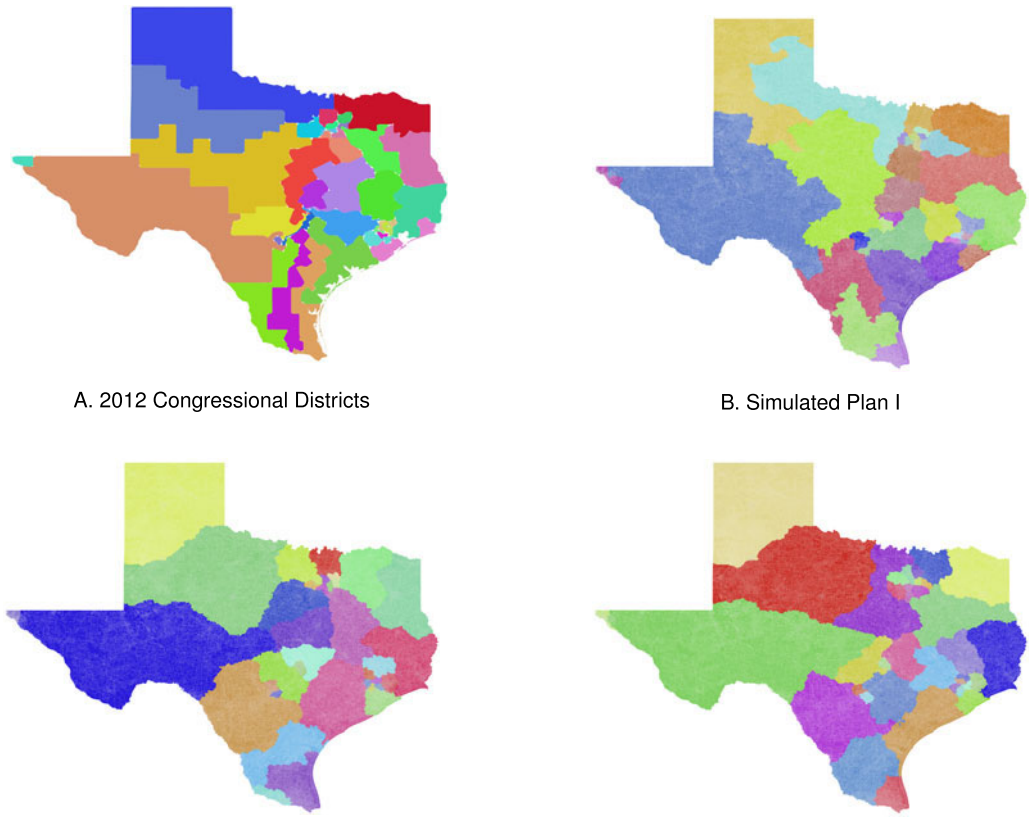

Figure 6. Three neutral maps of Texas exhibit the types of variation typical of our neutral mapping algorithm.

Table 2. A comparison of the actual number of majority-minority districts in Mississippi, Virginia, and Texas and estimates of the number of majority-minority districts in 10,000 simulated maps of congressional districts. Data on district and block-level demographics taken from the US Census data provided by NHGIS.

	Total districts	No. of majority-minority dists. in 2012 map	Estimated likelihood of 2012 maj-min dists.
Mississippi[a]	4	1	$p < 0.3073$
Virginia[a]	11	2	$p < 0.0001$
Texas[b]	36	9	$p < 0.1589$

a Proportion of residents categorized as "Black or African American alone" greater than 0.5.

b Proportion of residents categorized as "Hispanic or Latino" greater than 0.5.

Finally, the algorithm generated a variety of realistic maps of Texas's congressional districts. The actual map, included here as Figure 6(a), has several districts in the densely populated Dallas, Houston, and Austin areas. Figures 6(b)–(d), all have a geographically small districts in these more densely populated areas. Just like the actual map, each simulated map included here has a smaller district in the far west El Paso area, and geographically expansive districts covering sparsely populated West Texas and the Texas panhandle regions. One contrast between the simulated maps and the map in Figure 6(a) is the way the simulated maps deal with the highly hispanic region of South Texas. Texas's 2012 map divides this heavily Latino area into three districts that run from the border with Mexico to Austin. The actual districts in this region are heavily hispanic leaving neighboring districts with fewer latino voters. The algorithm drew the examples in Figures 6(b)–(d) without reference to race and divide up the heavily hispanic population of South Texas neutrally. For some of these neutral maps, the result may be a more even distribution of hispanic residents across more than three districts in South Texas.

4.2 State performance

The algorithm retrieves an approximate null distribution of the number of majority-minority districts in each of the states we examine. That is, it allows us to create an expectation of the number of majority-minority districts in each state had mapmakers only considered population and the contiguity of geographic units that make up the district. Using census data on the racial composition of each state's actual congressional districts, we characterize the degree to which the racial composition of each state's actual districts differ from the neutral counterfactual retrieved by the algorithm. In the most recent round of redistricting following the 2010 census, blacks constituted a majority of the residents in one out of four congressional districts in Mississippi, two out of eleven congressional districts in Virginia. Hispanic residents were a majority in nine out of thirty-six districts in Texas.

The data summarized in Table 2 provide an expectation of the number of majority-minority seats each state should produce. Those data indicate the probability that the actual number of majority-minority districts in Mississippi, Virginia, and Texas arose from a neutral process. In Mississippi, fewer than 1/3 of the simulated maps produced one majority black seat. We may interpret the frequency of majority-minority seats in our simulated maps of Mississippi, 3,073/10,000 produced one majority black seat, to mean that there is less than an approximately 30.73% chance that the number of majority-minority districts we actually observe in Mississippi arose from a neutral process. In Virginia, none of the 10,000 neutrally drawn maps produced a majority black district. Thus we can say that there is less than an approximately 0.001% chance that the actual map of Virginia's congressional district arose from a neutral process. Finally, 1,589/10,000 of our neutrally drawn maps of Texas produced nine or more majority hispanic districts. Thus, we may say that there is less than an approximately 16% chance that the actual map of Texas's congressional districts arose from a neutral process. In short, we cannot

confidently reject the notion that Mississippi ($p < 0.3073$) and Texas ($p < 0.1589$) produced the number of majority-minority districts using a neutral process. On the other hand, it is extremely unlikely that Virginia ($p < 0.0001$) arrived at two majority-minority districts by way of a neutral process.[27]

5 Possible Extensions

The algorithm we propose in this article neutrally generates a set of districts without indication of bias that are contiguous, balanced, and relatively compact; however, analysts may wish to apply additional constraints to the process of drawing legislative districts. For example, some jurisdictions seek to keep whole existing political units like municipalities or counties. In addition, mapmakers may seek to generate a particular number of majority-minority districts. While we forgo the imposition of additional constraints in this article, it is worth noting briefly that, with modification, the graph partitioning approach we propose could incorporate additional constraints.

Applying additional constraints to the sample drawn by the algorithm requires; first, that we allow for heterogeneity in the relationship between geographic units. Recall, we weighted geographic units (vertices) according to the number of people that reside in the unit. In the analysis we present here, all the relationships between units (edges) are weighted equally, but that need not be so. Our approach could be altered to weigh the relationship between units, and those weights could vary according to any number of factors of interest. For instance, we might apply higher weights to edges between vertices that should fall in the same district (e.g. units in the same municipality). Likewise, we could apply a lower weight to edges between units that need not fall in the same district (units *not* in the same municipality).

Second, we could alter the algorithm to minimize the weight of edges it cuts to create districts. Fortunately, there already exist a host of graph partitioning algorithms developed by computer scientists that minimize the edge cut while maintaining the balance in the weight applied to vertices across partitions (for example, see Karypis and Kumar 1995, 1998). These algorithms could be modified to address the problem of dividing geographic units into legislative districts. Using this approach, we have begun the process of incorporating additional constraints into studies of the impact of maintaining communities of interest (the practice of keeping cities and counties whole in legislative maps) and majority-minority districts on legislative districts and policy outcomes.

6 Conclusion

In this article, we have proposed a computationally efficient algorithm that generates a set of districts that are contiguous, balanced, relatively compact, and which do not show the kinds of biases found in previous algorithms. We base our algorithm on one proposed by computer scientists to divide tasks weighted by computational complexity evenly between processors. In our application of the algorithm, geographic units are analogous to tasks and the population of the geographic units are the weights. The algorithm gains efficiency by first simplifying a map into a much simpler graph, partitioning the simpler version of the graph, and then projecting that partition back into the complex version of the map. In the last step of the process we apply the Kernighan–Lin algorithm, used by computer scientists to refine assignments of computational tasks to a computer's processor, to attain districts that are more balanced in terms of population.

27 Cirincione, Darling, and O'Rourke (2000) report a similar finding regarding South Carolina's 1990 congressional map. They use their algorithm to retrieve a null distribution; however, their algorithm partitioned relatively coarse block groups into six districts making the redistricting problem relatively simpler than any of the redistricting problems we present in the section.

As a demonstration of the algorithm's capabilities, we compare an approximate distribution of neutral districts to the actual congressional districts in Mississippi, Virginia, and Texas produced in the round of redistricting following the 2010 census. The algorithm generated 10,000 valid and distinct maps consisting of balanced and contiguous districts, from computationally intensive block-level data for each state. The patterns of minority concentration we observe in each of these states are inconsistent with patterns that might have arisen through a neutral process. In other words, what we observe appears to be a deliberate attempt by mapmakers in Mississippi, Virginia, and Texas to generate majority-minority seats. Absent such an effort, we would expect a set of districts in each state that would make it challenging for minority voters to select a candidate of their choosing.

The concentration of minority voters almost certainly reverberates through the politics of each of these states. In particular, by concentrating minority voters, who tend to support Democratic candidates, into a few districts, mapmakers also concentrate Democratic voters into a few districts. The consequence is that the remaining districts are almost certainly more Republican than we would expect if the process of redistricting followed more neutral patterns. We forgo an analysis of the partisan implications of these patterns of redistricting here, but we acknowledge that the method we have presented may be usefully applied in the evaluation of potential racial or partisan gerrymanders.

The purpose of this article is to demonstrate a method for drawing neutral legislative districts that is an advance over prior published algorithms; however, the analysis we present here allows us to make substantive conclusions about legislative districts in Mississippi, Virginia, and Texas. Our conclusions suggest at least two avenues of further research. First, we may explore the extent to which the inclusion of majority-minority districts allows for minority representation in other states. Determining the extent of packing in other jurisdictions would require analyzing more states than space permits in this article. Since our algorithm handles even the largest jurisdictions using the most complex data, analyzing any other jurisdictions would not be problematic. Second, we can use the algorithm to characterize the distribution of partisan voters in a null distribution. The null distribution retrieved by the algorithm also serves as a counterfactual against which we can compare patterns of partisanship in enacted maps. We intend to pursue questions related to the broader practice and implications of racial and partisan gerrymandering in future work.

Appendix A. Formal Presentation of Algorithm

Step 1 ("Coarsening")

We transform G_0 into a series of smaller graphs $G_1, G_2, G_3, \ldots, G_m$ where $|A_0| > |A_1| > |A_2| > |A_3| > \cdots > |A_m|$. The algorithm selects a vertex $u \in A_i$ with probability $1/n_i$ where $n_i = |A_i|$. If u has not been selected previously, then the algorithm *matches* u with $v \in V_i^u$ with probability $1/|V_i^u|$ where V_i^u is the set of unmatched vertices adjacent to u. If $\exists\ v \in V_i^u$, then the algorithm collapses u and v into a *multinode* $a_j \in A_{i+1}$. In G_{i+1}, the weight of the multinode, $w(a_j) = w(u) + w(v)$, and $E_{a_j} = E_u \cup E_v$. If $V_i^u = \emptyset$, then u remains unmatched. Unmatched vertices are copied over to G_{i+1}.

Step 2 ("Partitioning")

We compute a k-way partition P_m of graph G_m that divides V_m into k parts each containing $|A_m|/k$ vertices. The algorithm chooses a multinode $A_i \in A_m$ with probability $1/|A_m|$. It then chooses another block $u \in V_i^{A_i}$ with probability $1/|V_i^{A_i}|$ and combines the multinode into a district $P_m[v]$. If $w(u) + w(A_i) \geq 1/kW(A_m)$ or $w(u) + w(P_m[v]) \geq 1/kW(A_m)$, the algorithm stops adding multinodes to the district. If $w(u) + w(A_i) < 1/2W(A_m)$ or $w(u) + w(P_m[v]) < 1/2W(A_m)$, then the algorithm repeats step 2; however, it chooses $u \in V_i^{P_m[v]}$ in every iteration after the first.

Step 3 ("Uncoarsening")

By going through a set of intermediate partitions $P_{m-1}, P_{m-2}, \ldots, P_1, P_0$, the algorithm projects P_m of G_m back onto G_0. At every step of the uncoarsening process, the algorithm assigns $P_i[u] = P_{i+1}[v],\ \forall\ v \in V_i^u$.

Step 4 ("Refinement")

Consider a partition that has two parts v and u. For each $P_{m-1}, P_{m-1}, \ldots, P_0$ in the uncoarsening step, let v and u be two parts of P_i. The algorithm selects $v'_{i+1} \subset v_{i+1}$ and $u'_{i+1} \subset u_{i+1}$ where v'_{i+1} is contiguous with u_{i+1} and u'_{i+1} is contiguous with v_{i+1}. If $|w(v_{i+1}) - w(u_{i+1})| > |w(v_{i+1} \setminus v'_{i+1} \cup u'_{i+1}) - w(u_{i+1} \setminus u'_{i+1} \cup v'_{i+1})|$ then it sets $v_i = v_{i+1} \setminus v'_{i+1} \cup u'_{i+1}$ and $u_i = u_{i+1} \setminus u'_{i+1} \cup v'_{i+1}$, otherwise $v_i = v_{i+1}$ and $u_i = u_{i+1}$.

Step 5 ("Repeat")

If the resulting partition is a set of contiguous and balanced districts, we record the partition for later analysis. If not, the flawed partition is discarded and the algorithm restarts.

Appendix B. Algorithmic Complexity

B.1 Our proposed algorithm

Table B1. Asymptotic cost of our proposed algorithm.

Step	Asymptotic cost (inclusive to current step)	Assumptions
1. Coarsening	Constant	None
2. Partitioning	$n^2 \log n$	The process of assigning a node to a partition (multinode), is equivalent to sorting a list. Computer scientists have shown that sorting a list requires $n \log n$ operations. Since all nodes must be assigned to a partition, this process must occur n times. Thus, partitioning the graph requires $n^2 \log n$ operations.
3. Uncoarsening	Constant	None
4. Refinement	$n^2 \log nD \log D$	In order to achieve a balanced set of districts, the algorithm trades nodes between districts. For every node in either district a or district b, we find the best node with which it may switch with a neighboring district. This is equivalent to searching a sorted list, a $\log n$ operation. The algorithm then finds the set of switches that makes the partition more balanced, an $O(n)$ operation. The process of actually switching nodes is an $O(n)$ operation. This process is an inner operation that has a complexity of $O(n \log n + n + n)$ which reduces to $O(n \log n)$. The whole process repeats as long as the previous iteration achieves a more balanced set of partitions, this can happen up to n times. Thus, the complexity of refining two districts is $O(n^2 \log n)$. The algorithm repeats this process for every pair of adjacent districts in list of pairs ordered by imbalance, which has complexity $O(D \log D)$. Thus, the whole process has a complexity of $O(D \log D n^2 \log n)$ in the worst case.

B.2 Cirincione, Darling, and O'Rourke (2000)

Table B2. Asymptotic Cost of Cirincione, Darling, and O'Rourke (2000).

Step	Asymptotic cost (inclusive to current step)	Assumptions
1. From the set of unassigned geographic units, randomly select one unit.	Constant	
2. Identify all units that are unassigned and adjacent to the selected unit.	Constant	Maximum degree of a planar graph is bounded above by six, which is constant.
3. Randomly select one of the adjacent unassigned units and add it to the district.	Constant	
4. Repeat Steps 2 and 3 until the district reaches the predetermined population threshold.	n/D	n/D units must be added to a district.
5. Repeat Steps 1–4 until all units are assigned.	n	This must be done D times.
6. Continue until algorithm draws D districts or restart if process cannot draw a contiguous district with the required population.	$\frac{n}{\prod_1^{D-1} p(n,i)}$	Every time Steps 1–5 are run, there is a chance that some partitions will not be contiguous. If the algorithm is just taking a greedy walk through the graph, there is some probability function $p(n, i)$ that describes the probability that the ith district will not be contiguous. This leads the cost of the algorithm to be a power function of D.

References

Altman, Micah, Brian Amos, Michael P. McDonald, and Daniel A. Smith. 2015. Revealing preferences: Why gerrymanders are hard to prove, and what to do about it. http://dx.doi.org/10.2139/ssrn.2583528.

Altman, Micah, Jeff Gill, and Michael P. McDonald. 2004. *Numerical issues in statistical computing for the social scientist*, vol. 508. Hoboken, NJ: John Wiley & Sons.

Altman, Micah, and Michael P. McDonald. 2011. BARD: Better automated redistricting. *Journal of Statistical Software* 42(4):1–28.

Bozkaya, Burcin, Erhan Erkut, and Gilbert Laporte. 2003. A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research* 144(1):12–26.

Browdy, Michelle H. 1990. Simulated annealing: An improved computer model for political redistricting. *Yale Law & Policy Review* 8(1):163–179.

Chen, Jowei, and David Cottrell. 2016. Evaluating partisan gains from Congressional gerrymandering: Using computer simulations to estimate the effect of gerrymandering in the US House. *Electoral Studies* 44:329–340.

Chen, Jowei, and Jonathan Rodden. 2013. Unintentional gerrymandering: Political geography and electoral bias in legislatures. *Quarterly Journal of Political Science* 8(3):239–269.

Chou, Chung-I, and Sai-Ping Li. 2006. Taming the gerrymander? statistical physics approach to political districting problem. *Physica A: Statistical Mechanics and its Applications* 369(2):799–808.

Cirincione, Carmen, Thomas A. Darling, and Timothy G. O'Rourke. 2000. Assessing South Carolina's 1990s congressional districting. *Political Geography* 19(2):189–211.

Engstrom, Richard L., and John K. Wildgen. 1977. Pruning thorns from the thicket: An empirical test of the existence of racial gerrymandering. *Legislative Studies Quarterly* 2(4):465–479.

Fifield, Benjamin, Michael Higgins, Kosuke Imai, and Alexander Tarr. 2015a. A new automated redistricting simulator using Markov chain Monte Carlo. Working Paper, Princeton University, Princeton, NJ.

Fifield, Benjamin, Michael Higgins, Kosuke Imai, and Alexander Tarr. 2015b. Redist: Markov chain Monte Carlo methods for redistricting simulation. Available at the Comprehensive R Archive Network (CRAN), http://cran.r-project.org/package=redist.

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

Garfinkel, Robert S., and George L. Nemhauser. 1970. Optimal political districting by implicit enumeration techniques. *Management Science* 16(8):B-495.

Hendrickson, Bruce, and Robert Leland. 1995. A multi-level algorithm for partitioning graphs. In *Supercomputing, 1995 Proceedings of the IEEE/ACM SC95 Conference*. San Diego, CA: IEEE, pp. 28–28.

Johnson, Shawn. 2016a. Expert: Republicans 'virtually 100 percent' certain to retain majorities under GOP map. Wisconsin Public Radio. http://www.wpr.org/expert-republicans-virtually-100-percent-certain-retain-majorities-under-gop-map.

Johnson, Shawn. 2016b. Wisconsin gerrymandering trial wraps up. Wisconsin Public Radio. http://www.wpr.org/expert-republicans-virtually-100-percent-certain-retain-majorities-under-gop-map.

Karypis, George, and Vipin Kumar. 1995. Analysis of multilevel graph partitioning. In *Proceedings of the 1995 ACM/IEEE Conference on Supercomputing*. San Diego, CA: ACM, p. 29.

Karypis, George, and Vipin Kumar. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing* 20(1):359–392.

Kernighan, Brian W., and Shen Lin. 1970. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal* 49(2):291–307.

Magleby, Daniel B., and Daniel B. Mosesson. 2017 Replication data for: A new approach for developing neutral redistricting plans. Harvard Dataverse, V1. doi:10.7910/DVN/R9GKJA.

McCarty, Nolan, Keith T. Poole, and Howard Rosenthal. 2009. Does gerrymandering cause polarization? *American Journal of Political Science* 53(3):666–680.

McDonald, Michael D., and Robin E. Best. 2015. Unfair partisan gerrymanders in politics and law: A diagnostic applied to six cases. *Election Law Journal* 14(4):312–330.

Nagel, Stuart S. 1965. Simplified bipartisan computer redistricting. *Stanford Law Review* 17:863–899.

O'Loughlin, John, and Anne-Marie Taylor. 1982. Choices in redistricting and electoral outcomes: The case of Mobile, Alabama. *Political Geography Quarterly* 1(4):317–339.

Remmert, Hanna. 2016. *Blue waters supercomputer used to develop a standard for gerrymandering*. University of Illinois Press Release. http://www.ncsa.illinois.edu/news/story/blue_waters_supercomputer_used_to_develop_a_standard_for_partisan_gerrymand.

Tam Cho, Wendy K., and Yan Y. Liu. 2016. Toward a talismanic redistricting tool: A computational method for identifying extreme redistricting plans. *Election Law Journal* 15(4):351–366.

Vickrey, William. 1961. On the prevention of gerrymandering. *Political Science Quarterly* 76(1):105–110.

Wang, Samuel S.-H. 2016. Three tests for practical evaluation of partisan gerrymandering. *Stanford Law Review* 68:1263.

Weaver, James B., and Sidney W. Hess. 1963. A procedure for nonpartisan districting: Development of computer techniques. *Yale Law Journal* 73(2):288–308.