A New Automated Redistricting Simulator Using Markov Chain Monte Carlo*

Benjamin Fifield† Michael Higgins‡ Kosuke Imai§ Alexander Tarr¶

March 15, 2017

Abstract

Legislative redistricting is a critical element of representative democracy. A number of political scientists have used simulation methods to sample redistricting plans under various constraints in order to assess their impacts on partisanship and other aspects of representation. However, while many optimization algorithms have been proposed, surprisingly few simulation methods exist in the literature. Furthermore, the standard algorithm has no theoretical justification, scales poorly, and is unable to incorporate fundamental substantive constraints required by redistricting processes in the real world. To fill this gap, we formulate redistricting as a graph-cut problem and propose a new automated redistricting simulator based on Markov chain Monte Carlo. We show how this algorithm can incorporate various constraints including equal population, geographical compactness, and status quo biases. Finally, we apply simulated and parallel tempering to improve the mixing of the resulting Markov chain. Through a small-scale validation study, we show that the proposed algorithm outperforms the standard algorithm in terms of both speed and ability to approximate a target distribution. We also apply the proposed methodology to the data from New Hampshire and Pennsylvania and show that our algorithm scales and yields new substantive insights. The open-source software is available for implementing the proposed methodology.

Keywords: gerrymandering, graph cuts, Metropolis-Hastings algorithm, simulated tempering, parallel tempering, Swendsen-Wang algorithm

*We thank Jowei Chen, Jacob Montgomery, and seminar participants at Chicago Booth, Dartmouth, Duke, and Microsoft Research for useful comments and suggestions. We thank James Lo, Jonathan Olmsted, and Radhika Saksena for their advice on computation. The open-source R package `redist` for implementing the proposed methodology is available as Fifield *et al.* (2015)

†Ph.D. Candidate, Department of Politics, Princeton University, Princeton NJ 08544. Email: bfifield@princeton.edu, URL: `http://www.benfifield.com`

‡Assistant Professor, Department of Statistics, Kansas State University, Manhattan KS 66506. Email: mikehiggins@ksu.edu, URL: `http://www-personal.ksu.edu/~mikehiggins`

§Professor, Department of Politics and Center for Statistics and Machine Learning, Princeton University, Princeton NJ 08544. Phone: 609–258–6601, Email: kimai@princeton.edu, URL: `http://imai.princeton.edu`

¶Ph.D. Candidate, Department of Electrical Engineering, Princeton University, Princeton NJ 08544. Email: atarr@princeton.edu

1 Introduction

Legislative redistricting is a critical element of representative democracy. Previous studies have found that redistricting influences turnout and representation (e.g., Abramowitz, 1983; Gelman and King, 1994; Ansolabehere *et al.*, 2000; McCarty *et al.*, 2009; Barreto *et al.*, 2004). From a public policy perspective, redistricting is potentially subject to partisan gerrymandering. After the controversial 2003 redistricting in Texas, for example, Republicans won 21 congressional seats in the 2004 election (Democrats won 11) whereas they had only 15 seats in 2002 (Democrats won 17). To address this concern, numerous remedies, including geographical compactness and partisan symmetry requirements, have been proposed (see Grofman and King, 2007; Fryer and Holden, 2011, and references therein).

The development of automated redistricting algorithms, which is the goal of this paper, began in the 1960s. Vickrey (1961) argued that such an "automatic and impersonal procedure" can eliminate gerrymandering (p. 110). After *Baker v. Carr* (1962) where the Supreme Court ruled that federal courts may review the constitutionality of state legislative apportionment, citizens, policy makers, and scholars became interested in redistricting. Weaver and Hess (1963) and Nagel (1965) were among the earliest attempts to develop automated redistricting algorithms (see also Hess *et al.*, 1965). Since then, a large number of methods have been developed to find an *optimal* redistricting plan for a given set of criteria (e.g., Garfinkel and Nemhauser, 1970; Browdy, 1990; Bozkaya *et al.*, 2003; Chou and Li, 2006; Fryer and Holden, 2011; Tam Cho and Liu, 2016). These optimization methods serve as useful tools when drawing district boundaries (see Altman *et al.*, 2005, for an overview).

However, the main interest of substantive scholars has been to characterize the *distribution* of possible redistricting plans under various criteria for detecting instances of gerrymandering and understanding the causes and consequences of redistricting (e.g., Engstrom and Wildgen, 1977; O'Loughlin, 1982; Cirincione *et al.*, 2000; McCarty *et al.*, 2009; Chen and Rodden, 2013). In 39 of

the 50 U.S. states, for example, state politicians control the redistricting process and approve redistricting plans through standard statutory means. Therefore, an important institutional and public policy question is how to effectively constrain these politicians through means such as compactness requirements (e.g., Niemi *et al.*, 1990), in order to prevent the manipulation of redistricting for partisan ends. Simulation methods allow substantive scholars to answer these questions by approximating distributions of possible electoral outcomes under various institutional constraints.

Yet, surprisingly few simulation algorithms exist in the methodological literature. In fact, most, if not all, of these existing studies use essentially the same Monte Carlo simulation algorithm where a geographical unit is randomly selected as a "seed" for each district and then neighboring units are added to contiguously grow this district until it reaches the pre-specified population threshold (e.g., Cirincione *et al.*, 2000; Chen and Rodden, 2013). Unfortunately, no theoretical justification is given for these existing simulation algorithms, and some of them are best described as ad-hoc. Moreover, they scale poorly and are unable to incorporate some of the constraints required by redistricting processes in the real world. A commonly used algorithm of this type is proposed by Cirincione *et al.* (2000) and implemented by Altman and McDonald (2011) in their open-source software. We hope to improve this state of the methodological literature.

To fulfill this methodological gap, in Section 2, we propose a new automated redistricting simulator using Markov chain Monte Carlo (MCMC). We formulate the task of drawing districting boundaries as the problem of graph-cuts, i.e., partitioning an adjacency graph into several connected subgraphs. We then adopt a version of the Swendsen-Wang algorithm to sample contiguous districts (Swendsen and Wang, 1987; Barbu and Zhu, 2005). We further extend this basic algorithm to incorporate various constraints commonly imposed on redistricting plans, including equal population requirements and geographical compactness. Finally, we apply simulated and parallel tempering to improve the mixing of the resulting Markov chain (Marinari and Parisi, 1992; Geyer and Thompson, 1995). Therefore, unlike the existing algorithms, the proposed algorithms are designed to yield a representative sample of redistricting plans under various constraints. The

open-source software, an R package `redist`, is available for implementing the proposed methodology (Fifield *et al.*, 2015).

In Section 3, we conduct a small-scale validation study where all possible redistricting plans under various constraints can be enumerated in a reasonable amount of time. We show that the proposed algorithms successfully approximate this true population distribution while the standard algorithm fails even in this small-scale redistricting problem. We then conduct empirical studies in realistic settings using redistricting and U.S. Census data from New Hampshire and Pennsylvania. The former is the smallest redistricting example with only two states while the latter is a more realistic application. In these applications, the computation of the true population distribution is not feasible and so we evaluate the empirical performance of the proposed algorithms by examining several standard diagnostics of MCMC algorithms. Our partisan bias analysis shows that the algorithm can be used to make small changes to Pennsylvania's congressional districts in 2008 that can nearly eliminate partisan bias. In both simulation and empirical studies, we show that the proposed algorithm runs much faster and is able to incorporate various constraints. Lastly, Section 4 gives concluding remarks.

2 The Proposed Methodology

In this section, we describe the proposed methodology. We begin by formulating redistricting as a graph-cut problem. We then propose a Markov chain Monte Carlo algorithm to uniformly sample redistricting plans with n contiguous districts. Next, we show how to incorporate various constraints such as equal population and geographical compactness. Finally, we improve the mixing of the MCMC algorithm by applying simulated and parallel tempering. A brief comparison with the existing algorithms is also given.

2.1 Redistricting as a Graph-cut Problem

Consider a typical redistricting problem where a state consisting of m geographical units (e.g., census blocks or voting precincts) must be divided into n contiguous districts. We formulate this redistricting problem as that of graph-cut where an adjacency graph is partitioned into a set of connected subgraphs (Altman, 1997; Mehrotra *et al.*, 1998). Formally, let $G = \{V, E\}$ represent an adjacency graph where $V = \{\{1\}, \{2\}, \ldots, \{m\}\}$ is the set of nodes (i.e., geographical units of redistricting) to be partitioned and E is the set of edges connecting neighboring nodes. This means that if two units, $\{i\}$ and $\{j\}$, are contiguous, there is an edge between their corresponding nodes on the graph, $(i, j) \in E$.

Given this setup, redistricting can be seen equivalent to the problem of partitioning an adjacency graph G. Formally, we partition the set of nodes V into n blocks, $\mathbf{v} = \{V_1, V_2, \ldots, V_n\}$ where each block is a non-empty subset of V, and every node in V belongs to one and only one block, i.e., $V_k \cap V_\ell = \emptyset$ and $\bigcup_{k=1}^{n} V_k = V$. Such a partition $\mathbf{v}$ generates an adjacency subgraph $G(\mathbf{v}) = (V, E(\mathbf{v}))$ where $E(\mathbf{v}) \subseteq E$. Specifically, an edge (i, j) belongs to $E(\mathbf{v})$ if and only if $(i, j) \in E$ and nodes $\{i\}$ and $\{j\}$ are contained in the same block of the partition, i.e., $\{i\}, \{j\} \in V_k$. Because $E(\mathbf{v})$ is obtained by removing some edges from E or "cutting" them, redistricting represents a graph cut problem. Finally, since each resulting district must be contiguous, we require each block of the partition to be connected—for any two nodes $\{i\}$ and $\{j\}$ in a block $V_k \in \mathbf{v}$, there exists a path of edges within V_k that joins these two nodes, or more formally, there exists a set of nodes $\{\{i\} = \{i_0\}, \{i_1\}, \{i_2\}, \ldots, \{i_{m'-1}\}, \{i_{m'}\} = \{j\}\} \subset V_k$ such that, for all $\ell \in \{1, \ldots, m'\}$, $(i_{\ell-1}, i_\ell) \in E(\mathbf{v})$. A partition comprised of n connected blocks is called valid. Let $\Omega(G, n)$ denote the set of all valid partitions.

Figure 1 presents two illustrative examples used in our validation study in Section 3.1. These examples are taken from actual Florida precinct data in an attempt to create realistic, albeit small, examples. A state is represented by an adjacency graph where nodes are geographical units

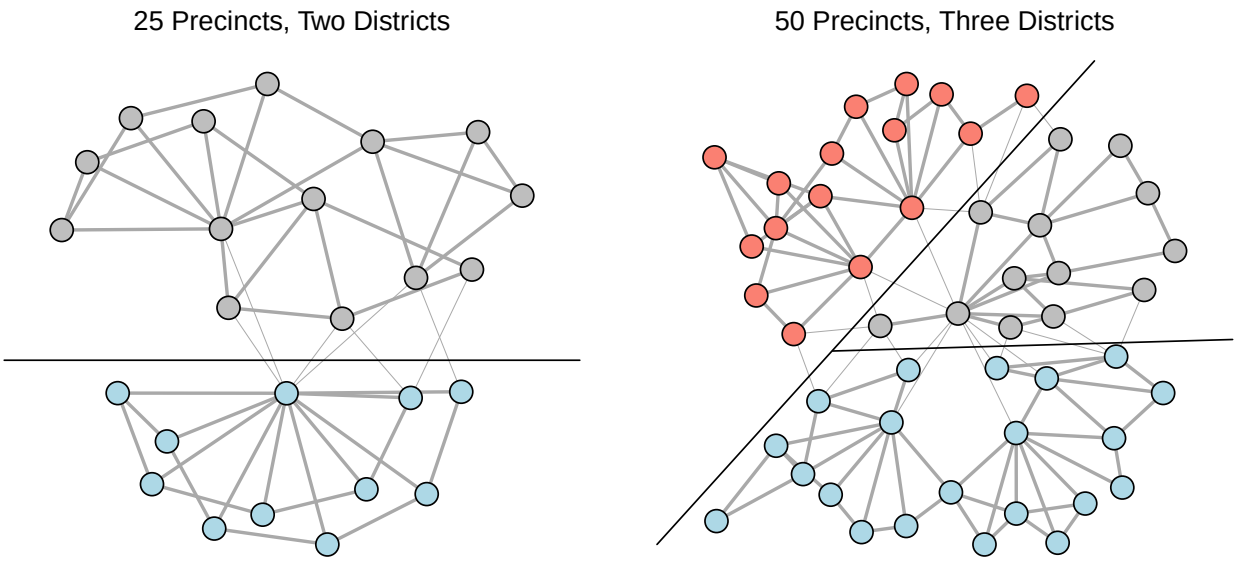

Figure 1: Redistricting as a Graph-cut Problem. A state is represented by an adjacency graph where nodes are geographical units and edges between two nodes imply their contiguity. Under this setting, redistricting is equivalent to removing or cutting some edges (light grey) to form connected subgraphs, which correspond to districts. Different districts are represented by different colors. Two illustrative examples, used in our validation study in Section 3.1, are given here.

and edges between two nodes imply their contiguity. The figure demonstrates that redistricting a state into n districts is equivalent to removing some edges of an adjacency graph (light grey) and forming n connected subgraphs.

2.2 The Basic Algorithm for Sampling Contiguous Districts

We propose a new automated simulator to uniformly sample valid redistricting plans with n contiguous districts. The contiguity of valid partitions dramatically increases the difficulty of developing such an algorithm. Intuitive methods for constructing partitions at random – e.g. randomly assigning precincts to districts – have a minuscule chance of yielding contiguous districts, and enumerating all partitions with contiguous districts is too large of a problem to be tractable in realistic redistricting settings. For more discussion, see Section 3.1.

Our MCMC algorithm is designed to obtain a dependent but representative sample from the uniform distribution of valid redistricting plans. In particular, we modify and extend SWC-1 of Barbu and Zhu (2005), which combines the Swendsen-Wang algorithm (Swendsen and Wang, 1987) with a Metropolis-Hastings step (Metropolis *et al.*, 1953; Hastings, 1970). This algorithm begins

with a valid partition $\mathbf{v}_0$ (e.g., an actual redistricting plan adopted by the state) and transitions from a valid partition $\mathbf{v}_{t-1}$ to another partition $\mathbf{v}_t$ at each iteration t. Here, we describe the basic algorithm for sampling contiguous districts. Later in the paper, we extend this basic algorithm in several important ways so that common constraints imposed on redistricting can be incorporated and so that the algorithm can achieve reasonable mixing for states with a larger number of districts.

Figure 2 illustrates our algorithm using the 50 precinct example with 3 districts given in the right panel of Figure 1. Our algorithm begins by randomly "turning on" edges in $E(\mathbf{v}_{t-1})$; each edge is turned on with probability q. In the left upper plot of Figure 2, the edges that are turned on are indicated with darker grey. Next, we identify components that are connected through these "turned-on" edges and are on the boundaries of districts in $\mathbf{v}_{t-1}$. Each such connected component is indicated by a dotted polygon in the right upper plot. Third, among these, a subset of non-adjacent connected components are randomly selected as shown in the left lower plot (two in this case). These connected components are reassigned to adjacent districts to create a candidate partition. Finally, the acceptance probability is computed based on two kinds of edges from each of selected connected components, which are highlighted in the left lower plot: (1) "turned-on" edges, and (2) "turned-off" edges that are connected to adjacent districts. We accept or reject the candidate partition based on this probability.

Our algorithm guarantees that its stationary distribution is equal to the uniform distribution of all valid partitions, thereby yielding a uniformly sampled sequence of redistricting plans with contiguous districts. We now formally describe this algorithm.

Algorithm 1 (Sampling Contiguous Redistricting Plans) *We initialize the algorithm by obtaining a valid partition* $\mathbf{v}_0 = \{V_{10}, V_{20}, \ldots, V_{n0}\}$*, where each block* V_{k0} *is connected in the graph, and then repeat the following steps at each iteration* t*,*

Step 1 ("Turn on" edges): *From the partition* $\mathbf{v}_{t-1} = \{V_{1,t-1}, V_{2,t-1}, \ldots, V_{n,t-1}\}$*, obtain the adjacency graph* $G(\mathbf{v}_{t-1}) = (V, E(\mathbf{v}_{t-1}))$ *with*

$$E(\mathbf{v}_{t-1}) = \{(i,j) \in E : \exists\, V_{k,t-1} \in \mathbf{v}_{t-1} \; s.t. \; i, j \in V_{k,t-1}\}. \tag{1}$$

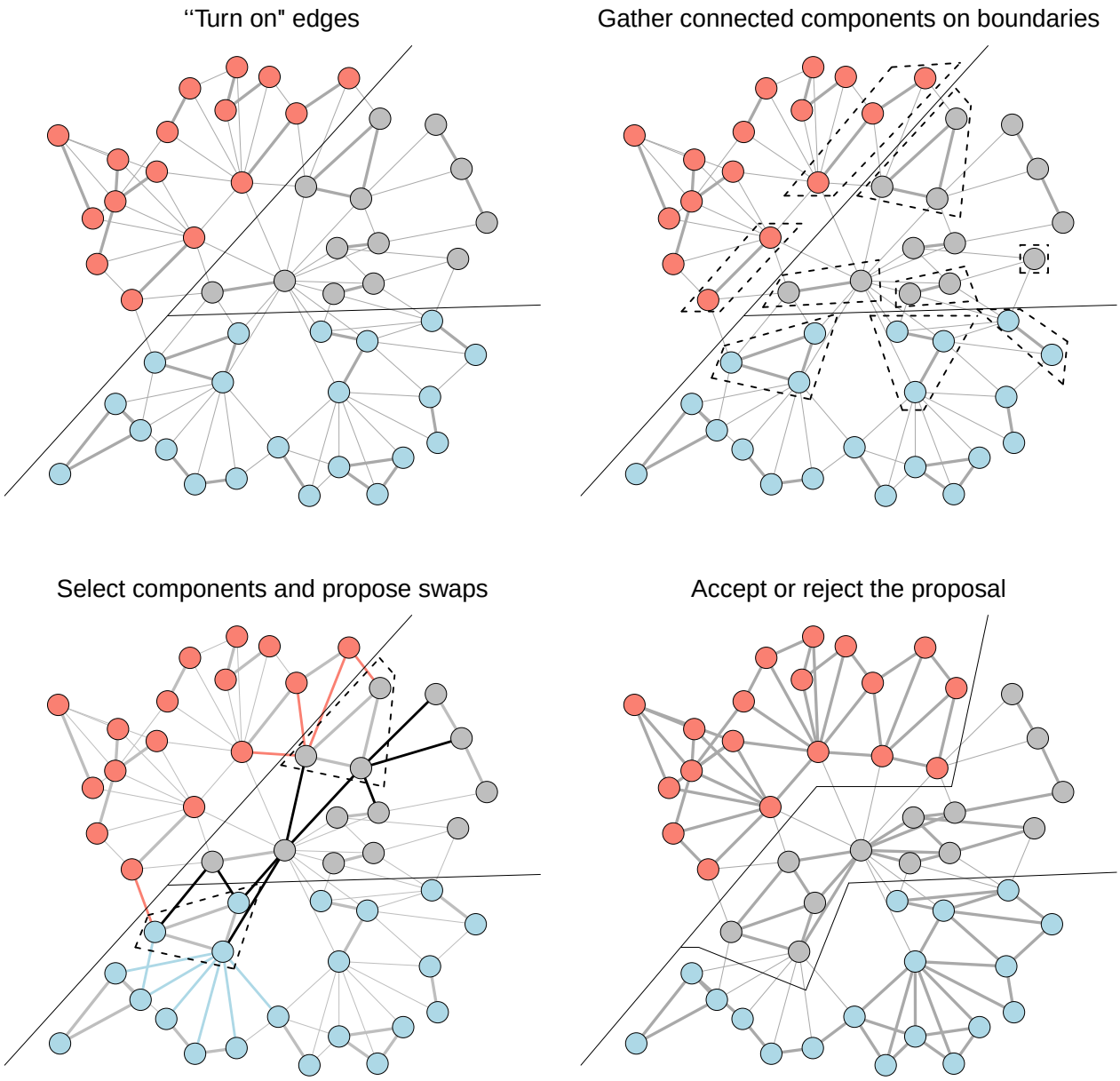

Figure 2: The Basic Algorithm for Sampling Contiguous Districts. The plots illustrate the proposed algorithm (Algorithm 1) using the 50 precinct data given in the right panel of Figure 1. First, in the left upper plot, each edge other than those which are cut in Figure 1 is “turned on” (dark grey) independently with certain probability. Second, in the right upper plot, connected components on the boundaries are identified (dashed polygons). Third, in the left lower plot, a certain number of non-adjacent connected components on boundaries are randomly selected (dashed polygons) and the acceptance ratio is calculated by counting certain edges (colored edges). Finally, in the right lower plot, the proposed swap is accepted using the Metropolis-Hastings ratio.

Form the edge set $E_{on}(\mathbf{v}_{t-1}) \subset E(\mathbf{v}_{t-1})$ where each edge $e \in E(\mathbf{v}_{t-1})$ is independently added to $E_{on}(\mathbf{v}_{t-1})$ with probability q.

Step 2 (Gather connected components): *Find all components that are connected within $E_{on}(\mathbf{v}_{t-1})$. Let* **CP** *denote this set of connected components.*

Step 3 (Select non-adjacent connected components along boundaries): *Randomly select a set of non-adjacent connected components $\mathbf{V_{CP}}$ from* **CP** *with $|\mathbf{V_{CP}}| = R \geq 1$ such that each component $V_{CP} \in \mathbf{V_{CP}}$ is a strict subset of a block in $\mathbf{v}_{t-1}$ and also lies along a partition boundary of $\mathbf{v}_{t-1}$. That is, for all $V_{CP} \in \mathbf{V_{CP}}$, there exists $k \in \{1, 2, \ldots, n\}$ such that $V_{CP} \subset V_{k,t-1}$ and $V_{CP} \cap V_{k,t-1} = \emptyset$ and $(i, j) \in E$ for some $\{i\} \in V_{CP}$ and $\{j\} \in V_{k,t-1}$.*

Step 4 (Propose swaps): *Initialize $\mathbf{v}_t^* = (V_{1t}^*, V_{2t}^*, \ldots, V_{nt}^*) = (V_{1t}, V_{2t}, \ldots, V_{nt})$. For each $V_{CP} \in \mathbf{V_{CP}}$, perform the following procedure:*

(a) Find which block $V_{kt} \in \mathbf{v}_t$ contains V_{CP}.

(b) Randomly assign V_{CP} to a neighboring block in $\mathbf{v}_t^*$. That is, define $\#adj(V_{CP}, \mathbf{v}_t)$ as the number of blocks in $\mathbf{v}_t$ that are adjacent to V_{CP} and propose to assign V_{CP} from block V_{kt} to block V_{k^*t} with probability

$$P(k^*|V_{CP}, \mathbf{v}_t) = \begin{cases} 1/\#adj(V_{CP}, \mathbf{v}_t), & V_{CP} \text{ is adjacent to } V_{k^*t}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

(c) Set $V_{k^*t}^* = V_{k^*t}^* \cup V_{CP}$ and set $V_{kt}^* = V_{kt}^* \setminus V_{CP}$. If $\mathbf{v}_t^*$ does not meet the requirements of a valid partition, return to Step 3. Observe that, after each proposed swap, $\mathbf{v}_t^*$ remains a connected set partition.

Step 5 (Accept or reject the proposal): *Set*

$$\mathbf{v}_t \;=\; \begin{cases} \mathbf{v}_t^*, & \text{with probability} \quad \alpha(\mathbf{v}_{t-1} \to \mathbf{v}_t^* | \mathbf{CP}), \\ \mathbf{v}_{t-1}, & \text{with probability} \quad 1 - \alpha(\mathbf{v}_{t-1} \to \mathbf{v}_t^* | \mathbf{CP}). \end{cases} \tag{3}$$

The acceptance probability is given by

$$\alpha(\mathbf{v}_{t-1} \to \mathbf{v}_t^*, \mathbf{CP}) \;=\; \min\left(1, \frac{P\left(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}^*, R\right)(1-q)^{\left|\overline{E}_{\mathbf{V_{CP}}, \mathbf{v}^*}\right|}}{P\left(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R\right)(1-q)^{\left|\overline{E}_{\mathbf{V_{CP}}, \mathbf{v}}\right|}}\right) \tag{4}$$

$$\approx\; \min\left(1, \left(\frac{|\mathbf{B}(\mathbf{CP}, \mathbf{v}_{t-1})|}{|\mathbf{B}(\mathbf{CP}, \mathbf{v}_t^*)|}\right)^R \frac{(1-q)^{\left|\overline{E}_{\mathbf{V_{CP}}, \mathbf{v}_t^*}\right|}}{(1-q)^{\left|\overline{E}_{\mathbf{V_{CP}}, \mathbf{v}_{t-1}}\right|}}\right). \tag{5}$$

Here, $\overline{E}_{\mathbf{V_{CP}}, \mathbf{v}} = \{(i, j) \in E(\mathbf{v}) : \exists\, V^* \in \mathbf{V_{CP}} \text{ s.t. } \{i\} \in V^*, \{j\} \notin V^*\}$ is called a Swendsen–Wang cut and denotes the set of edges in $E(\mathbf{v})$ that need to be cut to form connected components; $P\left(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}^*, R\right)$ is the probability of selecting the set of connected components

$\mathbf{V_{CP}}$ *in Step 3; and* $\mathbf{B}(\mathbf{CP}, \mathbf{v})$ *is the set of all connected components in* $\mathbf{CP}$ *which lie along the boundary of* $\mathbf{v}$.

In the Supplementary Appendix, we prove the following theorem, which states that under some assumptions we believe to hold in our application setting, the Markov chain produced by the proposed algorithm has a unique stationary distribution which is approximately uniform on the population of all valid partitions $\Omega(G, n)$ (Tierney, 1994).

THEOREM 1 *Suppose that the following three conditions hold:*

1. *Every valid partition can be obtained through a sequence of moves given in Algorithm 1.*
2. *There exists a pair of starting and candidate partitions* $\mathbf{v}, \mathbf{v}^*$ *and set of connected components* $\mathbf{CP}$ *such that* $0 < \alpha(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) < 1$.
3. *The selection of the number of boundary connected components* R *is independent of the current partition* $\mathbf{v}$ *and the outcome of connected components* $\mathbf{CP}$.

Then the stationary distribution of the Markov chain given by Algorithm 1 is uniform on $\Omega(G, n)$.

The acceptance ratio given in equation (4) is obtained through the Metropolis criterion (Metropolis *et al.*, 1953; Hastings, 1970):

$$\alpha(\mathbf{v} \to \mathbf{v}^*) \;=\; \min\left(1,\; \frac{q(\mathbf{v}^* \to \mathbf{v})}{q(\mathbf{v} \to \mathbf{v}^*)}\right), \tag{6}$$

where $q(\mathbf{v} \to \mathbf{v}^*)$ denotes the probability that, starting from partition $\mathbf{v}$, an iteration of Algorithm 1 described above obtains a candidate partition $\mathbf{v}^*$ through Steps 1–4. Computing numerators and denominators of this ratio separately is combinatorially expensive. However, following Barbu and Zhu (2005), we show in the Supplementary Appendix that substantial cancellation occurs, yielding the simplified expression given in (4).

In small-scale studies, $P\left(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}\right)$ may be computed exactly. However, for problems on the scale of redistricting, computing this probability is combinatorially expensive. When the set of boundary components is large compared to R, we may approximate $P\left(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}\right)$ by $\frac{R!}{|\mathbf{B}(\mathbf{CP}, \mathbf{v})|^R}$.

This makes the resulting convergence to a uniform distribution approximate rather than exact. Further details on the validity and accuracy of our approximation can found in the Supplementary Appendix.

The approximation requires counting all boundary connected components and finding all edges within $E(\mathbf{v}_{t-1})$ and $E(\mathbf{v}_t^*)$ that join a node in any connected component $V_{CP} \in \mathbf{V_{CP}}$ to a node not contained in V_{CP}. Since components in $\mathbf{V_{CP}}$ are not adjacent, this will ensure that the node not contained in V_{CP} will not be contained in another block in $\mathbf{V_{CP}}$.

Several additional remarks are in order. First, when implementing this algorithm, Step 3 requires the three operations: (1) identify all nodes that form a boundary of multiple partitions by comparing $G(\mathbf{v}_{t-1})$ with the original adjacency graph G, (2) identify all connected components that include at least one such node via the breadth-first or depth-first search algorithm, and (3) identify the partition to which each connected component belongs.

Second, in Step 3, we choose a positive integer R by randomly sampling it from a distribution with $\Pr(R = 1) > 0$ and $\Pr(R \ll |\mathbf{B}(\mathbf{CP}, \mathbf{v})|) \approx 1$ at each iteration. Placing positive probability on $R = 1$ allows for a single node swaps, making the assumption of ergodicity more plausible given sufficient connectivity of the adjacency graph G. When $1 < R \ll |\mathbf{B}(\mathbf{CP}, \mathbf{v})|$, the algorithm proposes multiple swaps. This increases the mixing rate of our algorithm while ensuring that condition 3 holds (approximately). Thus, we choose a sampling distribution which allows R to take a value greater than 1 while keeping the probability of $R = 1$ positive (e.g., a truncated Poisson distribution).

Third, conditions 1 and 2 seem likely to hold for redistricting applications. Under every adjacency graph tested, we have found at least one set of initial and candidate partitions such that $0 < \alpha(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) < 1$. Additionally the irreducibility condition 2 can be verified for each adjacency graph in which all valid partitions can be enumerated. For all verifiable examples in which single node swaps are permitted in the algorithm and the initial adjacency graph G is 2-connected—that is, when the graph remains connected after removing any one node and all of

its incident edges—the irreducibility condition was satisfied. Although a formal proof is difficult to obtain, we conjecture that this result holds in general.

Fourth, in the original algorithm of Barbu and Zhu (2005), R is set to 1 and instead the authors use a large value of q to create larger connected components. This strategy for improving mixing of the algorithm, though sensible in other settings, is not feasible for our case. The reason is that larger connected components typically include more units from the interior of each block. This in turn dramatically lowers the acceptance probability since many of the connected components shatter its currently assigned block when removed. Instead, our mixing of the Markov chain relies on making many simultaneous swaps of small connected components, which may result in slower convergence when compared with Barbu and Zhu (2005).

Finally, while this basic algorithm yields a sample of redistricting plans with contiguous districts, it does not incorporate common constraints imposed on redistricting process, including equal population and geographical compactness. In addition, our experience shows that the algorithm does not scale well for states with a medium or larger number of districts. Therefore, we now describe two important modifications to the basic algorithm.

2.3 Constraints and Reweighting

In a typical redistricting process, several additional constraints are imposed. Two most commonly applied constraints are equal population and geographical compactness. We first consider the equal population constraint. Suppose that we use p_i to denote the population size for node $\{i\}$ where the population parity for the state is given by $\bar{p} \equiv \sum_{i=1}^{m} p_i/n$. Then, the population equality constraint can be written as,

$$P_{\mathbf{v}} \;=\; \max_{1 \leq k \leq n} \left| \frac{\sum_{i \in V_k} p_i}{\bar{p}} - 1 \right| \;\leq\; \delta \tag{7}$$

where δ determines the degree to which one wishes to impose the constraint. For example, $\delta = 0.03$ implies that the population of all districts must be within 3% of the population parity.

Next, we consider the geographical compactness. No consensus exists about the exact meaning

of compactness and several alternative definitions have been proposed in the literature (see Niemi *et al.*, 1990). Here, we adopt the measure recently proposed by Fryer and Holden (2011). Let w_i be the population density of node $\{i\}$ and d_{ij} represent the distance between the centroids of nodes $\{i\}$ and $\{j\}$. The measure, which is called the relative proximity index, is based on the sum of squared distances among voters in each district relative to its minimum value. Then, the compactness constraint can be written as,

$$R_{\mathbf{v}} \;=\; \frac{\sum_{k=1}^{n} \sum_{i,j \in V_k, i<j} w_i w_j d_{ij}^2}{\min_{\mathbf{v}' \in \Omega(G,n)} \sum_{k=1}^{n} \sum_{i,j \in V'_k, i<j} w_i w_j d_{ij}^2} \;\leq\; \epsilon \tag{8}$$

where $V'_k \in \mathbf{v}'$, ϵ determines the strength of this constraint, and $\Omega(G, n)$ is the set of all redistricting plans with n contiguous districts. Fryer and Holden (2011) develops an approximate algorithm to efficiently compute the minimum of the sum of squared distances, i.e., the denominator of equation (8). The authors also show that this measure is invariant to geographical size, population density, and the number of districts of a state, thereby allowing researchers to compare the index across different states and time periods.

How can we uniformly sample redistricting plans under these additional constraints? One possibility is to discard any candidate partition that does not satisfy the desired constraints. In Algorithm 1, after Step 4, one could check whether the candidate partition $\mathbf{v}_t^*$ satisfies the constraints and if not go back to Step 3. However, such a strategy often dramatically slows down the algorithm and worsens mixing. Alternatively, researchers could run Algorithm 1 without any modification and then simply discard any sampled redistricting plans that do not meet the constraints. The problem of this approach is that many sampled plans may be discarded when strong constraints are imposed.

To overcome this difficulty, we propose to modify Algorithm 1 in the following manner. We first oversample the redistricting plans that are likely to meet the constraints. This means that fewer sampled plans are discarded due to the failure to satisfy the constraints. We then reweight the remaining redistricting plans such that they together approximate the uniform sampling from the population of all valid redistricting plans satisfying the constraints. To do this, we consider

the Gibbs distribution from statistical physics,

$$P(\mathbf{v}) \;=\; \frac{1}{z(\beta)} \exp\left(-\beta \sum_{V_k \in \mathbf{v}} \psi(V_k)\right) \tag{9}$$

where $\beta \geq 0$ is the inverse temperature and $z(\beta)$ is the normalizing constant. The function $\psi(\cdot)$ is chosen so that it reflects the constraint of interest. For example, we use $\psi(V_k) = |\sum_{i \in V_k} p_i/\bar{p} - 1|$ and $\psi(V_k) = \sum_{i,j \in V_k} w_i w_j d_{ij}^2$ for the equal population and geographical compactness constraints, respectively.

Algorithm 1 can be modified easily to sample from the non-uniform stationary distribution given in equation (9). In particular, we only need to change the acceptance probability in equation (5) of Step 5 to,

$$\alpha(\mathbf{v}_{t-1} \to \mathbf{v}_t^*) \;=\; \min\left(1,\; \frac{g_\beta(\mathbf{v}_t^*)\,|\mathbf{B}(\mathbf{CP}, \mathbf{v}_{t-1})|}{g_\beta(\mathbf{v}_{t-1})\,|\mathbf{B}(\mathbf{CP}, \mathbf{v}_t^*)|} \cdot (1-q)^{\left|\overline{E}_{\mathbf{v}_{\mathbf{CP}}, \mathbf{v}_t^*}\right| - \left|\overline{E}_{\mathbf{v}_{\mathbf{CP}}, \mathbf{v}_{t-1}}\right|}\right) \tag{10}$$

where $g_\beta(\mathbf{v}) \equiv \exp\left(-\beta \sum_{V_k \in \mathbf{v}} \psi(V_k)\right)$. Lastly, we reweight the sampled plans by $1/g_\beta(\mathbf{v})$ to approximate the uniform sampling from the population of all possible valid redistricting plans. If we resample the sampled plans with replacement using this importance weight, then the procedure is equivalent to the sampling/importance resampling (SIR) algorithm (Rubin, 1987).

2.4 Simulated and Parallel Tempering

One major drawback of the reweighting approach is that when each plan is weighted according to equation (9) the algorithm may have a harder time moving through the sample space. We use simulated and parallel tempering to improve the mixing of Algorithm 1 in such situations (Marinari and Parisi, 1992; Geyer and Thompson, 1995). We begin by describing how to apply simulated tempering in this context.

Recall that we want to draw from the distribution given in equation (9). We initialize a sequence of *inverse temperatures* $\{\beta^{(\ell)}\}_{\ell=0}^{r-1}$ where $\beta^{(0)}$ corresponds to the *cold temperature*, which is the target parameter value for inference, and $\beta^{(r-1)} = 0$ represents the *hot temperature* with $\beta^{(0)} > \beta^{(1)} > \cdots > \beta^{(r-1)} = 0$. After many iterations, we keep the MCMC draws obtained when

$\beta = \beta^{(0)}$ and discard the rest. By sampling under warm temperatures, simulated tempering allows for greater exploration of the target distribution. We then reweight the draws by the importance weight $1/g_{\beta^{(0)}}(\mathbf{v})$.

Specifically, we perform simulated tempering in two steps. First, we run an iteration of Algorithm 1 using the modified acceptance probability with $\beta = \beta^{(l)}$. We then make another Metropolis-Hastings decision on whether to change to a different value of β. The details of the algorithm are given below.

ALGORITHM 2 (SIMULATED TEMPERING) *Given the initial valid partition* $\mathbf{v}_0$ *and the initial temperature value* $\beta_0 = \beta^{(\kappa_0)}$ *with* $\kappa_0 = r - 1$*, the simulated tempering algorithm repeats the following steps at each iteration* t*,*

Step 1 (Run the basic algorithm with the modified acceptance probability): *Using the current partition* $\mathbf{v}_{t-1}$ *and the current temperature* $\beta_{t-1} = \beta^{(\kappa_{t-1})}$*, obtain a valid partition* $\mathbf{v}_t$ *by running one iteration of Algorithm 1 with the acceptance probability given in equation* (10)*.*

Step 2 (Choose a candidate temperature): *We set* $\kappa_t^* = \kappa_{t-1} - 1$ *with probability* $u(\kappa_{t-1}, \kappa_{t-1} - 1)$ *and set* $\kappa_t^* = \kappa_{t-1} + 1$ *with probability* $u(\kappa_{t-1}, \kappa_{t-1} + 1) = 1 - u(\kappa_{t-1}, \kappa_{t-1} - 1)$ *where* $u(\kappa_{t-1}, \kappa_{t-1} - 1) = u(\kappa_{t-1}, \kappa_{t-1} + 1) = 1/2$ *when* $1 \leq \kappa_{t-1} \leq r - 2$*, and* $u(r-1, r-2) = u(0, 1) = 1, u(r-1, r) = u(0, -1) = 0$*.*

Step 3 (Accept or reject the candidate temperature): *Set*

$$\kappa_t \;=\; \begin{cases} \kappa_t^*, & \text{with probability } \gamma(\kappa_{t-1} \to \kappa_t^*), \\ \kappa_{t-1}, & \text{with probability } 1 - \gamma(\kappa_{t-1} \to \kappa_t^*) \end{cases} \tag{11}$$

where

$$\gamma(\kappa_{t-1} \to \kappa_t^*) \;=\; \min\left(1,\; \frac{g_{\beta^{(\kappa_t^*)}}(\mathbf{v}_t)\; u(\kappa_t^*, \kappa_{t-1})\; w_{\kappa_t^*}}{g_{\beta^{(\kappa_{t-1})}}(\mathbf{v}_t)\; u(\kappa_{t-1}, \kappa_t^*)\; w_{\kappa_{t-1}}}\right) \tag{12}$$

where w_ℓ *is an optional weight given to each* $l \in \{0, 1, \ldots, r-1\}$*.*

Much like simulated tempering, parallel tempering is also useful for improving mixing in MCMC algorithms and for sampling from multimodal distributions (Geyer, 1991). Parallel tempering differs from simulated tempering in that instead of varying the temperature within a single

Markov chain, we run r copies of Algorithm 1 at r different temperatures, and after a fixed number of iterations we exchange the corresponding temperatures between two randomly selected adjacent chains using the Metropolis criterion. This algorithm has an advantage over Algorithm 2 in that we do not need to choose the prior probability of β, which typically has a significant effect on the mixing performance. However this advantage comes at the expense of increased computation as we are now running r chains instead of just one.

The nature of parallel tempering suggests that it should be implemented in a parallel architecture, which can be used to minimize computation time. Altekar *et al.* (2004) describe such an implementation using parallel computing and MPI, which we use as the basis for implementing our algorithm described below.

ALGORITHM 3 (PARALLEL TEMPERING) *Given r initial valid partitions $\mathbf{v}_0^{(0)}, \mathbf{v}_0^{(1)}, \ldots, \mathbf{v}_0^{(r-1)}$, a sequence of r decreasing temperatures $\beta^{(0)} > \beta^{(1)} > \cdots > \beta^{(r-1)} = 0$ with $\beta^{(0)}$ the target temperature for inference, and a swapping interval T, the parallel tempering algorithm repeats the following steps every iteration $t \in \{0, T, 2T, 3T, \ldots\}$,*

Step 1 (Run the basic algorithm with the modified acceptance probability): *For each chain $i \in \{0, 1, \ldots, r-1\}$, using the current partition $\mathbf{v}_t^{(i)}$ and the corresponding temperature $\beta^{(i)}$, obtain a valid partition $\mathbf{v}_{t+T}^{(i)}$ by running T iterations of Algorithm 1 with the acceptance probability given in equation* (10)*. This step is executed concurrently for each chain*

Step 2 (Propose a temperature exchange between two chains): *Randomly select two adjacent chains j and k and exchange information about the temperatures $\beta^{(j)}, \beta^{(k)}$ and the unnormalized likelihoods of the current partitions $g_{\beta^{(j)}}\left(\mathbf{v}_{t+T}^{(j)}\right), g_{\beta^{(k)}}\left(\mathbf{v}_{t+T}^{(k)}\right)$ using MPI*

Step 3 (Accept or reject the temperature exchange): *Exchange temperatures (i.e $\beta^{(j)} \leftrightarrows \beta^{(k)}$) with probability $\gamma\left(\beta^{(j)} \leftrightarrows \beta^{(k)}\right)$ where*

$$\gamma\left(\beta^{(j)} \leftrightarrows \beta^{(k)}\right) = \min\left(1, \frac{g_{\beta^{(j)}}\left(\mathbf{v}_{t+T}^{(k)}\right) g_{\beta^{(k)}}\left(\mathbf{v}_{t+T}^{(j)}\right)}{g_{\beta^{(j)}}\left(\mathbf{v}_{t+T}^{(j)}\right) g_{\beta^{(k)}}\left(\mathbf{v}_{t+T}^{(k)}\right)}\right) \tag{13}$$

All previously generated samples are assumed to have been generated at the current temperature of the chain

We note that the mixing performance of Algorithm 3 is affected by the choice of the temperature sequence $\beta^{(i)}$. While no sequence has been shown to be optimal in the literature, sequences with geometric spacing have been shown heuristically to produce reasonable results (Atchadé *et al.*, 2011; Kone and Kofke, 2005). For this reason, we used the sequence $\beta^{(i)} = \left(\beta^{(0)}\right)^{1-\frac{i}{r-1}}, i \in \{0, 1, \ldots, r-1\}$ for our implementation.

2.5 Comparison with Existing Algorithms

A number of substantive researchers use Monte Carlo simulation algorithms to sample possible redistricting plans under various criteria in order to detect instances of gerrymandering and understand the causes and consequences of redistricting (e.g., Engstrom and Wildgen, 1977; O'Loughlin, 1982; Cirincione *et al.*, 2000; McCarty *et al.*, 2009; Chen and Rodden, 2013). Most of these studies use a similar Monte Carlo simulation algorithm where a geographical unit is randomly selected as a "seed" for each district and then neighboring units are added to contiguously grow this district until it reaches the pre-specified population threshold. A representative of such algorithms, proposed by Cirincione *et al.* (2000) and implemented by Altman and McDonald (2011) in their open-source `BARD` package, is given here.

Algorithm 4 (The Standard Redistricting Simulator (Cirincione *et al.*, 2000)) *For each district, we repeat the following steps.*

Step 1: *From the set of unassigned units, randomly select the seed unit of the district.*

Step 2: *Identify all unassigned units adjacent to the district.*

Step 3: *Randomly select one of the adjacent units and add it to the district.*

Step 4: *Repeat Steps 2 and 3 until the district reaches the predetermined population threshold.*

Additional criteria can be incorporated into this algorithm by modifying Step 3 to select certain units. For example, to improve the compactness of the resulting districts, one may choose an adjacent unassigned unit that falls entirely within the minimum bounding rectangle of the emerging

district. Alternatively, an adjacent unassigned unit that is the closest to emerging district can be selected (see Chen and Rodden, 2013).

Nevertheless, the major problem of these simulation algorithms is their adhoc nature. For example, as the documentation of `BARD` package warns, the creation of earlier districts may make it impossible to yield contiguous districts. More importantly, the algorithms come with no theoretical result and are not even designed to uniformly sample redistricting plans even though researchers have a tendency to assume that they are. In contrast, the proposed algorithms described in Sections 2.2–2.4 are built upon the well-known theories and strategies developed in the literature on the Markov chain Monte Carlo methods. The disadvantage of our algorithms, however, is that they yield a dependent sample and hence their performance will hinge upon the degree of mixing. Thus, we now turn to the assessment of the empirical performance of the proposed algorithms.

3 Empirical and Simulation Studies

In this section, we assess the performance of the proposed algorithms in two ways. First, we conduct a small-scale validation study where, due to its size, all possible redistricting maps can be enumerated in a reasonable amount of time. We show that our algorithms can approximate the target distribution well when the standard algorithm commonly used in the literature fails. Second, we use the actual redistricting data to examine the convergence behavior of the proposed algorithms in more realistic settings using the redistricting data from New Hampshire (two districts) and Pennsylvania (nineteen districts). For these data, the computation of the true population distribution is not feasible. Instead, we evaluate the empirical performance of the proposed algorithms by examining the standard diagnostics of MCMC algorithms. We conclude with a runtime comparison between the standard and proposed algorithms, which shows that the proposed algorithm can scale quickly enough to answer substantive questions that the standard algorithm is unable to address.

To conduct these analyses, we integrate precinct-level data from two sources. We utilize

precinct-level shape files and electoral returns data from the Harvard Election Data Archive to determine precinct adjacency and voting behavior. We supplement this data with basic demographic information from the U.S. Census Bureau P.L. 94–171 summary files, which are compiled by the Census Bureau and disseminated to the 50 states in order to obtain population parity in decennial redistricting.

3.1 A Small-scale Validation Study

We conduct a validation study where we analyze the convergence of our algorithm to the target distribution on the 25 precinct set, which is shown as an adjacency graph in Figure 1. Due to the small size of these sets, all possible redistricting plans can be enumerated in a reasonable amount of time. We begin by considering the problem of partitioning each of these graphs into two districts. We apply the proposed algorithm (Algorithm 1) with the starting map obtained randomly by running the standard algorithm (Algorithm 4) once. In addition, we apply the standard algorithm, as implemented in the `BARD` package (Altman and McDonald, 2011), to compare its performance with that of our proposed algorithm. We then consider partitions of the 25 precinct set into three districts. The results of the proposed algorithm are based on a single chain of 10,000 draws while those of the standard algorithm are based on the same number of independent draws.

Before we give results, it should be noted that, even for this small-scale study, the enumeration of all valid partitions is a non-trivial problem. For partitions of 25 precincts into three districts, of the roughly $3^{25}/6 \approx 1.41 \times 10^{11}$ possible partitions, $82,623$ have three contiguous districts, and $3,617$ have district populations within 20% of parity.

A brief description of our enumeration algorithm is as follows. In the case of two districts, we choose an initial starting node and form a partition where one district is that initial node and the other district is the complement, provided the complement is connected. We then form connected components of two nodes comprised of that starting node and and nodes that are adjacent to that node. We identify all valid partitions where one district is a two-node component and the

other district is the complement of the component. We continue forming connected components of incrementally increasing sizes and finding valid partitions until all possible partitions are found. In the case of three precincts, if the complement of a connected component is comprised of two additional connected components, we store that partition as valid. If the complement is a single connected component, we apply the two-district algorithm on the complement. After this enumeration, we identify which partitions have districts with populations within a certain percentage of parity.

Figure 3 presents the results of the validation study with three districts and 25 precincts. We apply the proposed algorithm (Algorithm 1) with the starting map obtained randomly from the standard algorithm (Algorithm 4) (upper panel). These algorithms are also implemented with the simulated tempering (Algorithm 2; black dot-dashed lines) and parallel tempering (Algorithm 3; blue solid lines) strategies (the lower panel).

To implement these algorithms, we specify a sequence of temperatures $\{\beta^{(\ell)}\}_{\ell=0}^{r}$. For the population deviation of 20%, we chose a target temperature of $\beta^{(0)} = 5.4$, and for the population deviation of 10%, we chose a target temperature of $\beta^{(0)} = 9$. In both cases, we use $\beta^{(0)} = 0$. We choose these setups so that the rejection ratio is in the recommended 20–40% range (Geyer and Thompson, 1995) and the target temperature value is chosen based on the number of plans that meet the population constraint. In both cases, we use a subset of draws taken under the target temperature. We then resample the remaining draws using the importance weights $1/g_{\beta^{(\ell)}}(\mathbf{v})$, and finally subset down to the set of remaining draws that fall within the population target.

The left-upper plot of Figure 3 shows that when no constraint is imposed the proposed algorithm approximates the target distribution well while the sample from the standard algorithm is far from being representative of the population. In the plots of the middle and right columns, we impose the equal population constraint where only up to 20% and 10% deviation from the population parity is allowed, respectively. It is no surprise that the standard algorithm completely fails to approximate the true distribution as well in these cases (the middle and right plots in

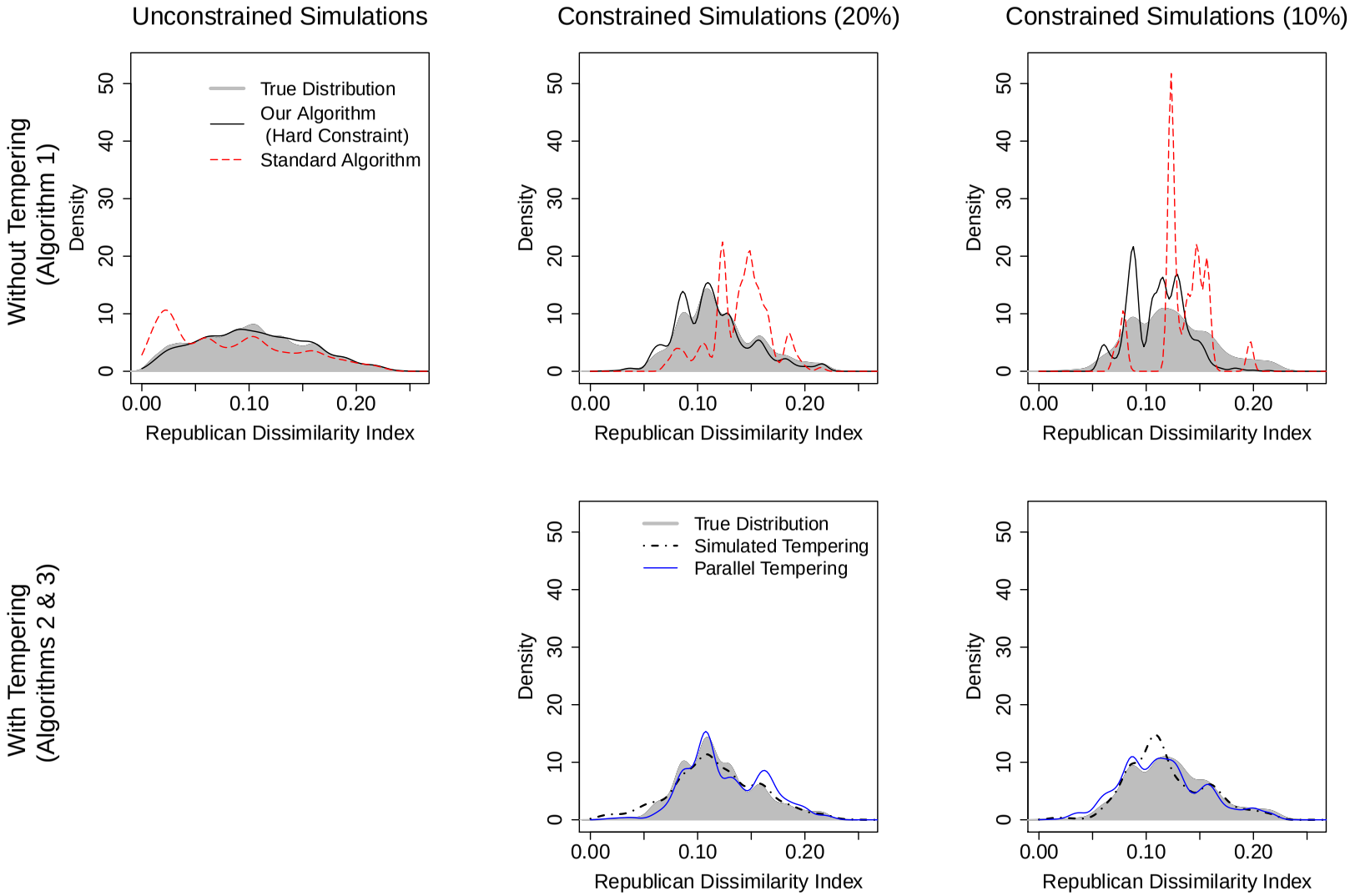

Figure 3: A Small-scale Validation Study with Three Districts. The underlying data is the 25 precinct set shown in the left plot of Figure 1. The plots in the first row show that the proposed algorithm (Algorithm 1; solid black lines) approximates well the true population distribution (grey histograms) when no (left plot) or weak (middle plot) equal population constraint is imposed. However, the algorithm exhibits poor performance when a stronger equal population constraint (right plot) is imposed. Finally, the standard algorithm (Algorithm 4; red dashed lines) fails to approximate the target distribution in all cases. In contrast, in the plots of the second row, the proposed algorithm with simulated tempering (Algorithm 2; black dot-dashed line) approximates the true population distribution well even when a stronger constraint is placed. The same exact pattern is observed for the parallel tempering algorithm (Algorithm 3; blue solid line). The results for each algorithm is based on a single chain of 10,000 draws.

the upper panel). In contrast, the proposed algorithms with simulated and parallel tempering approximate the true population distribution well. Even when a stronger constraint, i.e., 10%, is placed, the proposed algorithms with simulated tempering (Algorithm 2) and parallel tempering (Algorithm 3) maintain a good approximation.

3.2 Empirical Studies

The scale of the validation study presented above is small so that we can enumerate all possible redistricting plans in a reasonable amount of time. This allowed us to examine how well each algorithm is able to approximate the true population distribution. However, the scale of the study is too small to be realistic. Below, we apply the proposed algorithms to the 2008 election data and conduct standard convergence diagnostics of MCMC algorithms. While we cannot compare the distribution of sampled maps with the true population distribution, this empirical study enables us to investigate the performance of the proposed methods in realistic settings.

3.2.1 New Hampshire: Global Simulations

We first consider New Hampshire. The state has two congressional districts and consists of 327 precincts, and so this is one of the simplest realistic redistricting problems. The left panel of Figure 4 shows the implemented statewide redistricting plan as of 2008. Under this plan, Democrats and Republicans won a single congressional seat each. In 2008, Obama won 54% of votes in this state while his 2012 vote share was 52%. Redistricting in New Hampshire is determined by its state legislature and plans are passed as standard statutes, which makes them subject to gubernatorial veto. We apply the proposed basic algorithm (Algorithm 1), simulated tempering algorithm (Algorithm 2), and parallel tempering algorithm (Algorithm 3). The target population consists of all redistricting plans with contiguous districts and a maximum of 1% deviation from the population parity.

A total of 10 chains are run until 500,000 draws are obtained for each of the three algorithms. Inference is based on a total of 64,800 draws, which is the lowest number of draws across the

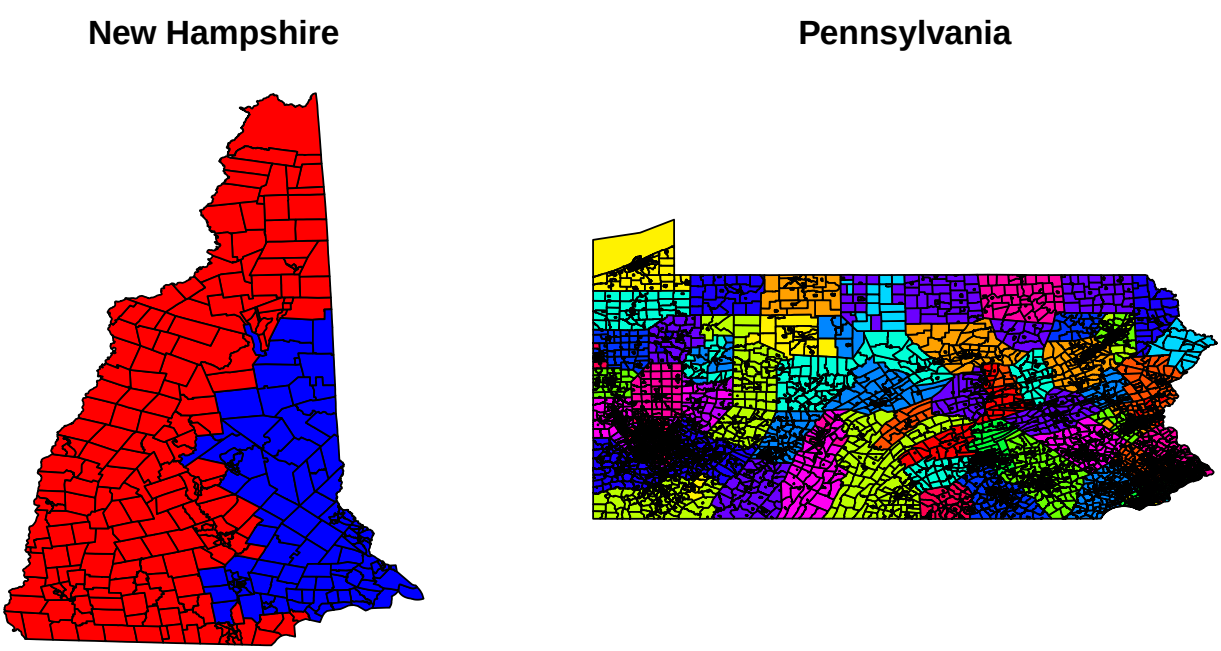

Figure 4: Precinct-level Maps of New Hampshire (327 precincts, two congressional districts) and Pennsylvania (9,256 precincts, 19 congressional districts). Colors correspond to precinct congressional district assignments in 2008. In New Hampshire, Democrats and Republicans each hold a single congressional seat under the 2008 redistricting plan. In Pennsylvania, Democrats hold twelve congressional seats while Republicans hold seven seats.

three algorithms that both satisfy the population constraint and were drawn under the target temperature value, $\beta^{(0)} = 27$. For starting values, we use independent draws from the standard algorithm (Algorithm 4 as implemented in the `BARD` package). For both the simulated and parallel tempering algorithms, after some preliminary analysis, we have decided to allow $\beta^{(\ell)}$ to take values between 0 and 27, using geometric spacing. As in the small-scale verification study, we only use draws taken under the target temperature, and then reweight according to the importance weights $1/g_{\beta^{(\ell)}(\mathbf{v})}$ before selecting all remaining draws that fall within the target parity deviation of 1%.

Figure 5 presents the results. The figure shows the autocorrelation plots (left column), the trace plots (middle column), and the Gelman-Rubin potential scale reduction factors (Gelman and Rubin, 1992; right column) for the basic algorithm (top panel), the simulated tempering algorithm (middle panel) and the parallel tempering algorithm (bottom panel). We use the logit transformed Republican dissimilarity index for all diagnostics. Both the simulated and parallel tempering

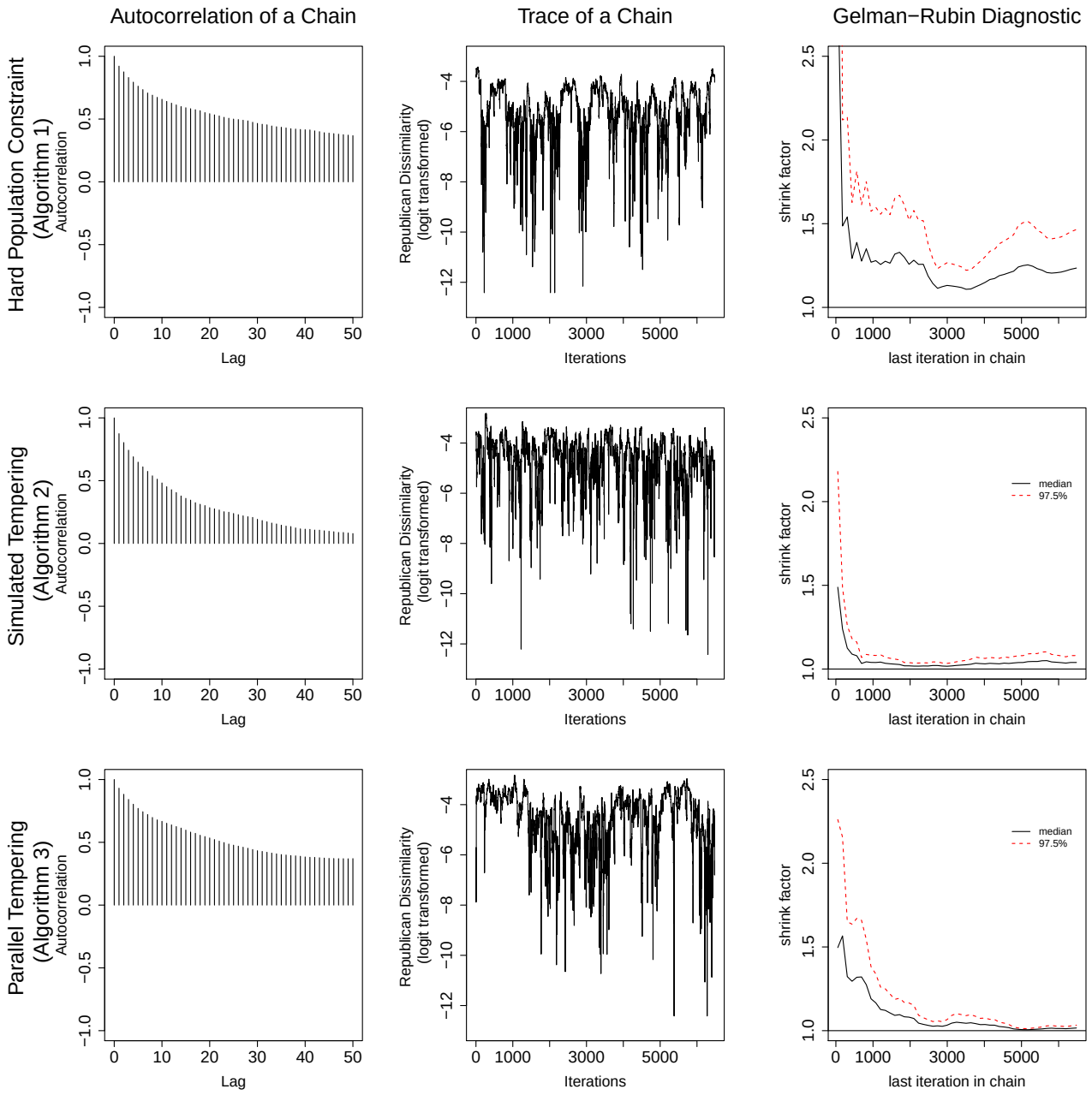

Figure 5: Convergence Diagnostics of the Proposed Algorithm for the 2008 New Hampshire Redistricting Data. The proposed basic algorithm (Algorithm 1; top panel), the simulated tempering algorithm (Algorithm 2; middle panel), and the parallel tempering algorithm (Algorithm 3; bottom panel) are applied to the New Hampshire data with 327 precincts and 2 congressional districts. The target population consists of all redistricting plans with contiguous districts and a maximum of 1% deviation from the population parity. A total of 10 chains are run with different starting maps for each algorithm until 500,000 draws are obtained, and inference is based on a total of 64,800 draws (the number of draws in the simulated tempering algorithm that are both drawn under the target temperature and satisfy the target population constraint). For the logit transformed Republican dissimilarity index, the autocorrelation plots (left column), the trace plots (middle column), and the Gelman-Rubin potential scale reduction factors (right column) are presented. The simulated and parallel tempering algorithms outperform the basic algorithm across all three diagnostics.

algorithms significantly outperform the basic algorithm. The former has a lower autocorrelation and mixes better. In addition, the potential scale reduction factor goes down quickly, suggesting that all the chains with different starting maps become indistinguishable from each other after approximately 1,000 draws.

We consider the distribution of Democratic-held congressional seats using the sample of plans generated by the parallel tempering algorithm. Under the implemented redistricting plan, Democrats held both of New Hampshire's two congressional districts, first winning both in the 2006 midterm elections and then holding on to both seats again in the 2008 general elections. In our simulations, we find that this is an overwhelmingly common outcome. Out of the 64,800 valid draws, only 87 of those suggest a redistricting plan where Democrats hold on to a single congressional district, and no simulations flip both seats to the Republicans. This suggests that, when assuming voting behavior similar to the 2008 general election, redistricting has little effect in New Hampshire with Democrats having a significant advantage in holding on to the two congressional seats.

3.2.2 Pennsylvania: Local Simulations

Next, we analyze the 2008 election data from Pennsylvania to examine how small changes to the existing congressional map would alter election outcomes. This state has a total of 19 congressional districts and 9,256 precincts, thereby providing a more challenging example when compared to New Hampshire. Thus, instead of exploring the entire space of valid redistricting plans as done for New Hampshire, we conduct "local simulations," in which the goal is to obtain a representative sample of valid redistricting plans within 5% deviation from the actual redistricting plan in place in 2008 (shown in the right-hand panel of Figure 4). Since in practice the redistricting plans do not change dramatically, this represents a realistic set of redistricting plans. As done for New Hampshire, we also make sure that the deviation from the population parity stays within 5% for all simulated redistricting plans. In 2008, 54% of the electorate voted for Obama while his vote share in the 2012 election for this state was 52%. Redistricting in Pennsylvania is determined by the Pennsylvania General Assembly, and a map is passed by majority vote of the commission's

members. The proposed map is then subject to a gubernatorial veto.

We utilize parallel tempering (Algorithm 3) together with the reweighting method described in Section 2.3 to obtain a sample from the target population. That is, we choose $\psi(V_k)$ in equation (9) to be the proportion of precincts in district k that are not shared with the corresponding district of the implemented redistricting plan. We then wish to obtain a sample of redistricting plans which satisfies the constraint $\frac{1}{K}\sum_{k=1}^{K}\psi(V_k) \leq \delta$ for some δ. For example, if we choose $\delta = 0.05$, then we obtain a sample of valid redistricting plans, 5% of whose precincts are switched from other districts of the 2008 redistricting plan. Finally, we also discard the simulated plans that fall outside 5% of population parity.

Following the recommendations given in the literature (Atchadé *et al.*, 2011; Kone and Kofke, 2005), we chose $\beta^{(\ell)}$ to be geometrically spaced, i.e., $\beta^{(\ell)} = \left(\beta^{(0)}\right)^{1-\frac{\ell}{r-1}}, \ell \in \{0, 1, \ldots, r-1\}$. After some preliminary analysis, we set the target temperature to be 2500 and $r = 10$, which led to good mixing behavior and provides a sufficient number of valid plans for subsequent inference. Since we are only exploring the local neighborhood of the 2008 plan, we initialize the Markov chain three times with different random seeds from the same starting map. We run each of the 10 temperature chains for 120,000 simulations for each initialization and thin the chain by 12. We then base inference off of those 10,000 simulations for each of the three initialized MPI chains. A standard diagnostic suggests that the algorithm has converged well: the Gelman-Rubin statistic falls under 1.1 after approximately 8,000 simulations after thinning.

Figure 6 presents the results of this analysis. We examine how the partisan bias and electoral competitiveness changes as the simulated plan moves farther from the implemented plan. Following the literature (see Grofman and King, 2007, and references therein), we use deviation from partisan symmetry as a measure of partisan bias. This measure evaluates the partisan implications of the counterfactual election outcomes under hypothetical uniform swings across districts. We vary the 2008 Democratic and Republican two-party vote shares uniformly across precincts from the actual vote share to various degrees.

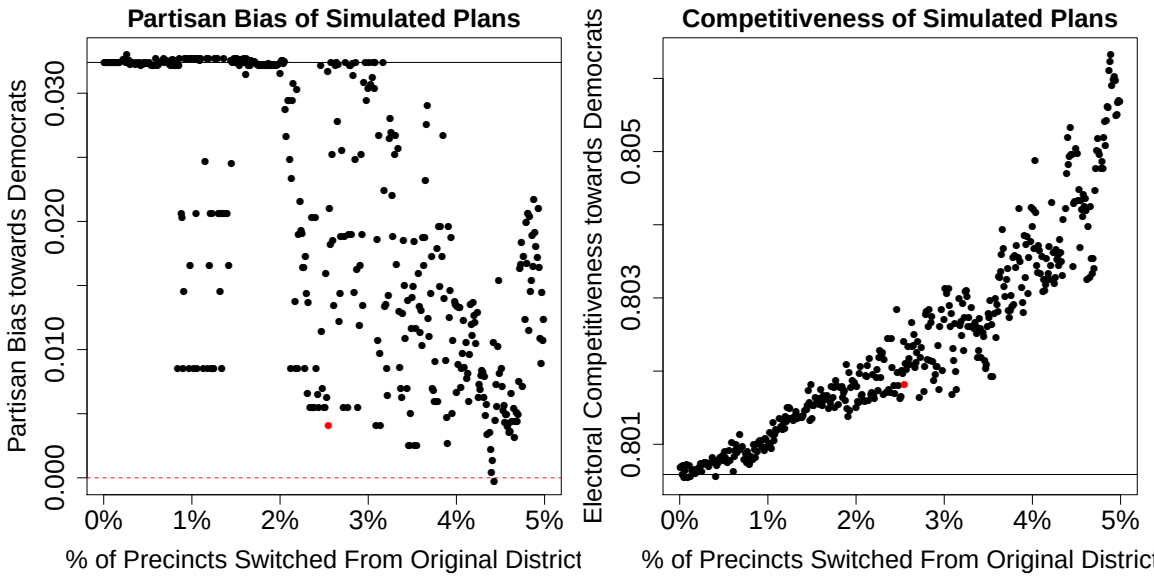

Figure 6: Local Simulations for the 2008 Pennsylvania Redistricting Plan. The plot shows that the average partisan bias (left panel) of the simulated plans decreases nearly to zero and average electoral competitiveness (right panel) increases sharply as more precincts are switched out of the original redistricting plan. The solid line represents the partisan bias (left panel) and electoral competitiveness (right panel) of the original plan, while the red dashed line indicates an unbiased redistricting plan, in which the seats gained from a vote swing of any value is equal to the number of seats lost in response to a vote swing of equal magnitude in the opposite direction. The minimal-bias plan we examine in Figure 7 is marked as a solid red circle in both panels.

Specifically, we consider all vote swings where the statewide two-party vote shares are between 40% and 60%. These vote shares are then aggregated according to each redistricting plan, yielding the counterfactual two-party vote shares at the district level. These district-level vote shares are in turn used to determine the number of hypothetical Democrat and Republican winners of the election. The resulting information is summarized as the seats-votes curve (Tufte, 1973), which is a non-decreasing step-function $f(x)$ evaluated within the range of vote shares x from 40% to 60%. Finally, let $f^*(x)$ be a non-decreasing step-function that is symmetric around the 50%–50% two-party vote share. Then, the partisan bias is formally defined as the standardized difference between the area below $f(x)$ and the area below $f^*(x)$,

$$\text{partisan bias} \;=\; \frac{1}{\eta}\int_{0.5-\eta}^{0.5+\eta} \left(f(x) - f^*(x)\right) dx \;=\; \frac{1}{\eta}\int_{0.5-\eta}^{0.5+\eta} f(x)dx - 1 \tag{14}$$

where the area under $f^*(x)$ is always equal to a uniform vote swing η. In our case, $0 \leq \eta \leq$

0.1. This partisan bias measure is scaled so that -1 indicates a maximally Republican-biased plan (where no vote swing of any magnitude favoring Democrats can ever flip a seat away from Republicans), while 1 indicates a maximally Democrat-biased plan (where no vote swing of any magnitude favoring Republicans can ever flip a seat away from Democrats). A bias measure of 0 represents an unbiased plan, where uniform vote swings result in equal seat gains or losses for each party.

To define electoral competitiveness, we follow Tam Cho and Liu (2016), who define competitiveness as a weighted and scaled average of vote share and seat share. According to this measure, a plan with an electoral competitiveness measure of 1 has an equal share of Democratic and Republican votes in each district, and seats are split equally between Democrats and Republicans across the state.[1] In contrast, a plan where electoral competitiveness is equal to 0 is dominated by Democrats or Republicans – all seats are held by one party, and the other party wins no votes for any seat. Formally, electoral competitiveness is defined as,

$$\text{electoral competitiveness} \;=\; 1 - T_p(1 + \alpha T_e)\beta \tag{15}$$

where

$$\begin{aligned} T_p &= \frac{1}{K}\left(\sum_{k=1}^{k}\left|V_k^D - \frac{1}{2}\right|\right) \\ T_e &= \left|S^D - \frac{1}{2}\right| \end{aligned}$$

where V_k^D is the vote-share for Democrats in district k, and S^D is the share of seats in the state won by Democrats. α determines the weight given to T_e, and β is a weighting factor that scales the measure to vary between 0 and 1. We follow Tam Cho and Liu (2016) by setting α to 1 and β to 4/3.

[1] In the original measure, the value of zero indicates the highest degree of competitiveness whereas the value of one indicates the complete lack of competitiveness. We flip this in order to make the interpretation more straightforward.

Figure 6 presents the analysis where the solid black line indicates the partisan bias and electoral competitiveness of the implemented plan, and the red dashed line represents an unbiased plan. The implemented plan, with a standardized partisan bias measure of approximately 0.035 and an electoral competitiveness measure of 0.801, is biased in favor of the Democrats. We find that the partisan bias of the initial plan can be nearly eliminated by swapping approximately 3% of all precincts in Pennsylvania. While the Pennsylvania map cannot be made perfectly competitive by making local swaps, the electoral competitiveness does increase as the simulations move further away from the implemented plan. The positive trend observed as more precincts are swapped out is driven by the change in the T_p term — congressional districts such as District 14 (representing the city of Pittsburgh), which in the implemented plan has a Democratic vote share of 90%, become more moderate as additional conservative, suburban precincts are swapped into the district.

We explore this finding further in Figure 7, which plots the partisan distribution of swapped and unswapped precincts in Pennsylvania's District 1 (top row) and 6 (bottom row) in the minimally biased plan found by swapping fewer than 3% of precincts in the local simulations. We achieve this plan by swapping 277 precincts from the original Pennsylvania map into new districts. Congressional District 1 encompasses Central and South Philadelphia, and supported President Obama with nearly 90% of its vote. As a result, Democratic control of the seat is heavily insulated from partisan swings. The minimally biased plan substitutes out 48 precincts with an average Democratic vote share of 90.5% for 34 new precincts from the surrounding suburbs with an average Democratic vote share of 84%. While still heavily Democratic, these changes make the new district more vulnerable to voter swings in the direction of the Republicans.

The bottom row of Figure 7 conducts the same analysis for Pennsylvania's third Congressional district, which is located in the northwest corner of Pennsylvania. In 2008, this was the only district in Pennsylvania to flip parties — in the case of this district, from Republicans to Democrats. In a Democrat-biased redistricting plan, Republican-held seats should be more vulnerable to swings such as those in 2008, when Democrats flipped 21 seats in the House of Representatives. To achieve

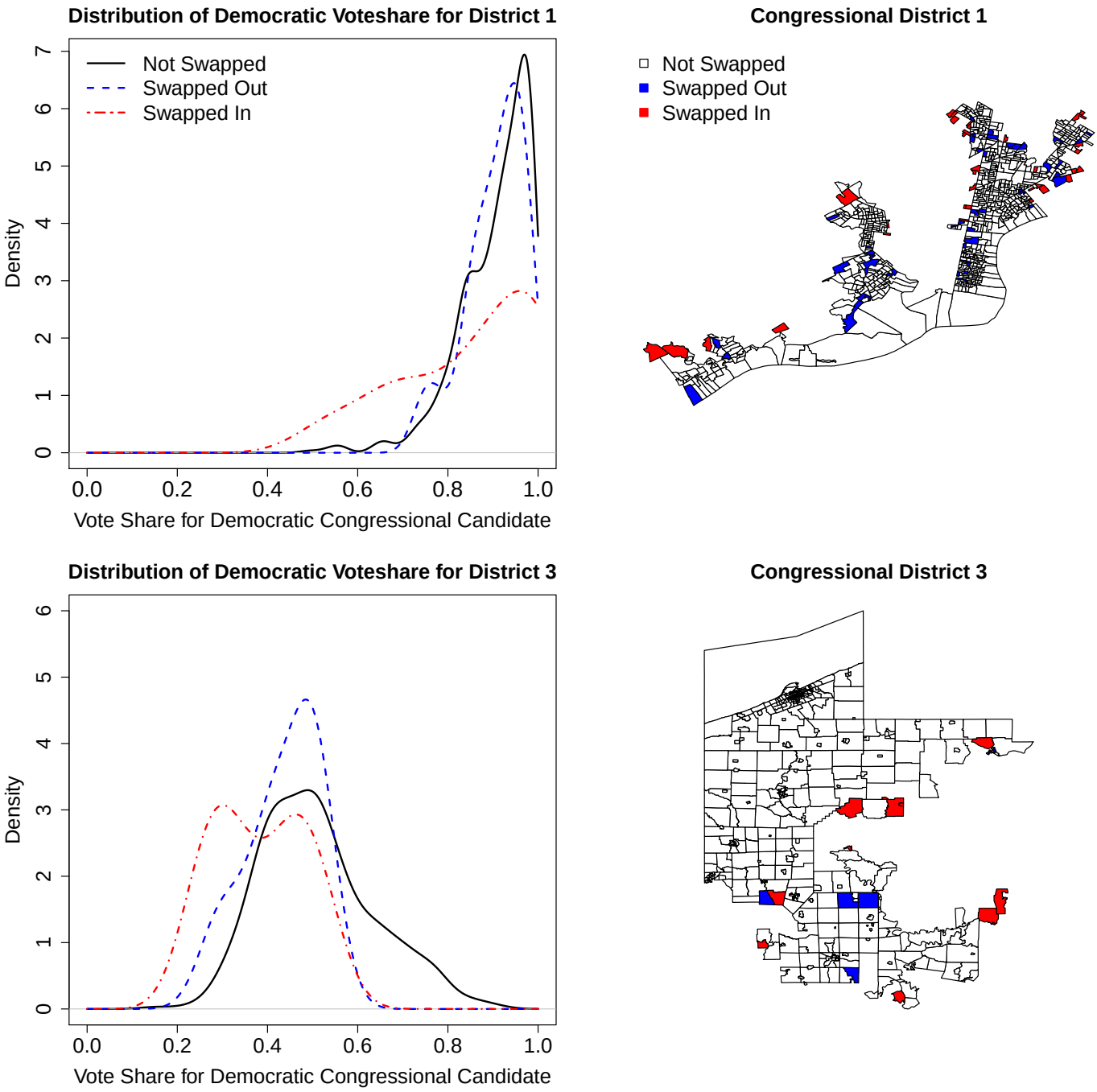

Figure 7: Partisan Patterns of the Swapped and Unswapped Precincts in Pennsylvania Local Simulations. The left column plots the distribution of the Democratic congressional vote-share for the precincts left untouched (black solid line), swapped in (red dot-dashed line), and swapped out (blue dashed line) of a congressional district to obtain a nearly unbiased plan. The right column shows the geography of the swapped and unswapped precincts in Congressional District 1 (top row) and Congressional District 3 (bottom row). In both congressional districts, the algorithm swaps out more partisan precincts for more moderate precincts in order to achieve more symmetric responses to voter swings.

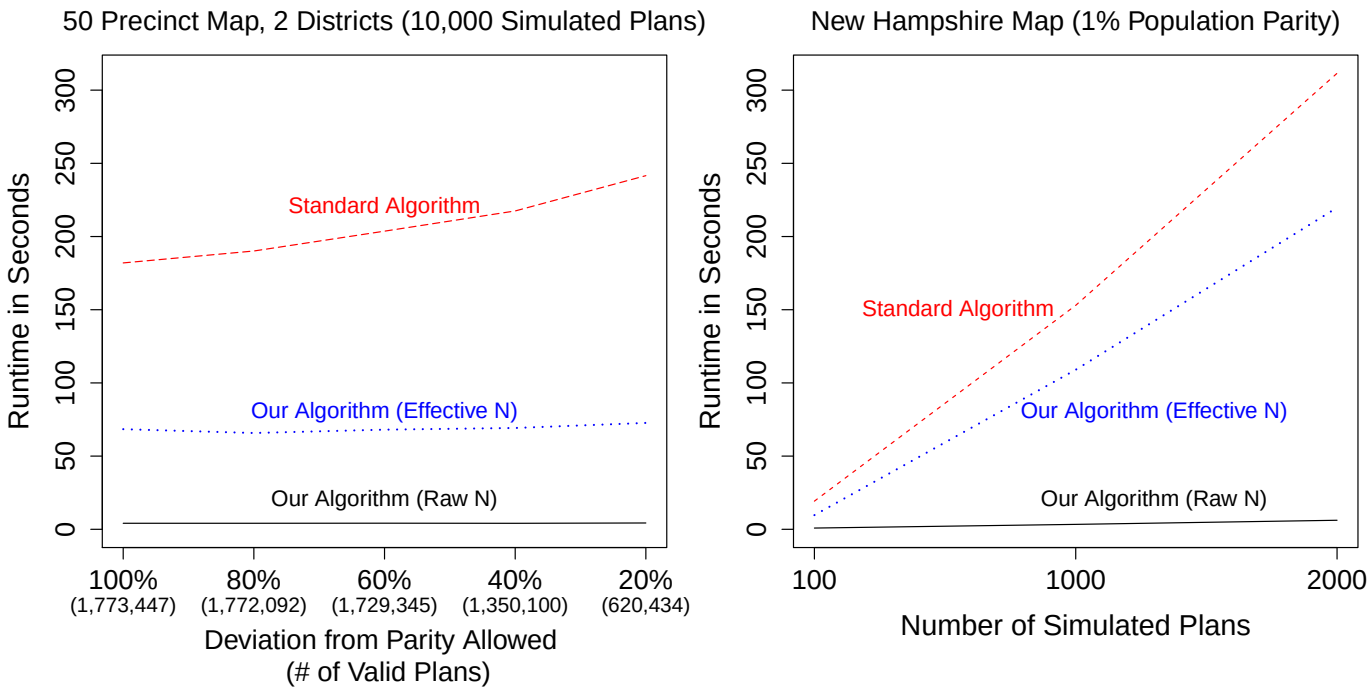

Figure 8: Runtime Comparison between the Proposed and Standard Algorithms in a Small-scale Validation Study and the New Hampshire Map. The runtime is compared between the proposed basic algorithm (Algorithm 1) and the standard algorithm (Algorithm 4). In the small-scale validation study, each algorithm is run until it yields 10,000 draws, while on the New Hampshire map, we vary the number of simulations while keeping the population parity limit constant at 1%. The runtime is much shorter for the proposed algorithm than the standard algorithm (Red dashed lines) when using the total number of simulated plans (Raw N; solid black lines), and even when measuring by the effective number of simulated plans (Effective N; dotted blue lines).

a more unbiased statewide plan, the algorithm in District 3 swaps in more conservative precincts (average Democratic vote share of 38.2%) for moderate precincts (average Democratic vote share of 43.5%), thereby making it more resilient to such swings. While the change in total Republican votes (an unbiased plan would have gained approximately 1,000 additional Republican votes in District 3) would not have been enough to keep the seat from switching parties, the proposed change would have increased the bar necessary to flip the seat in Democratic-leaning swing years.

3.3 Runtime Comparison

Lastly, we show that the proposed algorithm runs much faster than the standard algorithm, allowing researchers to conduct analyses that would have been impossible using the standard

algorithm. Figure 8 compares the runtime between the proposed basic algorithm (Algorithm 1) and the standard algorithm (Algorithm 4) on a 50 precinct map across a series of population parity constraints (left plot) and on the New Hampshire map with a 1% population parity constraint imposed (right plot). The 50 precinct map can be seen in the right plot of Figure 1. In the 50 precinct map, we hold the simulations constant at 10,000 while varying the population parity constraint, while we run 100, 1,000, and 2,000 simulations on the New Hampshire map. All analyses are conducted on a Linux server with 2.66 GHz Nehalem processors and 3GB RAM per node. Both Algorithm 4 and Algorithm 1 are parallelized across 8 cores. The effective sample size of the MCMC chain adjusts for autocorrelation and is calculated on the Republican dissimilarity index of each simulated plan, using the `effectiveSize()` function in the R package `coda`.

We find that under all settings we consider in this paper the proposed algorithm scales much better than the standard algorithm when measured by either the total number (Raw N; solid black lines) or effective number (Effective N; blue dotted lines) of simulated plans. In addition, the difference in runtime between the algorithms increases as the strength of equal population constraint (x-axis) increases. Note that the New Hampshire is the smallest redistricting problem with only two districts whereas Pennsylvania is a more challenging but typical redistricting problem. While we tried conducting a runtime comparison on the Pennsylvania map, we were unable to run a sufficient number of simulations using the standard algorithm to fully compare them. In the analysis of Subsection 3.2, we generate 120,000 simulations on the Pennsylvania map for each of 3 initializations of Algorithm 3, which took approximately 30 hours to complete. When testing the standard algorithm (Algorithm 4), we also initialized the algorithm across 3 cores, but obtained only 213 draws with a population parity of 5% or less over the course of 72 hours. In addition, Algorithm 4 is unable to incorporate a local constraint as described above because rejection sampling rarely accepts proposed draws. This suggests that the standard algorithm is of little use when incorporating substantive constraints commonly imposed on redistricting in practice.

4 Concluding Remarks

Over the last half century, a number of automated redistricting algorithms have been proposed in the methodological literature. Most of these algorithms have been designed to find an optimal redistricting plan given a certain set of criteria. However, many substantive researchers have been interested in characterizing the distribution of redistricting plans under various constraints. Unfortunately, few such simulation algorithms exist and even the ones that are commonly used by applied researchers have no theoretical justification.

In this paper, we propose a new automated redistricting simulator using Markov chain Monte Carlo. Unlike the existing standard algorithm, the proposed algorithms have a theoretical justification and approximate the target distribution well in a small-scale validation study. Even in more realistic settings where the computational challenge is greater, our initial analyses shows promising performance of the proposed algorithms. Nevertheless, it is still unclear whether these algorithms scale to those states with an even greater number of districts than those considered here. In the future, we plan to investigate whether simulated and parallel tempering strategies can overcome the computational challenge posed by those large states.

Another promising line of research is to examine the factors that predict the redistricting outcome. For example, substantive researchers are interested in how the institutional features of redistricting process (e.g., bipartisan commission vs. state legislature) determines the redistricting process. Such an analysis requires inferences about the parameters that are underlying our generative model. In contrast, in this paper we restricted our attention to the question of how to simulate redistricting plans given these model parameters. Therefore, a different approach is required to address this and other methodological challenges.

References

Abramowitz, A. I. (1983). Partisan redistricting and the 1982 congressional elections. *Journal of Politics* **45**, 3, 767–770.

Altekar, G., Dwarkadas, S., Huelsenbeck, J. P., and Ronquist, F. (2004). Parallel metropolis coupled markov chain monte carlo for bayesian phylogenetic inference. *Bioinformatics* **20**, 3, 407–415.

Altman, M. (1997). The computational complexity of automated redistricting: Is automation the answer. *Rutgers Computer & Technology Law Journal* **23**, 81–142.

Altman, M., MacDonald, K., and McDonald, M. (2005). From crayons to computers: The evolution of computer use in redistricting. *Social Science Computer Review* **23**, 3, 334–346.

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

Ansolabehere, S., Snyder, J. M., and Stewart, C. (2000). Old voters, new voters, and the personal vote: Using redistricting to measure the incumbency advantage. *American Journal of Political Science* **44**, 1, 17–34.

Atchadé, Y. F., Roberts, G. O., and Rosenthal, J. S. (2011). Towards optimal scaling of metropolis-coupled markov chain monte carlo. *Statistics and Computing* **21**, 4, 555–568.

Barbu, A. and Zhu, S.-C. (2005). Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**, 8, 1239–1253.

Barreto, M. A., Segura, G. M., and Woods, N. D. (2004). Mobilizing effect of majority-minority districts. *American Political Science Review* **98**, 1, 65–75.

Bozkaya, B., Erkut, E., and Laporte, G. (2003). A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research* **144**, 12–26.

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

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

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

Cirincione, C., Darling, T. A., and O'Rourke, T. G. (2000). Assessing South Carolina's 1990s congressional districting. *Political Geography* **19**, 189–211.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to algorithms*, vol. 6. MIT press Cambridge.

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

Fifield, B., Tarr, A., and Imai, K. (2015). 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, R. and Holden, R. (2011). Measuring the compactness of political districting plans. *Journal of Law and Economics* **54**, 3, 493–535.

Garfinkel, R. S. and Nemhauser, G. L. (1970). Political districting by implicit enumeration techniques. *Management Science* **16**, 8, B495–B508.

Gelman, A. and King, G. (1994). A unified method of evaluating electoral systems and redistricting plans. *American Journal of Political Science* **38**, 513–554.

Gelman, A. and Rubin, D. B. (1992). Inference from iterative simulations using multiple sequences (with discussion). *Statistical Science* **7**, 4, 457–472.

Geyer, C. J. (1991). Markov chain Monte Carlo maximum likelihood. *Interface Foundation of North America* Retrieved from the University of Minnesota Digital Conservancy, `http://hdl.handle.net/11299/58440`.

Geyer, C. J. and Thompson, E. A. (1995). Annealing Markov chain Monte Carlo with applications to ancestral inference. *Journal of the American Statistical Association* **90**, 909–920.

Grofman, B. and King, G. (2007). The future of partisan symmetry as a judicial test for partisan gerrymandering after *lulac v. perry*. *Election Law Journal* **6**, 1, 2–35.

Hastings, W. K. (1970). Monte Carlo sampling methods usings Markov chains and their applications. *Biometrika* **57**, 97–109.

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

Kone, A. and Kofke, D. A. (2005). Selection of temperature intervals for parallel-tempering simulations. *The Journal of chemical physics* **122**, 20, 206101.

Marinari, E. and Parisi, G. (1992). Simulated tempering: A new Monte Carlo scheme. *Europhysics Letters* **19**, 451–458.

McCarty, N., Poole, K. T., and Rosenthal, H. (2009). Does gerrymandering cause polarization? *Amerian Journal of Political Science* **53**, 3, 666–680.

Mehrotra, A., Johnson, E., and Nemhauser, G. L. (1998). An optimization based heuristic for political districting. *Management Science* **44**, 8, 1100–1114.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics* **21**, 1087–1092.

Nagel, S. S. (1965). Simplified bipartisan computer redistricting. *Stanford Law Journal* **17**, 5, 863–899.

Niemi, R. G., Grofman, B., Carlucci, C., and Hofeller, T. (1990). Measuring compactness and the role of a compactness standard in a test for partisan and racial gerrymandering. *Journal of Politics* **52**, 1155–1181.

O'Loughlin, J. (1982). The identification and evaluation of racial gerrymandering. *Annals of the Association of American Geographers* **72**, 2, 165–184.

Rubin, D. B. (1987). Comment: A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputation when fractions of missing information are modest:the SIR algorithm. *Journal of the American Statistical Association* **82**, 398, 543–546.

Swendsen, R. H. and Wang, J. S. (1987). Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters* **58**, 86–88.

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

Tierney, L. (1994). Markov chains for exploring posterior distributions (with discussion). *The Annals of Statistics* **22**, 1701–1762.

Tufte, E. R. (1973). The relationship between seats and votes in two-party systems. *American Political Science Review* **67**, 2, 540–554.

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

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

A Supplementary Appendix

We now show that if the Markov chain is *irreducible*—that is, from any initial partition, every valid partition can be obtained through a sequence of moves according to the modified Swendsen-Wang algorithm—then the stationary distribution of the Markov chain is uniform on valid partitions. As is the case with any MCMC algorithm, it suffices to show that our algorithm satisfies detailed balance and that irreducibility implies ergodicity in order to prove the above statement.

A.1 Proof of Theorem 1

We begin first by extending several of the proofs given in Barbu and Zhu (2005) for the SWC-1 algorithm necessary for showing detailed balance and for simplifying the acceptance ratio. The proofs given in the original paper apply only for the case that a single connected component changes assignment at each iteration of the algorithm. We will show that these proofs still hold for the case of multiple, nonadjacent connected components.

A.1.1 Definitions

Let $P(\mathbf{v} \rightarrow \mathbf{v}^*)$ denote the probability that, given the current partition $\mathbf{v}$, an iteration of the algorithm obtains a partition $\mathbf{v}^*$. For a current partition $\mathbf{v}$, let $\Omega_{\mathbf{v}}$ denote all the set of all possible connected components $\mathbf{CP}$ that can be formed by "turning on" edges in Step 3, and let $\Omega_{\mathbf{V_{CP}},\mathbf{v}}$ denote the set of all connected components $\mathbf{CP}$ that partition $G(\mathbf{v})$ such that $\mathbf{V_{CP}} \subset \mathbf{CP}$. For a set of connected components $\mathbf{CP}$, let $\mathbf{E}_{on,\mathbf{v},\mathbf{CP}}$ denote all sets of edges $E_{\mathbf{CP}}$ such that, when those edges are turned on from current partition $\mathbf{v}$, the connected components $\mathbf{CP}$ are formed. Recall the definition of $E(\mathbf{v})$ in (1) and define $\overline{E}_{\mathbf{CP},\mathbf{v}} \equiv E(\mathbf{v}) \setminus E_{\mathbf{CP}}$ as the set of "turned-off" edges given a partition $\mathbf{v}$ and a set of turned-on edges $E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}$. Note that, for the same set of turned on edges $E_{\mathbf{CP}}$, $\overline{E}_{\mathbf{CP},\mathbf{v}}$ may differ depending on the partition $\mathbf{v}$. Lastly, for any subset of connected components $\mathbf{V_{CP}}$, define

$$\overline{E}_{\mathbf{V_{CP}},\mathbf{v}} \equiv \{(i,j) \in E(\mathbf{v}) : \exists\, V_{CP} \in \mathbf{V_{CP}}\ s.t.\ \{i\} \in V_{CP}, \{j\} \notin V_{CP}\} \tag{16}$$

as the set of edges that need to be "cut" to form blocks of vertices $\mathbf{V_{CP}}$. This set is also known as the set of *Swendsen-Wang cuts*.

A.1.2 Lemmas

We begin by proving the following lemmas:

LEMMA 1 *Let* $\mathbf{CP} \in \Omega_{\mathbf{v}}$. *Then for all* $E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}$, $E_{\mathbf{CP}} \subset E(\mathbf{v})$.

Proof It is sufficient to note that the edge set $E(\mathbf{v})$ is formed when all possible edges are turned on. Hence, $E_{\mathbf{CP}}$ cannot contain an edge that is not also in $E(\mathbf{v})$. □

LEMMA 2 *Let* $\mathbf{v}, \mathbf{v}^*, \mathbf{v} \neq \mathbf{v}^*$ *denote partitions satisfying* $P(\mathbf{v} \to \mathbf{v}^*) > 0$, *and suppose that* $\mathbf{v}^*$ *is obtained from* $\mathbf{v}$ *by reclassifying vertices in connected components* $\mathbf{V_{CP}}$. *Then for any set of connected components* $\mathbf{CP}$ *such that* $\mathbf{V_{CP}} \subset \mathbf{CP}$,

$$\mathbf{E}_{on,\mathbf{v},\mathbf{CP}} = \mathbf{E}_{on,\mathbf{v}^*,\mathbf{CP}}. \tag{17}$$

Proof Without loss of generality, suppose there is a set of edges $E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}$ such that $E_{\mathbf{CP}} \notin \mathbf{E}_{on,\mathbf{v}^*,\mathbf{CP}}$. Thus, there must be an edge $(i,j) \in E_{\mathbf{CP}}$ that can be turned on in $\mathbf{v}$ but cannot be turned on in $\mathbf{v}^*$; by Lemma 1, it follows that $(i,j) \in E(\mathbf{v})$ but $(i,j) \notin E(\mathbf{v}^*)$. Therefore, there exists a block $V_1 \in \mathbf{v}$ and two blocks $V_1^*, V_2^* \in \mathbf{v}^*$, $V_1^* \cap V_2^* = \emptyset$, such that $\{i\}, \{j\} \in V_1$, $\{i\} \in V_1^*$, $\{j\} \in V_2^*$.

Without loss of generality, now suppose vertex $\{i\}$ is reclassified between $\mathbf{v}$ and $\mathbf{v}^*$. Since only vertices in $\mathbf{V_{CP}}$ are reclassified when transitioning between partition $\mathbf{v}$ and $\mathbf{v}^*$, and since two adjacent connected components cannot both belong to $\mathbf{V_{CP}}$, it follows that there exists $V_{CP} \in \mathbf{V_{CP}}$, $V^* \in \mathbf{CP} \setminus \mathbf{V_{CP}}$, such that $\{i\} \in V_{CP}$ and $\{j\} \in V^*$. However, in order for $V_{CP} \subset \mathbf{CP}$, all edges connecting a vertex in V_{CP} to one outside of V_{CP} must be turned off. Thus, $(i,j) \notin E_{\mathbf{CP}}$, a contradiction. We conclude that (17) holds. □

LEMMA 3 *For any choice of* $\mathbf{V_{CP}}$*, if* $\mathbf{CP} \in \Omega_{\mathbf{V_{CP}},\mathbf{v}}$ *and if* $E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}$*, then* $\overline{E}_{\mathbf{V_{CP}},\mathbf{v}} \subset \overline{E}_{\mathbf{CP},\mathbf{v}}$

Proof Observe that no edge in $E_{\mathbf{CP}}$ connects vertices in two distinct connected components in **CP**. Hence, $\overline{E}_{\mathbf{CP},\mathbf{v}}$ must contain all edges $(i,j) \in E(\mathbf{v})$ such that $\{i\} \in V_1, \{j\} \in V_2$ for two distinct connected components $V_1, V_2 \in \mathbf{CP}$ in the same block of $\mathbf{v}$. Since $\mathbf{CP} \in \Omega_{\mathbf{V_{CP}},\mathbf{v}}$, $\mathbf{V_{CP}}$ must be comprised of a subset of non-adjacent connected components in **CP**. Thus, all edges in $\overline{E}_{\mathbf{V_{CP}},\mathbf{v}}$ must join two vertices from two distinct connected components $V_1, V_2 \in \mathbf{CP}$ contained within the same block in $\mathbf{v}$. Therefore $\overline{E}_{\mathbf{V_{CP}},\mathbf{v}} \subset \overline{E}_{\mathbf{CP},\mathbf{v}}$. □

LEMMA 4 *Let* $\mathbf{v}, \mathbf{v}^*, \mathbf{v} \neq \mathbf{v}^*$ *denote partitions satisfying* $P(\mathbf{v} \to \mathbf{v}^*) > 0$*, and suppose that* $\mathbf{v}^*$ *is obtained from* $\mathbf{v}$ *by reclassifying vertices in connected components* $\mathbf{V_{CP}}$*. Then for any edge set* $E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}$*,*

$$\overline{E}_{\mathbf{CP},\mathbf{v}} \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}} = \overline{E}_{\mathbf{CP},\mathbf{v}^*} \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}. \tag{18}$$

Proof In the formation of $\mathbf{v}^*$ from $\mathbf{v}$, we note that the edge sets $E(\mathbf{v})$ and $E(\mathbf{v}^*)$ differ only in the edges that join all $V_{CP} \in \mathbf{V_{CP}}$ to its respective block in the partition ($\mathbf{v}$ or $\mathbf{v}^*$), i.e. the Swendsen-Wang cuts. It follows that $E(\mathbf{v}^*)$ can be expressed as

$$E(\mathbf{v}^*) \;=\; \big(E(\mathbf{v}) \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}\big) \cup \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*} \tag{19}$$

To see this, recall that $E(\mathbf{v})$ and $E(\mathbf{v}^*)$ consist only of edges connecting nodes which belong to the same block in $\mathbf{v}$ and $\mathbf{v}^*$, respectively. Thus when constructing $E(\mathbf{v}^*)$ from $E(\mathbf{v})$, we must remove all edges connecting each $V_{CP} \in \mathbf{V_{CP}}$ to its respective block in $\mathbf{v}$, $\overline{E}_{\mathbf{V_{CP}},\mathbf{v}}$, and add all edges connecting each V_{CP} to its newly assigned block in $\mathbf{v}^*$, $\overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}$.

Next, we claim that

$$(E(\mathbf{v}) \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}) \cap \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*} = \emptyset \tag{20}$$

To see that this is true, suppose that $(E(\mathbf{v}) \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}) \cap \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*} \neq \emptyset$. Thus there exists some $e \in \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}$ that is also in $E(\mathbf{v}) \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}$, which implies that $e \in E(\mathbf{v})$ and $e \notin \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}$ and thus $\overline{E}_{\mathbf{V_{CP}},\mathbf{v}} \not\subset E(\mathbf{v})$. However this cannot be true since $\overline{E}_{\mathbf{V_{CP}},\mathbf{v}} \subset E(\mathbf{v})$ by definition, and we conclude that (20) must hold. We can therefore rewrite (19) as

$$E(\mathbf{v}^*) \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*} = E(\mathbf{v}) \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}} \tag{21}$$

Now since for any **CP** we have $E_{\mathbf{CP}} \subset E(\mathbf{v})$ and $E_{\mathbf{CP}} \subset E(\mathbf{v}^*)$, and since $\overline{E}_{\mathbf{CP},\mathbf{v}} \equiv E(\mathbf{v}) \setminus E_{\mathbf{CP}}$, we subtract $E_{\mathbf{CP}}$ from both sides of (21) and obtain the desired result. □

LEMMA 5 *For any* $\mathbf{v}, \mathbf{v}^*$ *satisfying* $P(\mathbf{v} \to \mathbf{v}^*) > 0$ *and for any choice of* $\mathbf{V_{CP}} \subset \Omega_{\mathbf{v}} \cap \Omega_{\mathbf{v}^*}$

$$\Omega_{\mathbf{V_{CP}},\mathbf{v}} = \Omega_{\mathbf{V_{CP}},\mathbf{v}^*} \tag{22}$$

Proof Without a loss of generality, suppose that (22) fails to hold and there exists a set of connected components $\mathbf{CP} \in \Omega_{\mathbf{V_{CP}},\mathbf{v}}$ such that $\mathbf{CP} \notin \Omega_{\mathbf{V_{CP}},\mathbf{v}^*}$. That is, there must be a connected component $V^* \in \mathbf{CP}$ that cannot be formed by turning on edges in $\mathbf{v}^*$. Thus, there must exist subsets of vertices $V_1^*, V_2^* \subset V^*$ such that V_1^* and V_2^* belong to the same block in $\mathbf{v}$ but are assigned to different blocks in $\mathbf{v}^*$. However, from $\mathbf{v}$ to $\mathbf{v}^*$, all vertices that belong to the same block $V_{CP} \in \mathbf{V_{CP}}$ are all assigned to the same block in $\mathbf{v}^*$; hence, without loss of generality, $V_1^* \subset V_{CP} \in \mathbf{V_{CP}}$ and $V_2^* \cap V_{CP} = \emptyset$. Thus, $V^*, V_{CP} \in \mathbf{CP}$, $V^* \neq V_{CP}$, and $V^* \cap V_{CP} \neq \emptyset$. That is, **CP** violates the definition of a partition of V, a contradiction. Therefore, $\Omega_{\mathbf{V_{CP}},\mathbf{v}} = \Omega_{\mathbf{V_{CP}},\mathbf{v}^*}$. □

A.1.3 Detailed Balance Condition

We will now extend the acceptance probability presented in Theorem 2 of Barbu and Zhu (2005) to work for reassigning multiple connected components along the partition boundary. As is the case in the original paper, there is only one path moving between the states $\mathbf{v}$ and $\mathbf{v}^*$: selecting

and changing the assignment of each $V_{CP} \subset \mathbf{V_{CP}}$. Therefore the ratio of proposal densities is given by

$$\begin{aligned}\frac{q(\mathbf{v}^* \to \mathbf{v})}{q(\mathbf{v} \to \mathbf{v}^*)} &= \frac{P(\mathbf{V_{CP}}|\mathbf{v}^*)\prod_{V_{CP}\in\mathbf{V_{CP}}} P(s|V_{CP},\mathbf{v}^*)}{P(\mathbf{V_{CP}}|\mathbf{v})\prod_{V_{CP}\in\mathbf{V_{CP}}} P(s|V_{CP},\mathbf{v})} \\ &= \frac{\sum_{\mathbf{CP}\in\Omega_{\mathbf{V_{CP}},\mathbf{v}^*}} P(\mathbf{CP}|\mathbf{v}^*)P(R|\mathbf{CP},\mathbf{v}^*)P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v}^*,R)\prod_{V_{CP}\in\mathbf{V_{CP}}} P(s|V_{CP},\mathbf{v}^*)}{\sum_{\mathbf{CP}\in\Omega_{\mathbf{V_{CP}},\mathbf{v}}} P(\mathbf{CP}|\mathbf{v})P(R|\mathbf{CP},\mathbf{v})P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v},R)\prod_{V_{CP}\in\mathbf{V_{CP}}} P(s^*|V_{CP},\mathbf{v})},\end{aligned} \tag{23}$$

where $P(\mathbf{CP}|\mathbf{v})$ is the probability that, given the current partition $\mathbf{v}$, the connected components $\mathbf{CP}$ are formed in Steps 1–2 of the algorithm, $P(R|\mathbf{CP},\mathbf{v})$ is the probability of selecting R boundary connected components in Step 3, $P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v},R)$ is the probability that, given the current partition $\mathbf{v}$, the set of connected components $\mathbf{CP}$, and the number of selected boundary connected components R, the connected components $\mathbf{V_{CP}}$ are selected at Step 3, and $P(s^*|V_{CP},\mathbf{v})$ is the probability of assigning $V_{CP} \in \mathbf{v}$ to block $V_{s^*m} \in \mathbf{v}^*$.

Unfortunately the sums in (23) are combinatorially expensive to compute. In the original paper the authors show that this ratio greatly simplifies due to cancellation, however the same cancellation does not occur with our algorithm due to the way we select the components $\mathbf{V_{CP}}$. To deal with this issue, we instead define the acceptance ratio using the unmarginalized proposal distribution $q(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP})$:

$$\begin{aligned}\alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) &\equiv \frac{q(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP})}{q(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP})}\frac{g(\mathbf{v}^*)}{g(\mathbf{v})} \\ &= \frac{P(\mathbf{CP}|\mathbf{v}^*)P(R|\mathbf{CP},\mathbf{v}^*)P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v}^*,R)\prod_{V_{CP}\in\mathbf{V_{CP}}} P(s|V_{CP},\mathbf{v}^*)}{P(\mathbf{CP}|\mathbf{v})P(R|\mathbf{CP},\mathbf{v})P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v},R)\prod_{V_{CP}\in\mathbf{V_{CP}}} P(s^*|V_{CP},\mathbf{v})}\frac{g(\mathbf{v}^*)}{g(\mathbf{v})},\end{aligned} \tag{24}$$

where $g(\mathbf{v})$ is an arbitrary unnormalized target distribution on all valid partitions. Using this ratio, the joint acceptance probability is therefore given by

$$\alpha(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) \equiv \min\left\{1, \alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP})\right\}. \tag{25}$$

We will now show that using this acceptance probability in our algorithm still preserves the detailed balance condition necessary for the existence of a stationary distribution $f(\mathbf{v}) \propto g(\mathbf{v})$.

For, $\mathbf{v} \neq \mathbf{v}^*$, define $P(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP})$ as the probability that one iteration of the algorithm forms connected components $\mathbf{CP}$, and that vertices in connected components $\mathbf{V_{CP}} \in \mathbf{CP}$ are reassigned to form $\mathbf{v}^*$ from current partition $\mathbf{v}$. We can express this probability as:

$$\begin{aligned} P(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) &= q(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP})\alpha(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) \\ &= P(\mathbf{CP}|\mathbf{v})P(R|\mathbf{CP}, \mathbf{v})P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R) \\ &\quad \times \left(\prod_{V_{CP} \in \mathbf{V_{CP}}} P(s^*|V_{CP}, \mathbf{v}) \right) \alpha(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}). \end{aligned} \tag{26}$$

We claim that for all $\mathbf{CP} \in \Omega_{\mathbf{V_{CP}}, \mathbf{v}}$ and for any distribution $f(\mathbf{v})$,

$$f(\mathbf{v})P(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) = f(\mathbf{v}^*)P(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}). \tag{27}$$

To see that this is true, note that when $\alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) < 1$, then $\alpha^*(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}) > 1$, and we have

$$\begin{aligned} f(\mathbf{v})P(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) &= f(\mathbf{v}^*)P(\mathbf{CP}|\mathbf{v}^*)P(R|\mathbf{CP}, \mathbf{v}^*)P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}^*, R) \prod_{V_{CP} \in \mathbf{V_{CP}}} P(s|V_{CP}, \mathbf{v}^*) \\ &= f(\mathbf{v}^*)P(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}), \end{aligned} \tag{28}$$

where we have used the fact that $\frac{g(\mathbf{v})}{g(\mathbf{v}^*)} = \frac{f(\mathbf{v})}{f(\mathbf{v}^*)}$. By symmetry of the acceptance ratio, we also see that (27) holds for $\alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) > 1$. When $\alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) = \alpha^*(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}) = 1$, we have

$$\begin{aligned} f(\mathbf{v})P(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) &= f(\mathbf{v})q(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) \\ &= f(\mathbf{v}^*)q(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}) = f(\mathbf{v}^*)P(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}), \end{aligned} \tag{29}$$

and hence (27) holds for all $\mathbf{CP} \in \Omega_{\mathbf{V_{CP}}, \mathbf{v}}$. Now since $\Omega_{\mathbf{V_{CP}}, \mathbf{v}} = \Omega_{\mathbf{V_{CP}}, \mathbf{v}^*}$ by Lemma 5, marginalizing out $\mathbf{CP}$ from the L.H.S. of (27) gives

$$\begin{aligned} f(\mathbf{v})P(\mathbf{v} \to \mathbf{v}^*) &= f(\mathbf{v}) \sum_{\mathbf{CP} \in \Omega_{\mathbf{V_{CP}}, \mathbf{v}}} P(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) \\ &= f(\mathbf{v}^*) \sum_{\mathbf{CP} \in \Omega_{\mathbf{V_{CP}}, \mathbf{v}}} P(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}) \\ &= f(\mathbf{v}^*) \sum_{\mathbf{CP} \in \Omega_{\mathbf{V_{CP}}, \mathbf{v}^*}} P(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}) = f(\mathbf{v}^*)P(\mathbf{v}^* \to \mathbf{v}), \end{aligned} \tag{30}$$

and therefore the detailed balance condition holds. □

A.1.4 Proof of Strong Ergodicity from Irreducibility

Ergodicity is established if it can be shown that the Markov chain is both aperiodic and positive recurrent. Since our Markov chain has a finite number of states, at least one state must be positive recurrent, and since irreducibility establishes a single communicating class, it follows that all states must be positive recurrent. Hence in order to show ergodicity, we need only to show that at least one state that is aperiodic. Thus if there exists some $\mathbf{v}, \mathbf{v}^*$ with $\mathbf{v} \neq \mathbf{v}^*$ and some $\mathbf{CP} \in \Omega_{\mathbf{V_{CP}},\mathbf{v}}$ such that $\alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) \neq 1$, then the Markov chain is aperiodic. Letting $P(\mathbf{v} \not\to \mathbf{v}^*)$ be the probability that $\mathbf{v}^*$ is proposed from $\mathbf{v}$ and rejected, under these conditions we have either

$$P(\mathbf{v} \to \mathbf{v}) \;\geq\; P(\mathbf{v} \not\to \mathbf{v}^*) \;\equiv\; \sum_{\mathbf{CP} \in \Omega_{\mathbf{V_{CP}},\mathbf{v}}} q(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) \left[1 - \alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP})\right] \;>\; 0 \quad (31)$$

or

$$P(\mathbf{v}^* \to \mathbf{v}^*) \;\geq\; P(\mathbf{v}^* \not\to \mathbf{v}) \;\equiv\; \sum_{\mathbf{CP} \in \Omega_{\mathbf{V_{CP}},\mathbf{v}^*}} q(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP}) \left[1 - \alpha^*(\mathbf{v}^* \to \mathbf{v}, \mathbf{CP})\right] \;>\; 0, \quad (32)$$

and hence that state has period 1. It is reasonable to assume that in any application setting, $\alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) = 1$ cannot always be true as there are a combinatorial number of states $\mathbf{v}$ and connected components $\mathbf{CP}$, and the scenarios in which everything in (23) always cancels out are extremely contrived and unrealistic. Even in the unlikely event that $\alpha^*(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP}) = 1$ for some network, the target distribution or the edge probabilities q_e can always be reweighted to produce an aperiodic state. Hence, the Markov chain must be aperiodic and, consequently, ergodic. □

A.2 Simplifying the Acceptance Ratio

We now show that through substantial cancellation, the acceptance probability $\alpha(\mathbf{v} \to \mathbf{v}^*, \mathbf{CP})$ is easily computable if $P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R)$ is easily computable. We note that steps in deriving the simplified form mimic those given in Barbu and Zhu (2005). We then discuss the computation of $P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R)$ under the case in which $\mathbf{V_{CP}}$ is uniformly selected at random from *eligible* boundary connected components–those whose removal will not shatter a current block of the partition.

A.2.1 Simplification of $P(\mathbf{CP}|\mathbf{v})$

Let $P(\mathbf{CP}|\mathbf{v})$ denote the probability that, given the current partition $\mathbf{v}$, we obtain the connected components $\mathbf{CP}$ in Steps 1–2 of Algorithm 1. It follows that we can write $P(\mathbf{CP}|\mathbf{v})$ as:

$$\begin{aligned} P(\mathbf{CP}|\mathbf{v}) &= \sum_{E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}} \prod_{e \in E_{\mathbf{CP}}} q_e \prod_{e \in \overline{E}_{\mathbf{CP},\mathbf{v}}} (1-q_e) \\ &= \sum_{E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}} \prod_{e \in E_{\mathbf{CP}}} q_e \prod_{e \in \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}} (1-q_e) \prod_{e \in \overline{E}_{\mathbf{CP},\mathbf{v}} \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}} (1-q_e). \end{aligned} \tag{33}$$

Since $\overline{E}_{\mathbf{V_{CP}},\mathbf{v}}$ does not depend on $\mathbf{E}_{on,\mathbf{v},\mathbf{CP}}$, we can factor out this term from sum.

$$P(\mathbf{CP}|\mathbf{v}) = \prod_{e \in \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}} (1-q_e) \sum_{E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}} \prod_{e \in E_{\mathbf{CP}}} q_e \prod_{e \in \overline{E}_{\mathbf{CP},\mathbf{v}} \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}} (1-q_e). \tag{34}$$

From Lemmas 2 and 4, we can write $P(\mathbf{CP}|\mathbf{v}^*)$ as

$$\begin{aligned} P(\mathbf{CP}|\mathbf{v}^*) &= \prod_{e \in \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}} (1-q_e) \sum_{E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v}^*,\mathbf{CP}}} \prod_{e \in E_{\mathbf{CP}}} q_e \prod_{e \in \overline{E}_{\mathbf{CP},\mathbf{v}^*} \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}} (1-q_e) \\ &= \prod_{e \in \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}} (1-q_e) \sum_{E_{\mathbf{CP}} \in \mathbf{E}_{on,\mathbf{v},\mathbf{CP}}} \prod_{e \in E_{\mathbf{CP}}} q_e \prod_{e \in \overline{E}_{\mathbf{CP},\mathbf{v}} \setminus \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}} (1-q_e), \end{aligned} \tag{35}$$

and so, all terms except for the initial term in the ratio $P(\mathbf{CP}|\mathbf{v}^*)/P(\mathbf{CP}|\mathbf{v})$ cancel out, and we obtain the same ratio derived in Barbu and Zhu (2005):

$$\frac{P(\mathbf{CP}|\mathbf{v}^*)}{P(\mathbf{CP}|\mathbf{v})} = \frac{\prod_{e \in \overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}} (1-q_e)}{\prod_{e \in \overline{E}_{\mathbf{V_{CP}},\mathbf{v}}} (1-q_e)}. \tag{36}$$

In the special case that edges are turned on with equal probability q, this ratio can be computed by counting the number of edges in the set of Swendsen-Wang cuts.

$$\frac{P(\mathbf{CP}|\mathbf{v}^*)}{P(\mathbf{CP}|\mathbf{v})} = \min\left(1, q^{|\overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}| - |\overline{E}_{\mathbf{V_{CP}},\mathbf{v}}|}\right). \tag{37}$$

A.2.2 Simplification of $\#adj(V_{CP}, \mathbf{v})$

We show that, for all $V_{CP} \in \mathbf{V_{CP}}$:

$$\#adj(V_{CP}, \mathbf{v}) = \#adj(V_{CP}, \mathbf{v}^*), \tag{38}$$

which implies that:

$$\prod_{V_{CP}\in\mathbf{V_{CP}}} P(s^*|V_{CP},\mathbf{v}) = \prod_{V_{CP}\in\mathbf{V_{CP}}} P(s|V_{CP},\mathbf{v}^*), \tag{39}$$

where s^* and s are chosen to form $\mathbf{v}^*$ from $\mathbf{v}$ and $\mathbf{v}$ from $\mathbf{v}^*$ respectively.

Without loss of generality, suppose V_{CP} is adjacent to blocks $V_1, V_2, \ldots, V_m \in \mathbf{v}$, where $m = \#adj(V_{CP},\mathbf{v})$, suppose that vertices in V_{CP} are assigned to V_m in the formation of partition $\mathbf{v}^*$, and suppose V_{CP} is contained in block $V_{m+1} \in \mathbf{v}$. The valid partition check in Step 4 ensures that $V_{CP} \neq V_{m+1}$. There must exist pairs of vertices that are adjacent in the graph G, $(\{i_1\},\{j_1\}),\ldots,(\{i_m\},\{j_m\})$, where $\{i_k\} \in V_{CP}, \{j_k\} \in V_k$. Moreover, there exists $\{i_{m+1}\} \in V_{CP}, \{j_{m+1}\} \in V_{m+1} \setminus V_{CP}$ that are adjacent in the graph $G(\mathbf{v})$. The vertices $\{i_k\}$ are not necessarily unique.

Since only vertices in connected components in $\mathbf{V_{CP}}$ change assignment when transitioning from $\mathbf{v}$ to $\mathbf{v}^*$, and since $\mathbf{V_{CP}}$ does not contain adjacent connected components, all vertices adjacent to V_{CP} do not change block assignment. Hence, all vertices adjacent to V_{CP} must belong to one of $V_1^*,\ldots,V_{m+1}^*$, in $\mathbf{v}^*$. Moreover, since $\{j_1\},\ldots,\{j_{m+1}\}$ do not change block assignment, V_{CP} is adjacent to each of $V_1^*,\ldots,V_{m-1}^*$ and also V_{m+1}^*. Hence, $\#adj(V_{CP},\mathbf{v}^*) = m = \#adj(V_{CP},\mathbf{v})$.

A.2.3 Finding the Joint Acceptance Probability

From (39) and (36) and noting that $P(R|\mathbf{CP},\mathbf{v}) = P(R)$ by assumption, it follows that we can write the ratio (24) as:

$$\begin{aligned}\alpha^*(\mathbf{v}\to\mathbf{v}^*,\mathbf{CP}) &= \frac{P(\mathbf{CP}|\mathbf{v}^*)P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v}^*,R)\prod_{V_{CP}\in\mathbf{V_{CP}}}P(s|V_{CP},\mathbf{v}^*)}{P(\mathbf{CP}|\mathbf{v})P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v},R)\prod_{V_{CP}\in\mathbf{V_{CP}}}P(s^*|V_{CP},\mathbf{v})}\frac{g(\mathbf{v}^*)}{g(\mathbf{v})} \\ &= \frac{P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v}^*,R)\prod_{e\in\overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}}(1-q_e)}{P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v},R)\prod_{e\in\overline{E}_{\mathbf{V_{CP}},\mathbf{v}}}(1-q_e)}\frac{g(\mathbf{v}^*)}{g(\mathbf{v})},\end{aligned} \tag{40}$$

and so, the joint acceptance probability defined in (25) simplifies to

$$\alpha(\mathbf{v}\to\mathbf{v}^*,\mathbf{CP}) = \min\left\{1, \frac{P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v}^*,R)\prod_{e\in\overline{E}_{\mathbf{V_{CP}},\mathbf{v}^*}}(1-q_e)}{P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v},R)\prod_{e\in\overline{E}_{\mathbf{V_{CP}},\mathbf{v}}}(1-q_e)}\frac{g(\mathbf{v}^*)}{g(\mathbf{v})}\right\}. \tag{41}$$

Hence, as claimed, the computability of $\alpha(\mathbf{v}\to\mathbf{v}^*,\mathbf{CP})$ depends on the computability of $P(\mathbf{V_{CP}}|\mathbf{CP},\mathbf{v},R)$.

A.2.4 Computation of $P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R)$

For a set of connected components $\mathbf{CP}$ and partition $\mathbf{v}$, define $\mathbf{B}(\mathbf{CP}, \mathbf{v})$ as the set of connected components in $\mathbf{CP}$ on the boundary of $\mathbf{v}$. Furthermore, define $\mathbf{B}^{\dagger}(\mathbf{CP}, \mathbf{v}) \subset \mathbf{B}(\mathbf{CP}, \mathbf{v})$ as the set of boundary connected components that do not *shatter* its district when removed—that is, its removal from its current district must not make the district non-contiguous.

We now discuss exact and approximate computation of $P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R)$ under various scenarios. We make the assumption that, given a number of connected components R to be selected from $\mathbf{CP}$, every subset of R components satisfying the conditions given in Step 4 are equally likely to comprise $\mathbf{V_{CP}}$.

- **Single swaps:** $R = 1$.

  A connected component for $\mathbf{V_{CP}}$ can be selected one either by obtaining all connected components in $\mathbf{B}^{\dagger}(\mathbf{CP}, \mathbf{v})$ and randomly selecting a connected component in this set, or by randomly selecting a component in $\mathbf{B}(\mathbf{CP}, \mathbf{v})$ and resampling from this set if removing the component shatters its district. In either case:

  $$P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R) = \frac{1}{|\mathbf{B}^{\dagger}(\mathbf{CP}, \mathbf{v})|}. \tag{42}$$

  Checking whether a connected component can shatter its district takes $O(|V|)$ time if the adjacency graph is sparse (Cormen *et al.*, 2001). However, if the number connected components that can shatter a district is small relative to the number of boundary connected components, we can approximate this probability simply by counting the number of boundary connected components. This may be a reasonable assumption if districts do not have narrow strings of precincts. By selecting $\mathbf{V_{CP}}$ through the "sample and check" method described above, this approximation can greatly reduce the amount of computation required.

  $$P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R) \approx \frac{1}{|\mathbf{B}(\mathbf{CP}, \mathbf{v})|}. \tag{43}$$

- **Multiple swaps:** $R > 1$.

  Obtaining a uniformly random sample from $\mathbf{B}^{\dagger}(\mathbf{CP}, \mathbf{v})$ is made computationally difficult by the condition that $\mathbf{V_{CP}}$ contains no adjacent connected components. When possible, such a sample can be obtained by enumerating all samples of size R that meet the conditions given in Step 3 of the algorithm, and randomly selecting one of these samples. Then, $P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R)$ is the inverse of the number of samples of size R.

  An approximately uniform sample can be obtained by sequentially selecting non-adjacent connected components, and for each component selected, checking whether or not that component will shatter its district when removed. Again, computing $P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R)$ exactly under this sampling scheme is computationally hard.

  We can approximate the probability $P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R)$ a few ways. For any connected component $V \in \mathbf{B}^{\dagger}(\mathbf{CP}, \mathbf{v})$, consider the number of connected components in $\mathbf{B}^{\dagger}(\mathbf{CP}, \mathbf{v})$ that are adjacent to V, and let d denote its maximum across all connected components V. If $dR \ll \left|\mathbf{B}^{\dagger}(\mathbf{CP}, \mathbf{v})\right|$, then sampling of connected components can be approximated by with-replacement sampling; the probability of repeating a connected component or one of its neighbors in the sample is almost zero. Hence, in this case,

$$P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R) \approx R! \left(\frac{1}{|\mathbf{B}^{\dagger}(\mathbf{CP}, \mathbf{v})|}\right)^{R}. \tag{44}$$

  If, in addition, the number connected components that can shatter a district is small in relation to the number of boundary connected components, then we can approximate this probability simply by counting the number of boundary connected components. Again, this approximation can greatly reduce the amount of computation required.

$$P(\mathbf{V_{CP}}|\mathbf{CP}, \mathbf{v}, R) \approx R! \left(\frac{1}{|\mathbf{B}(\mathbf{CP}, \mathbf{v})|}\right)^{R}. \tag{45}$$

  Substituting this approximation into the expression given in (41), assuming $q_e = q$ for all e, and choosing the uniform distribution as our target density, we obtain the approximate

Metropolis-Hastings ratio shown in (5).

A.2.5 Empirical Validation of Approximate Ratio

To show how the performance of the basic algorithm using the approximate acceptance ratio quickly approaches the performance of the exact acceptance ratio, we conduct simulation studies on a set of lattices. Results are shown below in Figure 9. We generate three lattices of dimensions 3x2, 3x3, and 4x3, where populations for each lattice node are distributed $\mathcal{TN}(100, 20, 0)$. We specify a distinct sub-group in each lattice node, whose population is distributed $\mathcal{N}(50, 10)$. We then specify every valid partition of these lattices into two contiguous districts and calculate the dissimilarity index of each districting plan, in order to compare our algorithm's performance against the true distribution.

The results of our simulation studies are shown in Figure 9. We expect that the approximate acceptance ratio should have the most difficulty recovering the true distribution on small maps, and less trouble as the size of the map increases. Using our exact and approximate acceptance ratios, we run our algorithm 100,000 times on each lattice and use QQ plots to see whether the draws approximate a uniform distribution. As expected, the exact acceptance ratio (top row) recovers the true distribution of each map. While the approximate acceptance ratio struggles on the smallest map, we find that its performance quickly approaches that of the approximate acceptance ratio, and that the simulated distributions under the two different acceptance ratios are both indistinguishable from the true distribution on the 4x3 lattice.

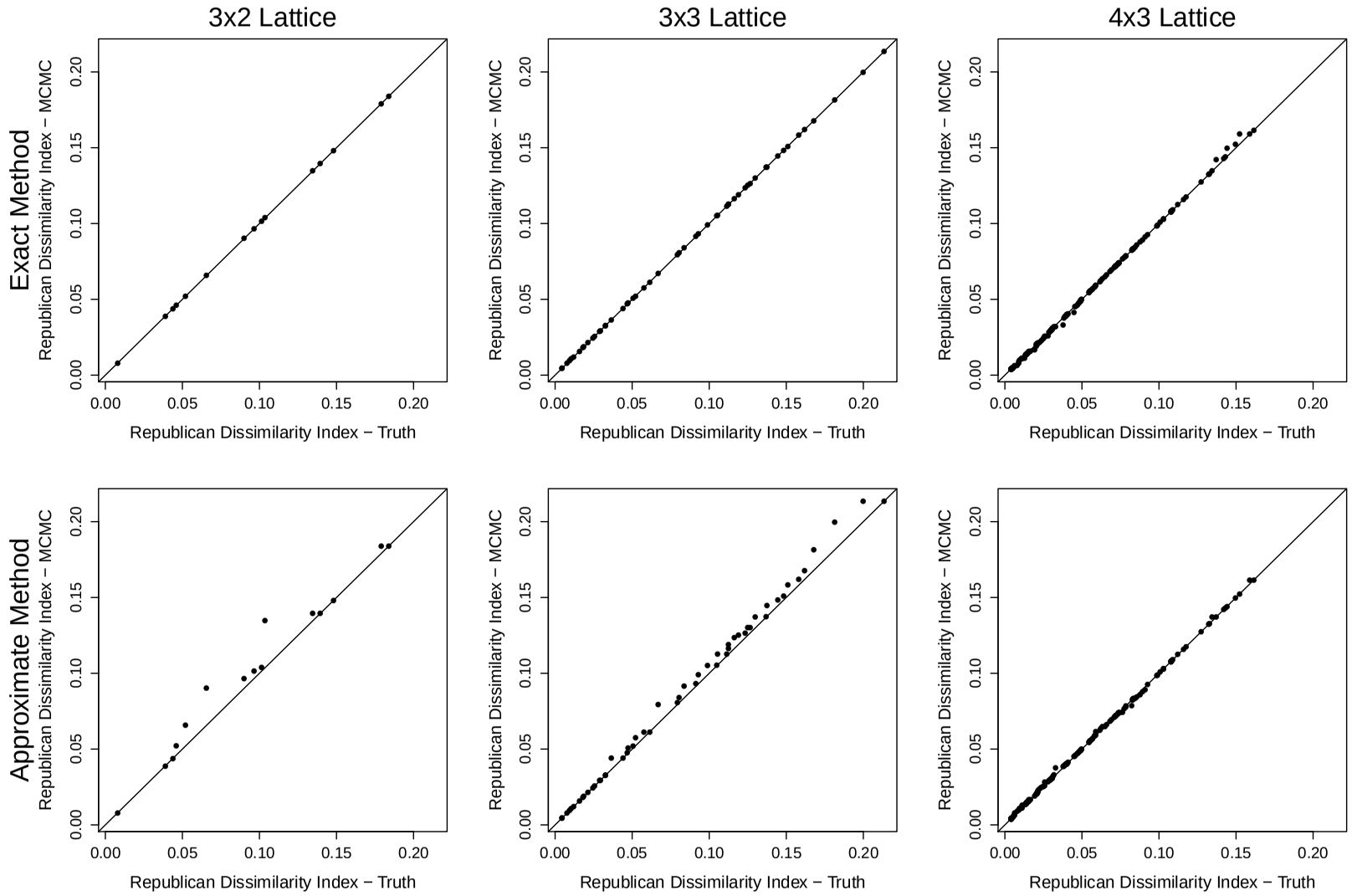

Figure 9: A Small-scale Validation Study using Lattices. The underlying data are a series of lattices with simulated area and group populations. The plots in the first row show that the exact acceptance ratio approximates nearly exactly each lattice. The plots in the bottom row show that while the approximate acceptance ratio struggles to simulate from the target distribution on small maps, the issue disappears when the size of the map increases. The results for each algorithm is based on a single chain of 100,000 draws.