Online Supplementary Information for "Automated Redistricting Simulation Using Markov Chain Monte Carlo."

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

March 2, 2020

S1 Proof of Theorem 1

We first provide some definitions that will be used throughout this proof. Let $\mathbf{E}_{on}(\mathbf{CP}, \pi)$ be the collection of turned-on edge sets $E_0 \subseteq E(\pi)$ such that the connected components $\mathbf{CP}$ are formed, and define the complementary set of turned-off edges $\overline{E}_0(\pi) \equiv E(\pi) \setminus E_0$. Also let $\Omega(\mathbf{V_{CP}}, \pi)$ be the collection of connected components sets $\mathbf{CP}$ that can be formed by "turning on" edges in $E(\pi)$ such that $\mathbf{V_{CP}} \subseteq \mathbf{CP}$.

S1.1 Lemmas

Next, we prove necessary lemmas. The first lemma characterizes the difference between the edge sets $E(\pi)$ and $E(\pi')$ in terms of the Swendsen-Wang cuts, while the second lemma establishes equivalence between several important sets needed for simplifying the acceptance probability.

LEMMA 1 *Suppose that π, π', and $\mathbf{CP}$ satisfy $q(\pi', \mathbf{CP} \mid \pi) > 0$, and that π' is proposed from π by reclassifying nodes in a set of non-adjacent connected components $\mathbf{V_{CP}} \subseteq \mathbf{CP}$. Then*

$$E(\pi) \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi) \;=\; E(\pi') \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi'), \tag{S1}$$

Proof Consider any edge $i \sim j \in E(\pi) \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi)$. Since $i \sim j \in E(\pi)$ and $i \sim j \notin \mathcal{C}(\mathbf{V_{CP}}, \pi)$, and since $\mathbf{V_{CP}}$ consists of non-adjacent components, then either i and j belong to some district $V_\ell \in \pi$, with $i, j \notin \mathbf{V_{CP}}$, or i and j belong to some $C_0 \in \mathbf{V_{CP}}$. In the former case, i, j do not change district

*Affiliated Researcher, Institute for Quantitative Social Science, Harvard University, Cambridge, MA 02138. Email: benfifield@gmail.com, URL: https://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 Government and Department of Statistics, Harvard University. 1737 Cambridge Street, Institute for Quantitative Social Science, Cambridge MA 02138. Email: imai@harvard.edu, URL: https://imai.fas.harvard.edu

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

assignment, and hence, belong to the same district $V_\ell \in \pi'$. In the latter case, all nodes in C_0 are reassigned to the same district $V_{\ell'} \in \pi'$. In both cases, $i \sim j \in E(\pi') \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi')$, and therefore, $E(\pi) \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi) \subseteq E(\pi') \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi')$.

Repeating this same argument starting from any edge in $E(\pi') \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi')$ yields $E(\pi') \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi') \subseteq E(\pi) \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi)$, and so, we obtain the desired equality S1. □

LEMMA 2 *Suppose that π, π', and $\mathbf{CP}$ satisfy $q(\pi', \mathbf{CP} \mid \pi) > 0$, and that π' is proposed from π by reclassifying nodes in a set of non-adjacent connected components $\mathbf{V_{CP}} \subseteq \mathbf{CP}$. Then*

$$\Omega(\mathbf{V_{CP}}, \pi) \;=\; \Omega(\mathbf{V_{CP}}, \pi'). \tag{S2}$$

Furthermore, for all $\mathbf{CP} \in \Omega(\mathbf{V_{CP}}, \pi)$,

$$\mathbf{E}_{on}(\mathbf{CP}, \pi) \;=\; \mathbf{E}_{on}(\mathbf{CP}, \pi'), \tag{S3}$$

and for all $E_0 \in \mathbf{E}_{on}(\mathbf{CP}, \pi)$,

$$\overline{E}_0(\pi) \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi) \;=\; \overline{E}_0(\pi') \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi'). \tag{S4}$$

Proof Since all $\mathbf{CP} \in \Omega(\mathbf{V_{CP}}, \pi)$ must at least have edges $\mathcal{C}(\mathbf{V_{CP}}, \pi)$ turned off by definition, all $\mathbf{CP}$ must be formed by turning on edges in $E(\pi) \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi)$. All three equations then follow from Lemma 1. □

S1.2 Simplifying the Acceptance Probability

We now simplify the acceptance probability. Suppose that π, π', and $\mathbf{CP}$ satisfy $q(\pi', \mathbf{CP} \mid \pi) > 0$, and that π and π' differ in their district assignment of a set of non-adjacent connected components $\mathbf{V_{CP}} \subseteq \mathbf{CP}$, with each $C_i \in \mathbf{V_{CP}}$ having district membership $C_i \subset V_{\ell_i}$ in π, and $C_i \subset V_{\ell'_i}$ in π'. Since $\mathbf{V_{CP}}$ is restricted to consist of non-adjacent connected components, there is only one path moving between partition π and π': selecting and changing the assignment of each $C_i \in \mathbf{V_{CP}}$. Therefore, the ratio of proposal densities is

$$\frac{q(\pi, \mathbf{CP} \mid \pi')}{q(\pi', \mathbf{CP} \mid \pi)} \;=\; \frac{P(\mathbf{CP} \mid \pi') P(R \mid \mathbf{CP}, \pi') P(\mathbf{V_{CP}} \mid \mathbf{CP}, R, \pi')}{P(\mathbf{CP} \mid \pi) P(R \mid \mathbf{CP}, \pi) P(\mathbf{V_{CP}} \mid \mathbf{CP}, R, \pi)} \frac{\prod_{C_i \in \mathbf{V_{CP}}} P(\ell_i \mid C_i, \pi')}{\prod_{C_i \in \mathbf{V_{CP}}} P(\ell'_i \mid C_i, \pi)}, \tag{S5}$$

where $P(\mathbf{CP}|\pi)$ is the probability that connected components $\mathbf{CP} \in \Omega(\mathbf{V_{CP}}, \pi)$ are formed by turning on edges in $E(\pi)$ in Step 1, $P(R \mid \mathbf{CP}, \pi)$ is the probability of selecting R boundary connected components in Step 3(a), $P(\mathbf{V_{CP}} \mid \mathbf{CP}, \pi, R)$ is the probability of selecting $\mathbf{V_{CP}}$ in Step 3(b), and $P(\ell'_i \mid C_i, \pi)$ is the probability of assigning component $C_i \in \mathbf{V_{CP}}$, with $C_i \subset V_{\ell_i}$, to a different bordering district $V_{\ell'_i} \in \pi$ in Step 4.

S1.2.1 Simplification of $P(\mathbf{CP} \mid \pi)$

Let $q_e = q$ denote the probability of turning on edge $e \in E(\pi)$ in Step 1. Since $\mathbf{CP}$ is formed following the same procedure as Barbu and Zhu (2005), we obtain the same probability,

$$P(\mathbf{CP} \mid \pi) \;=\; \prod_{e \in \mathcal{C}(\mathbf{V_{CP}}, \pi)} (1 - q) \sum_{E_0 \in \mathbf{E}_{on}(\mathbf{CP}, \pi)} \prod_{e \in E_0} q \prod_{e \in \overline{E}_0(\pi) \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi)} (1 - q). \tag{S6}$$

From Lemma 2, we can write $P(\mathbf{CP} \mid \pi')$ as

$$\begin{aligned} P(\mathbf{CP} \mid \pi') &= \prod_{e \in \mathcal{C}(\mathbf{V_{CP}}, \pi')} (1-q) \sum_{E_0 \in \mathbf{E}_{on}(\mathbf{CP}, \pi')} \prod_{e \in E_0} q \prod_{e \in \overline{E}_0(\pi') \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi')} (1-q) \\ &= \prod_{e \in \mathcal{C}(\mathbf{V_{CP}}, \pi')} (1-q) \sum_{E_0 \in \mathbf{E}_{on}(\mathbf{CP}, \pi)} \prod_{e \in E_0} q \prod_{e \in \overline{E}_0(\pi) \setminus \mathcal{C}(\mathbf{V_{CP}}, \pi)} (1-q). \end{aligned} \tag{S7}$$

Comparing (S7) with (S6), we obtain the same ratio derived in Barbu and Zhu (2005):

$$\frac{P(\mathbf{CP} \mid \pi')}{P(\mathbf{CP} \mid \pi)} = \frac{\prod_{e \in \mathcal{C}(\mathbf{V_{CP}}, \pi')} (1-q)}{\prod_{e \in \mathcal{C}(\mathbf{V_{CP}}, \pi)} (1-q)} = \frac{(1-q)^{|\mathcal{C}(\mathbf{V_{CP}}, \pi')|}}{(1-q)^{|\mathcal{C}(\mathbf{V_{CP}}, \pi')|}}. \tag{S8}$$

S1.2.2 Simplification of $P(R \mid \mathbf{CP}, \pi)$

Let $P_R(\cdot)$ denote the distribution for sampling R with cumulative distribution function $F(\cdot)$. Since R is generated by independently sampling from P_R until $R \leq |B(\mathbf{CP}, \pi)|$ is drawn, we have

$$\begin{aligned} P(R \mid \mathbf{CP}, \pi) &= \left[\sum_{k=0}^{\infty} \underbrace{P_R(R > |B(\mathbf{CP}, \pi)|)^k}_{k \text{ invalid draws}} \right] \underbrace{P_R(R, R \leq |B(\mathbf{CP}, \pi)|)}_{\text{valid sample on } k+1 \text{ draw}} \\ &= \mathbf{1}\{R \leq |B(\mathbf{CP}, \pi)|\} \frac{P_R(R)}{F(|B(\mathbf{CP}, \pi)|)}. \end{aligned} \tag{S9}$$

Noting that in any valid move $\pi \to \pi'$, $R \leq |B(\mathbf{CP}, \pi)|$, and assuming that $R \leq |B(\mathbf{CP}, \pi')|$, we obtain the ratio

$$\frac{P(R \mid \mathbf{CP}, \pi')}{P(R \mid \mathbf{CP}, \pi)} = \frac{F(|B(\mathbf{CP}, \pi)|)}{F(|B(\mathbf{CP}, \pi')|)}. \tag{S10}$$

We empirically observe this assumption to always hold in our application setting. In general, however, when a move is proposed with $|B(\mathbf{CP}, \pi')| < R \leq |B(\mathbf{CP}, \pi)|$, the proposed π' is rejected with probability 1 in order to preserve reversibility of the Markov chain.

S1.2.3 Simplification of $P(\ell_i' \mid C_i, \pi)$

Recall that our algorithm restricts $\mathbf{V_{CP}}$ to consist only of nonadjacent connected components on the boundary. Since only vertices in connected components $\mathbf{V_{CP}}$ change assignment when transitioning from π to π', it follows that all vertices adjacent to each $C_i \in \mathbf{V_{CP}}$ do not change district assignment, and hence, each C_i retains adjacency to the same districts in both π and π'. In our application C_i is assigned uniformly at random to one of its neighboring districts, and therefore,

$$\frac{\prod_{C_i \in \mathbf{V_{CP}}} P(\ell_i \mid C_i, \pi')}{\prod_{C_i \in \mathbf{V_{CP}}} P(\ell_i' \mid C_i, \pi)} = 1. \tag{S11}$$

S1.2.4 Joint Acceptance Ratio

Combining (S8), (S10), and (S11), and noting the definition for the acceptance ratio in (3),we obtain the result given in Theorem 1,

$$\alpha(\pi', \mathbf{CP} \mid \pi) = \min\left(1, \frac{P\left(\mathbf{V_{CP}} \mid \mathbf{CP}, R, \pi'\right) F(|B(\mathbf{CP}, \pi)|) \, (1-q)^{|\mathcal{C}(\pi', \mathbf{V_{CP}})|}}{P\left(\mathbf{V_{CP}} \mid \mathbf{CP}, R, \pi\right) F(|B(\mathbf{CP}, \pi')|) \, (1-q)^{|\mathcal{C}(\pi, \mathbf{V_{CP}})|}} \cdot \frac{g(\pi')}{g(\pi)}\right). \tag{S12}$$

Note that due to the $P(\mathbf{V_{CP}} \mid \mathbf{CP}, R, \pi)$ and $F(\cdot)$ terms, marginalizing out $\mathbf{CP}$ in this ratio, as was done in Barbu and Zhu (2005), would require summing over a combinatorial number of $\mathbf{CP} \in \Omega(\mathbf{V_{CP}}, \pi)$, and hence is computationally intractable in our application. □

S2 Proof of Theorem 2

In proving Theorem 2, it is sufficient to show that the Markov chain described by Algorithm 1 satisfies the two conditions: detailed balance and ergodicity.

The Metropolis-Hastings step in Algorithm 1 ensures that the following detailed balance condition holds:

$$f(\pi)P(\pi', \mathbf{CP} \mid \pi) \;=\; f(\pi')P(\pi, \mathbf{CP} \mid \pi'), \tag{S13}$$

where $P(\pi', \mathbf{CP} \mid \pi) = q(\pi', \mathbf{CP} \mid \pi)\alpha(\pi', \mathbf{CP} \mid \pi)$ is the unmarginalized transition probability. However, in order to show the existence of the stationary distribution $f(\pi)$, we need the detailed balance condition

$$f(\pi)P(\pi' \mid \pi) = f(\pi')P(\pi \mid \pi'), \tag{S14}$$

where

$$P(\pi' \mid \pi) = \sum_{\mathbf{CP} \in \Omega(\mathbf{V_{CP}}, \pi)} P(\pi', \mathbf{CP} \mid \pi) \tag{S15}$$

is the transition probability of our Markov chain with dependence on $\mathbf{CP}$ marginalized out.

Since $\Omega(\mathbf{V_{CP}}, \pi) = \Omega(\mathbf{V_{CP}}, \pi')$ by Lemma 2, marginalizing out $\mathbf{CP}$ from the left hand side of equation (S13) gives

$$\begin{aligned} f(\pi)P(\pi' \mid \pi) \;&=\; \sum_{\mathbf{CP} \in \Omega(\mathbf{V_{CP}}, \pi)} f(\pi)P(\pi', \mathbf{CP} \mid \pi) \\ &= \sum_{\mathbf{CP} \in \Omega(\mathbf{V_{CP}}, \pi')} f(\pi')P(\pi, \mathbf{CP} \mid \pi') \;=\; f(\pi')P(\pi \mid \pi'), \end{aligned} \tag{S16}$$

and hence, equation (S14) holds.

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, and hence, the Markov chain is positive recurrent. Similarly, since the assumption $0 < \alpha(\pi', \mathbf{CP} \mid \pi) < 1$ establishes π as an aperiodic state, then by irreducibility, all states must be aperiodic, and hence, the Markov chain is aperiodic. □

S3 Enumeration Algorithm

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 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.

S4 Simulated Tempering

In this section, we present the simulated tempering algorithm as well as the results of simulation and empirical studies.

S4.1 The Algorithm

Recall that we want to draw from the distribution given in equation (11).We initialize a sequence of *inverse temperatures* $\{\beta^{(i)}\}_{i=0}^{r-1}$ 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 $1/g_{\beta^{(0)}}(\pi)$.

Specifically, we perform simulated tempering in two steps. First, we run an iteration of Algorithm 1 using the modified acceptance probability with $\beta = \beta^{(i)}$. 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 S1 (SAMPLING CONTIGUOUS PLANS WITH SIMULATED TEMPERING AND SOFT CONSTRAINT) *Given the initial valid partition π_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 $t + 1$,*

Step 1 (Run the basic algorithm with the modified acceptance probability): *Using the current partition π_t and the current temperature $\beta_t = \beta^{(\kappa_t)}$, obtain a valid partition π_{t+1} by running one iteration of Algorithm 1 with the acceptance probability given in equation (8).*

Step 2 (Choose a candidate temperature): *Generate $u \sim \mathcal{U}(0, 1)$. We propose*

$$\kappa' = \begin{cases} \kappa_t - 1 & \text{if } u \leq q(\kappa_t, \kappa_t - 1) \\ \kappa_t + 1 & \text{otherwise} \end{cases}, \tag{S17}$$

where $q(\kappa_t, \kappa_t - 1) = 1/2$ when $1 \leq \kappa_{t-1} \leq r - 2$, $q(r - 1, r - 2) = q(0, 1) = 1$, and $q(r - 1, r) = q(0, -1) = 0$.

Step 3 (Accept or reject the candidate temperature): *Generate $v \sim \mathcal{U}(0, 1)$. Set*

$$\kappa_t = \begin{cases} \kappa' & \text{if } v \leq \gamma(\kappa' \mid \kappa_t) \\ \kappa_t & \text{otherwise} \end{cases}, \tag{S18}$$

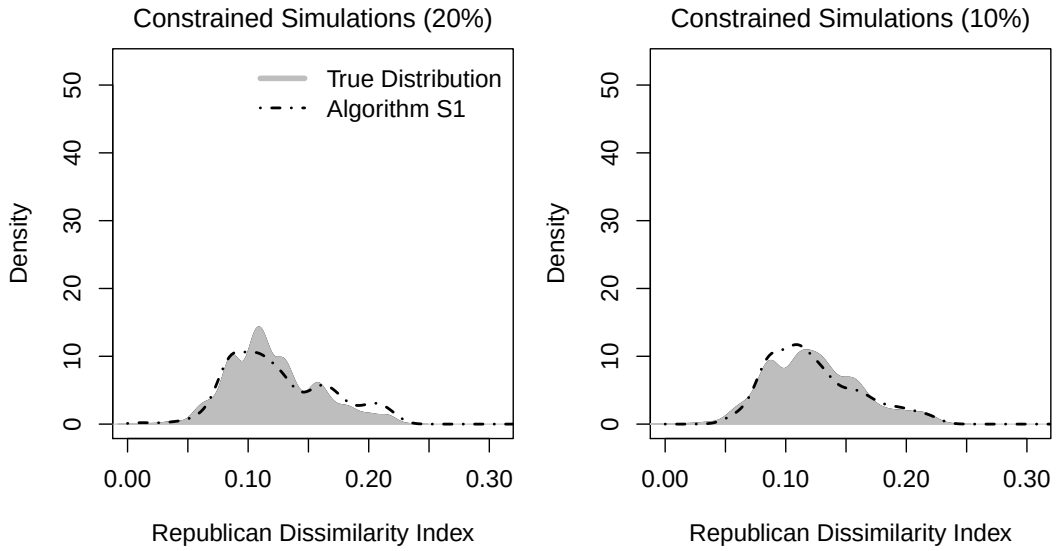

Figure S1: A Small-scale Validation Study with Three Districts for the Simulated Tempering. The underlying data is the 25 precinct set shown in the left plot of Figure 1. The plots show the performance of the algorithm using the Republican dissimilarity index measure. Both plots show that the proposed simulated tempering algorithm (Algorithm S1; black dot-dashed line) approximates well the true population distribution (grey histograms) when a population constraint is imposed. The results for the algorithm is based on a single chain of 10,000 draws.

where

$$\gamma(\kappa' \mid \kappa_t) \;=\; \min\left(1, \frac{g_{\beta(\kappa')}(\pi_{t+1}) q(\kappa', \kappa_t) w_{\kappa'}}{g_{\beta(\kappa_t)}(\pi_{t+1}) q(\kappa_t, \kappa') w_{\kappa_t}}\right) \tag{S19}$$

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

S4.2 Simulation Results

Here, we present simulation results for the simulated tempering algorithm introduced above using the small-scale validation study from Section 3.1. The results presented in Figure S1 are based on a single chain of 10,000 draws, using Algorithm S1. As in the parallel tempering simulations show in Figure 3, we chose a target temperature of $\beta^{(0)} = 5.4$ for the population deviation of 20%, and a target temperature of $\beta^{(0)} = 9$ for the population deviation of 10%. Results are then reweighted back to the uniform distribution using inverse probability reweighting. In both cases, we use $\beta^{(r-1)} = 0$, and we generate starting seed maps for each chain using the Random Seed and Grow procedure presented in Algorithm 3. The results show that the simulated tempering algorithm is able to successfully recover the target distribution under both population constraints.

S5 The Failure of the Proposed Algorithm with a Hard Constraint

We explore the reason why the proposed algorithm with a hard constraint (Algorithm 1.1 fails under the 10% equal population constraint as seen in the small-scale validation study of Section 3.1 (see the right plot of Figure 3). We empirically demonstrate that this failure is not due to running the algorithm for too few iterations. Rather, the failure comes from the fact that in the presence of the stricter population constraint, the proposed algorithm with a hard constraint cannot traverse from one valid partition to

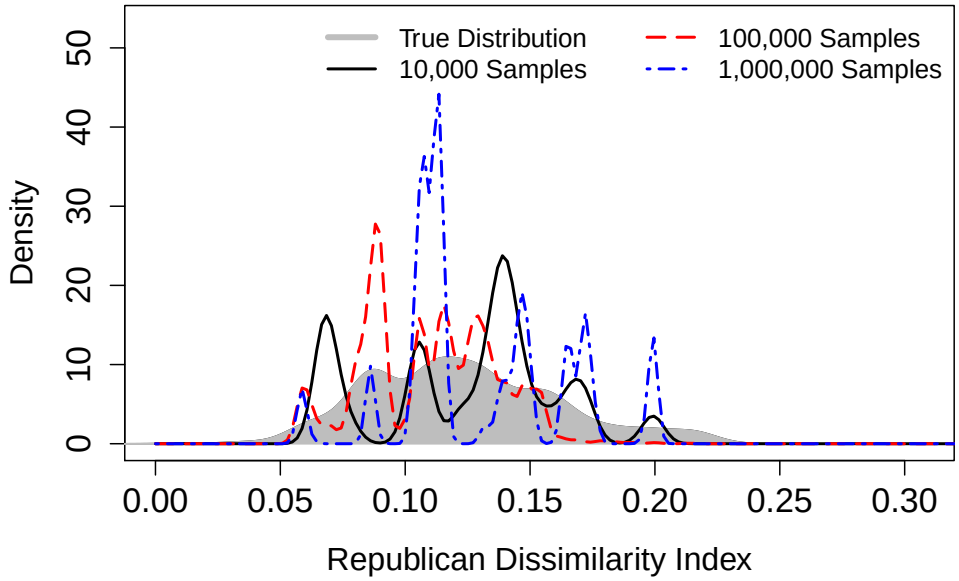

Figure S2: The Failure of the Proposed Algorithm with a Hard Constraint. The plot is based on the same setting as the one used for the right plot of Figure 3. Increasing the number of draws does not improve the performance of Algorithm 1.1, the proposed algorithm with a hard constraint.

another, due to invalid partitions that separate them. Figure S2 shows that running a longer chain does not help at all. Indeed, the longest chain based on one million iterations performs the worst and has difficulty exploring some parts of the target distribution.

S6 New Hampshire

S6.1 Setup

In this appendix, we apply the proposed algorithms to the 2008 election data in New Hampshire. Unlike Pennsylvania, which we analyzed in Section 3.2, New Hampshire represents a small redistricting problem with only two districts. Therefore, we conduct a "global exploration" by setting the target distribution to the uniform distribution on all valid redistricting plans under a population parity constraint of 1%.

New Hampshire has two congressional districts and 327 precincts, shown in Figure S3. Under the 2008 districting 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%. We apply the proposed algorithm with a hard constraint (Algorithm 1.1) and with parallel tempering (Algorithm 2). The target population consists of all redistricting plans with contiguous districts and a maximum of 1% deviation from population parity.

A total of 10 chains are run until 500,000 draws are obtained for each chain. For starting maps, we use independent draws from the standard algorithm (Algorithm 3 as implemented in the `BARD` package). After some preliminary analysis, we decided to allow $\beta^{(i)}$ to take values between 0 and 27, using geometric spacing. In this preliminary analysis, we ran the algorithm using a variety of values of $\beta^{(r-1)}$ ranging from 10 to 50, and selected $\beta^{(r-1)} = 27$ after analyzing acceptance probabilities, the number of samples drawn with sufficiently low population parity, and the ease with which the algorithm transferred

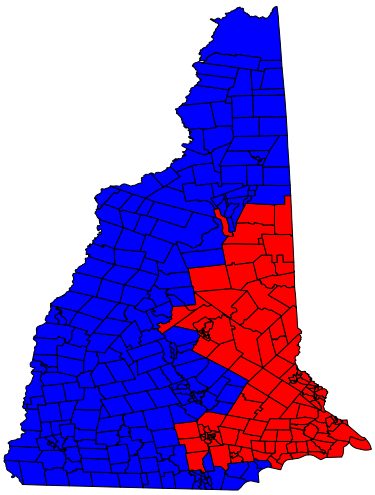

Figure S3: New Hampshire's 2008 Redistricting Plan. New Hampshire has 327 precincts that are divided into 2 congressional districts.

between values of the $\beta^{(i)}$ ladder. As in the small-scale verification study, we only use draws taken under the target temperature, and then reweight according to the importance weights $1/g_{\beta^{(i)}(\pi)}$ before selecting all remaining draws that fall within the target parity deviation of 1%. Inference is based on a total of 64,800 draws, which is the lowest number of draws across the two algorithms that satisfy the population constraint.

S6.2 Results of Global Simulations

Figure S4 presents the convergence diagnostics. The figure shows the autocorrelation plots (left column), the trace plots (middle column), and the Gelman-Rubin potential scale reduction factors (right column) for the basic algorithm (Algorithm 1.1; top panel) and the parallel tempering algorithm (Algorithm 2; bottom panel). We use the logit transformed Republican dissimilarity index, defined in equation (14), for all diagnostics. Here, the parallel tempering algorithm outperforms the basic algorithm. While the autocorrelations are slightly lower to those of the proposed algorithm with a hard constraint, 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. Although not shown here, we conduct the same analysis for the simulated tempering algorithm (Algorithm S1), which also successfully converges to the target distribution.

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.

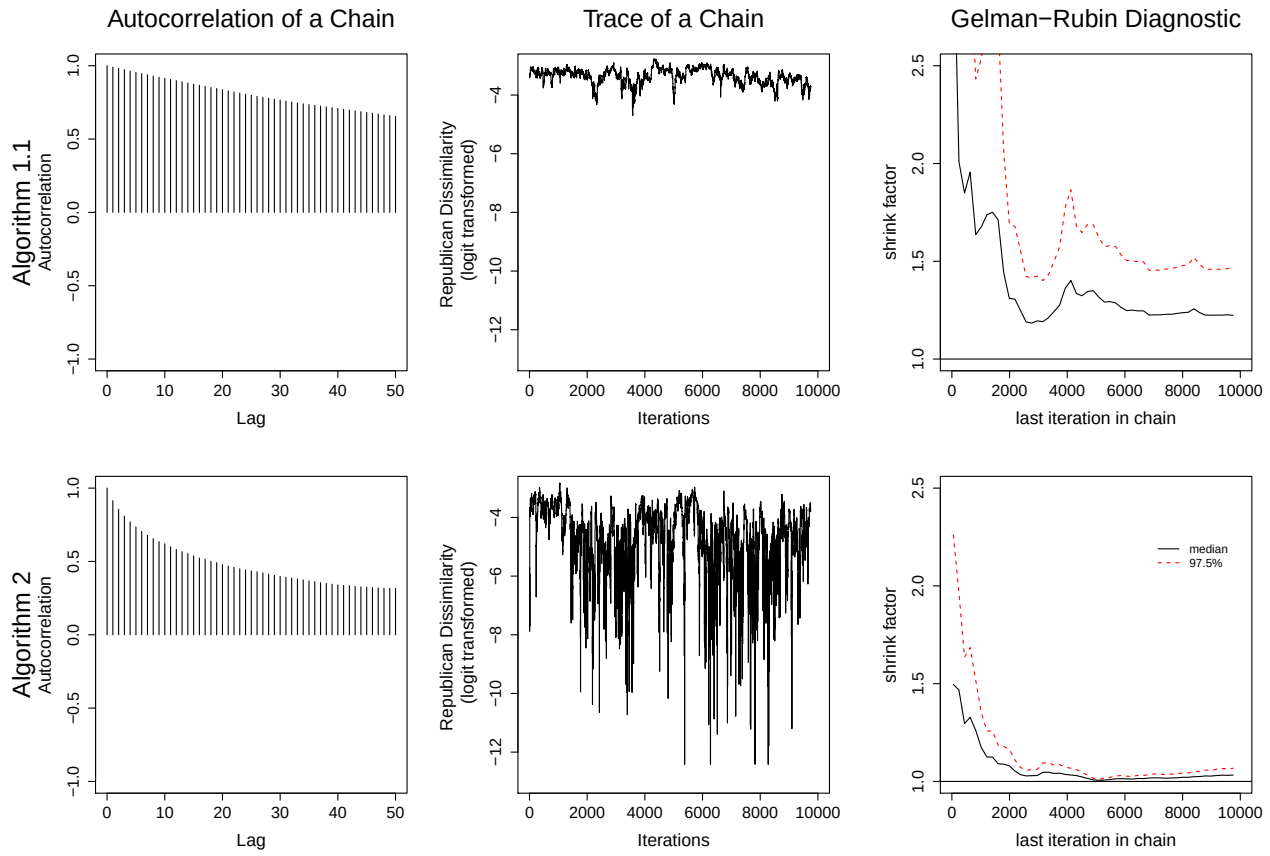

Figure S4: Convergence Diagnostics of the Proposed Algorithms for the 2008 New Hampshire Redistricting Data. The proposed basic algorithm with hard constraint (Algorithm 1.1; top panel) and the parallel tempering algorithm (Algorithm 2; bottom panel) are applied to the New Hampshire data with 327 precincts 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. 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.

S6.3 Single versus Multiple Swaps

We also examine how the single ($R = 1$) and multiple ($R \geq 1$) swaps affects the convergence behavior of the proposed algorithm using the New Hampshire data, under a realistic population parity constraint of 1%. For the multiple component swaps, we draw R according to the truncated Poisson distribution as described in Section 2.2 and set the mean of the truncated Poisson distribution to 32, so that the acceptance probability falls the recommended range of [20%, 40%]. We set a soft population constraint with $\beta = 27$ (Algorithm 1.2 and set q to 0.05, and run four separate chains for 20,000 iterations each. For each chain and each separate run of the algorithm, we draw starting values using the random seed-and-grow algorithm described in Algorithm 3. After running the chains, we discard any draws that fall outside a population parity bound of 1%. Figure S5 shows the results of this comparison. On the New Hampshire map, we find that according to the Gelman-Rubin diagnostic, the multiple swaps algorithm (right column) converges much faster than the single-swaps algorithm (left column), and also shows lower autocorrelation.

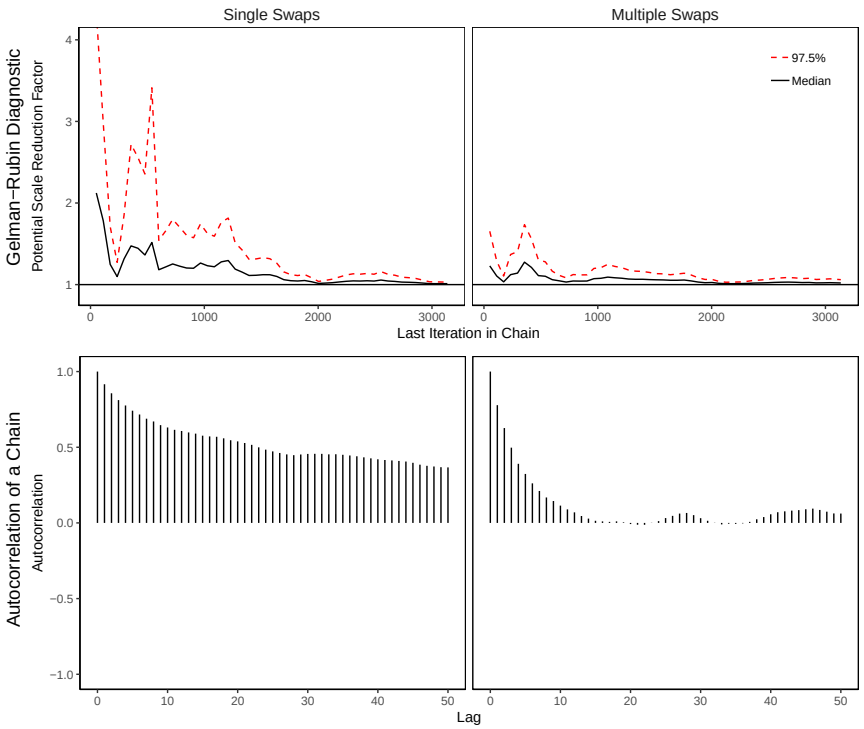

Figure S5: Convergence Diagnostics of the Proposed Algorithm with Single and Multiple Swaps using the 2008 New Hampshire Data. The proposed algorithm with a soft constraint (Algorithm 1.2) is applied to the New Hampshire data. The target population consists of all redistricting plans with contiguous districts and a maximum of 1% deviation from the population parity. We compare the basic algorithm with R set to 1 (swapping a single connected component in each iteration; left column) to simulations where R is set to 32 (swapping on average 32 connected components in each iteration; right column). We find that the multiple swaps algorithm converges quicker according to the Gelman-Rubin Rhat diagnostic, and exhibits lower autocorrelation.

S7 Runtime Comparison

Here, 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 S6 compares the wallclock runtime between the proposed basic algorithm with hard constraint (Algorithm 1.1) and the standard algorithm (Algorithm 3) 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 3 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

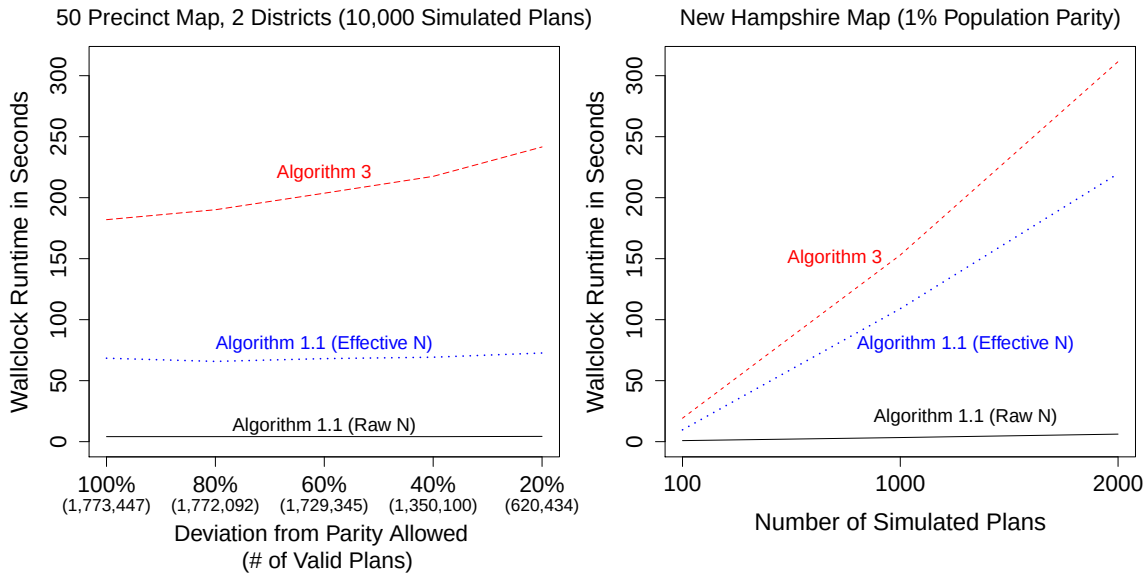

Figure S6: Runtime Comparison between the Proposed and Standard Algorithms in a Small-scale Validation Study and the New Hampshire Map. The runtime is compared between the proposed basic algorithm with hard constraint (Algorithm 1.1) and the standard algorithm (Algorithm 3). In the small-scale validation 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).

effective number (Effective N; blue dotted lines) of simulated plans. In addition, the difference in wall-clock 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 2, which took approximately 30 hours to complete. When testing the standard algorithm (Algorithm 3), 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 3 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.

S8 Accuracy of the Approximate Acceptance Ratio

We conduct two simulation studies to empirically examine the performance of the approximate acceptance ratio discussed in Section 2.3. First, we generate three lattices of dimensions 3×2, 3×3, and 4×3. We choose these small lattices because we can compute the true target distribution (i.e., uniform distribution) by enumerating all valid redistricting maps. In addition, as explained in Section 2.3, we expect the approximation to be poor for small maps. Therefore, these validation maps, which are much

smaller than actual redistricting maps, serve as a hard test of the proposed approximation.

To generate simulated populations for the lattices, we fit the negative binomial distribution to the precinct populations and Republican vote count, respectively, of New Hampshire (see Appendix S6). For the the precinct populations, the parameter values that maximize the likelihood of the negative binomial distribution are 0.577 (dispersion parameter) and 3973.465 (mean parameter). For Republican vote count, these maximum likelihood estimates are 0.647 (dispersion parameter) and 1020.135 (mean parameter). We draw the total and Republican populations for each precinct according to the negative binomial distribution with these parameter values. We then specify every valid partition of these lattices into two contiguous districts and calculate the Republican dissimilarity index of each districting plan, in order to compare our algorithm's performance against the true distribution.

Unfortunately, even in small-scale problems, computing the exact ratio is intractable due to the non-adjacency restriction of $\mathbf{V_{CP}}$, so we cannot measure the approximation error. Instead, we compare the performance of the proposed algorithm based on two versions of the acceptance ratio: the weak approximation given in Equation (8), which is based on the set of boundary components $B(\mathbf{CP}, \pi)$, and a stronger approximation based on the set of connected components which do not *independently* shatter a district when removed, $B_1^\dagger(\mathbf{CP}, \pi)$. Note that both approximations ignore the effects of component adjacency on the selection probability. The weak version, which we refer to as the B approximation, uses

$$\frac{P(\mathbf{V_{CP}} \mid \pi', \mathbf{CP}, R)}{P(\mathbf{V_{CP}} \mid \pi, \mathbf{CP}, R)} \approx \left(\frac{B(\mathbf{CP}, \pi)}{B(\mathbf{CP}, \pi')} \right)^R, \tag{S20}$$

while the stronger version, the $B^\dagger$ approximation, uses

$$\frac{P(\mathbf{V_{CP}} \mid \pi', \mathbf{CP}, R)}{P(\mathbf{V_{CP}} \mid \pi, \mathbf{CP}, R)} \approx \left(\frac{B_1^\dagger(\mathbf{CP}, \pi)}{B_1^\dagger(\mathbf{CP}, \pi')} \right)^R, \tag{S21}$$

For the sake of comparison, we do not impose an equal population constraint. Results are shown in Figure S7. Using our weak and strong approximations, we run the proposed basic algorithm (Algorithm 1) 1,000,000 times on each lattice and use QQ plots to examine the degree to which the performance of the proposed algorithm can approximate the true target distribution. The proposed algorithm with the stronger approximation based on $B^\dagger$ approximates the target distribution almost perfectly in all three cases, suggesting that the effects of component adjacency are negligible in the ratio of selection probabilities. In contrast, our proposed algorithm with the weaker approximation based on B acceptance ratio performs poorly in the smallest lattice, and yet the performance improves dramatically as the size of the lattice increases. Both of these suggest that component adjacency has a negligible effect on the selection probability ratio, and that the approximation $B(\mathbf{CP}, \pi)/B(\mathbf{CP}, \pi') \approx B^\dagger(\mathbf{CP}, \pi)/B^\dagger(\mathbf{CP}, \pi')$ becomes more valid as the size of the graph increases.

In a second simulation study, we directly compare the values of the acceptance ratios for the weak and strong approximations using a set of lattices ranging from 4×4 to 12×12. We run the proposed basic algorithm (Algorithm 1) for 5000 iterations to partition each lattice into 4 districts (we also tried the cases with 2 and 3 districts but the results are similar to those shown below). For each iteration of the proposed algorithm with the acceptance ratio based on the $B^\dagger$ approximation, (α_{strong}), we also store the acceptance ratio based on the B approximation, (α_{weak}). Figure S8 plots the distribution of the absolute

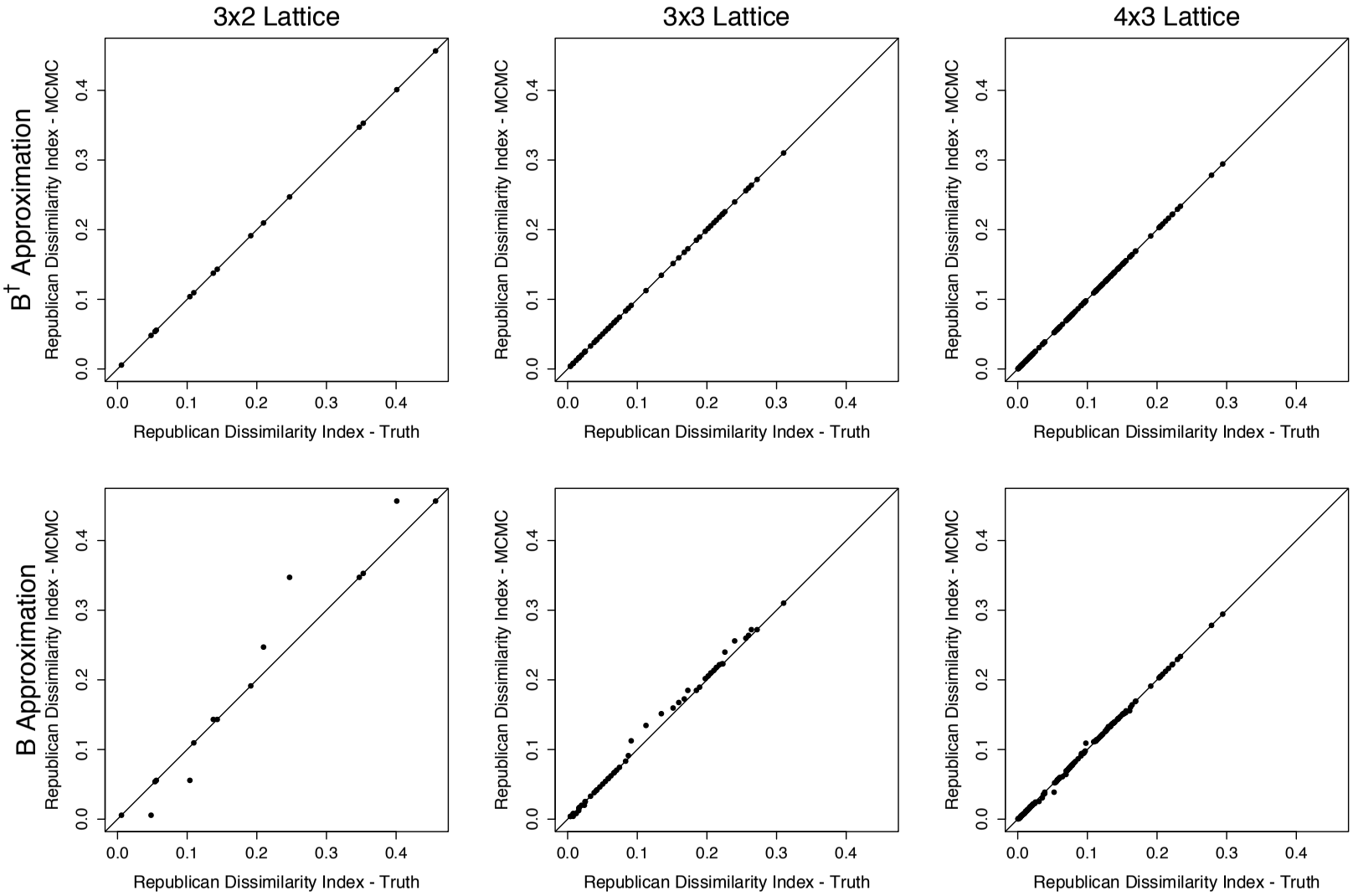

Figure S7: A Small-scale Validation Study Using Lattices. The underlying data are a series of lattices with simulated precinct populations. The plots in the first row show that the computation of the acceptance ratio based on the $B^\dagger$ approximation samples from the target population nearly perfectly. The plots in the bottom row show that the performance of the proposed algorithm based on the B approximation improves as the size of the map increases. The results for each algorithm is based on a single chain of 1,000,000 draws using the basic algorithm (Algorithm 1).

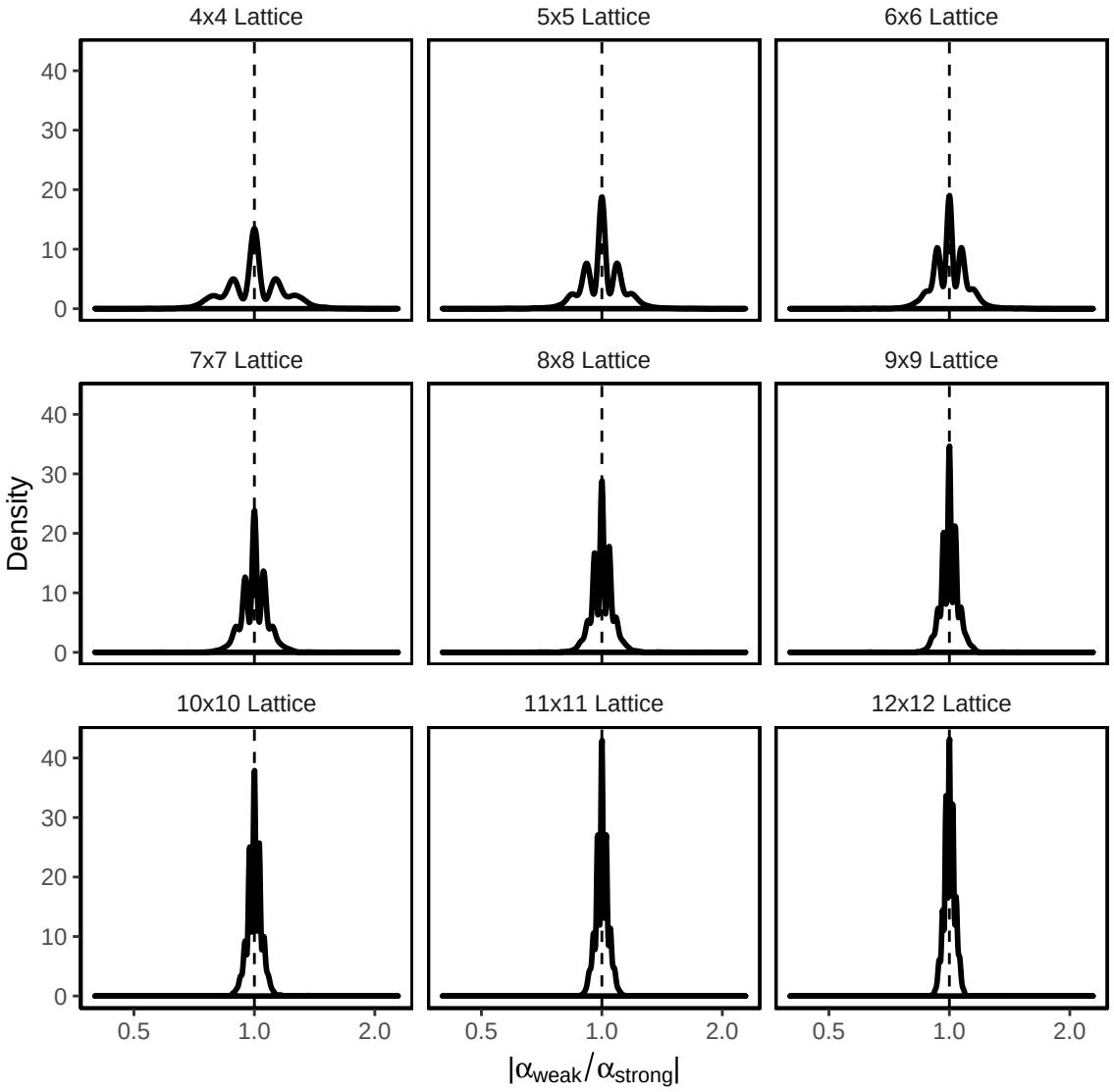

Figure S8: Comparison of the Exact and Approximate Acceptance Ratios. The underlying data are a series of lattices with simulated area and group populations. Each pane shows the results of simulations on lattices of increasing sizes, starting with a 4×4 lattice and going up to a 12×12 lattice. The proposed basic algorithm (Algorithm 1) is run to partition each lattice into four districts, and both the exact and approximate acceptance ratios are computed and stored for each iteration. We find that as the map increases in size, the approximate ratio becomes closer to the exact ratio. Results for partitioning the lattices into 2 and 3 districts show similar improvements in performance as the lattice increases in size.

ratio of the weak acceptance ratio over the strong acceptance ratio, i.e., $|\alpha_{\text{weak}}/\alpha_{\text{strong}}|$. This analysis shows that the weak approximation approaches the strong approximation as the size of the target map increases. For lattices above 10×10, the ratio of the weak acceptance ratio to the strong acceptance ratio is tightly clustered around 1. Given that the smallest actual redistricting problems in the United States are over three times as large as the largest lattice we consider here, the results shown above give us more reassurance that our weak approximation is likely to work well in actual redistricting problems.

For this study, we also examine the performance of the approximations by tracking how frequently

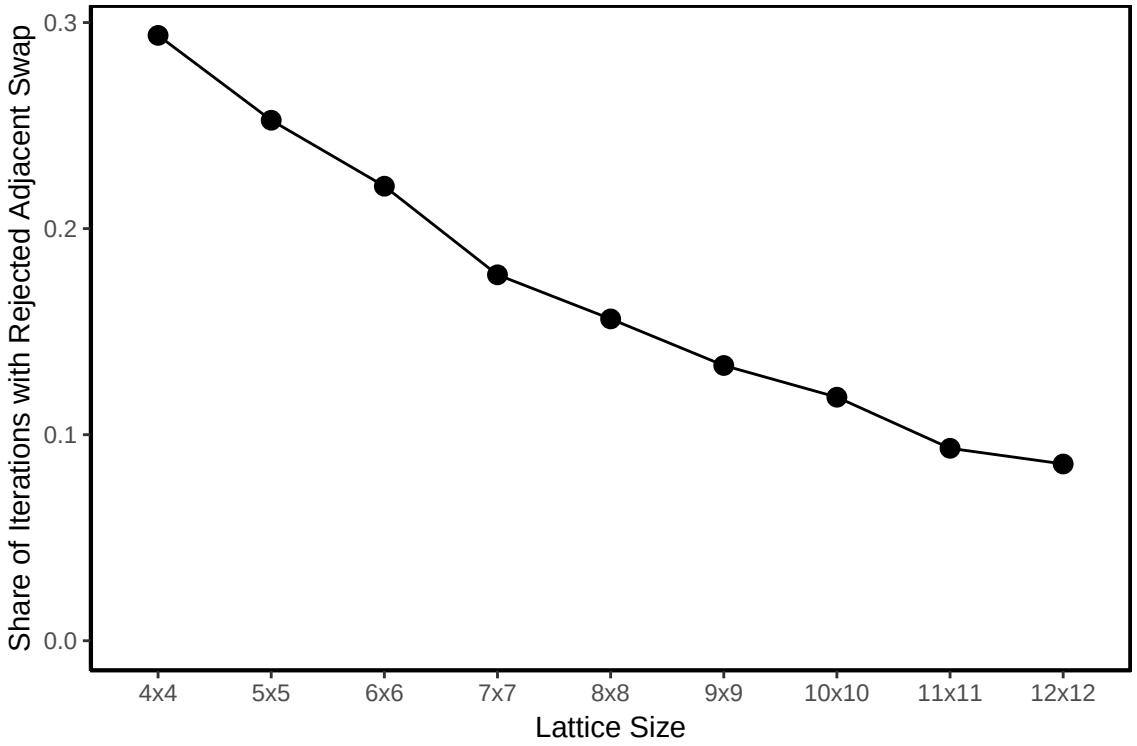

Figure S9: Tracking the Frequency of Rejecting Partitions due to Adjacency Restriction. The underlying data are a series of lattices of increasing sizes, starting with a 4×4 lattice and going up to a 12×12 lattice. The proposed basic algorithm (Algorithm 1) is run to partition each lattice into four districts with $\lambda = 2$, and in each iteration we check whether or not a proposed partition swap is ever rejected due to its adjacency with another proposed connected component in the same iteration. We find that as the map increases in size, the share of algorithm iterations that reject a partition due to adjacency quickly falls. Results for partitioning the lattices into 2 and 3 districts show similar improvements in performance as the lattice increases in size.

a proposed partition is rejected because it is adjacent to another proposed connected component in the same iteration (see Step 3 of Algorithm 1). For each simulation run in Figure S8, we also store when a partition is rejected due to this adjacency restriction — the less frequently it occurs, the better the strong ($B^\dagger$) and weak (B) approximations will perform relative to the exact acceptance ratio. The results are shown in Figure S9. We find that as the lattices increase in size, this rejection behavior that could lead to bias becomes much less likely. While in the 4×4 lattice, this occurs in around 30.5% of all iterations, it falls to 8.8% of iterations in the 12×12 lattice. These results give us more confidence that bias due to the approximation of the ratios with multiple swaps decreases as the size of the redistricting problem increases, and that it should be negligible in state-sized redistricting problems.

References

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.