Quantifying Gerrymandering by using the Recombination Markov Chain and Score Function

Andy Wu, Nicolas Johnson

November 2021

1 Research Context and Problem Statement

Many believe that gerrymandering is an issue of the past, but it remains a widespread issue in the United States. This issue can be attributed to the complex U.S. voting system. For context, most states in the U.S. are divided into a number of districts, and the total votes in each district elects a single elector or congressman. Oftentimes, insufficient restrictions limit how states draw their districts. States can split counties and ignore significant geographical landmarks, creating an imbalance in demographics (republicans, democrats, minorities, etc.) in each district and influencing the results of votes. Unfortunately, some states purposely draw biased redistricting plans to elect representatives of a particular political party. This practice, called gerrymandering, is infamous for undermining U.S. democracy.

In order to combat gerrymandering, we need ways to quantify how much a district plan is gerrymandered, and then we could use this quantification as evidence in court to invalidate any gerrymandered plans. However, quantifying the extent of gerrymandering is a difficult computational task. Recent studies have found an efficient method called the Markov Chain Monte Carlo (MCMC) algorithm. This algorithm randomly generates a large collection of potential redistricting plans, and researchers can then compare this collection of redistricting plans with actual government district plans. Because the MCMC is random, the large sample of randomly generated redistricting plans will converge to a non-partisan, normal distribution. This distribution is ideal for comparison with real district plans because of the large sample size and non-partisanship. If the real district plans do not occur frequently in the distribution, then those elections are likely subject to gerrymandering.

Studies have been steadily developing ways for using the MCMC approach to quantify gerrymandering. Mattingly and Vaughn are accredited for their important development of a score function for the MCMC algorithm [4]. The purpose of the score function was to constrain the population size and shape of a district for better representation of non-partisanship when generating redistricting plans. As a result, the score function helps sample randomly generated redistricting plans that are better suited for comparisons with real district plans. A later study in 2018, ”Quantifying Gerrymandering in North Carolina,” then builds on this concept by implementing the score function with a Flip Markov Chain approach (Figure 1) [3]. Their Flip Markov Chain approach groups precincts (a state-determined city/town area) into the required number of districts. The algorithm then generates new redistricting plans by randomly switching which district a precinct belongs to. This ”flipping” procedure can then be ran any number of times to generate more redistricting plans.

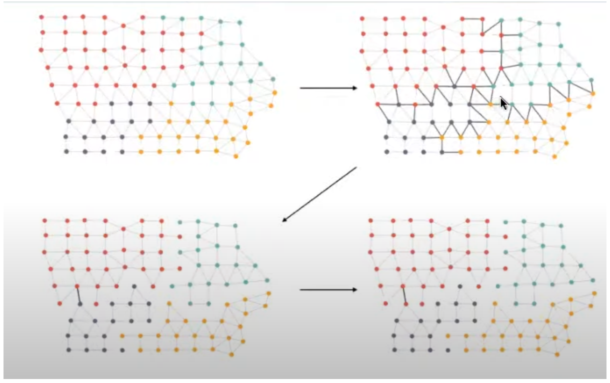

Figure 1: Flip Markov Chain: Each dot represents a precinct, and edges are drawn to connect each precinct to their neighboring precincts. The color of the dots represent what district they belong to. The edges that connect precincts of different districts (darker grey) are taken, and one of them are randomly selected. Then the algorithm will take the precinct connected by that edge, and it will swap which district that the precinct belongs to.

However, the Flip Markov Chain approach is relatively slow at generating redistricting plans because this method can only flip a single voting precinct at a time. There have been a few solutions to solving this problem, one of which is to use graph-cuts as done in ”Automated Redistricting Simulation Using Markov Chain Monte Carlo” [2]. The graph-cut approach works by putting the district plan in a graph and then partitioning it into smaller sub-graphs. This method then swaps the district-alignment of a sub-graph of precincts. Thus, graph-cut is faster than the Flip Markov Chain because it swaps multiple voting precinct at once rather than just one precinct. But this solution comes with a major drawback because the study has not been able to use their graph-cut method when there are multiple constraints for generating redistricting plans. This means that the graph-cut approach cannot be used with a score function because score functions take into account multiple constraints (including equal population and compactness).

Another solution to solve the speed of the MCMC algorithm is to use the new Recombination Markov Chain method [1]. This approach uses spanning trees (Figure 2) for more efficient generation of redistricting plans [5]. The Recombination Markov Chain merges two neighboring districts and randomly selects a root point (a precinct) in the merged districts. The algorithm then draws a tree from the root to all of the precincts in the merged districts. It then cuts a random edge in the tree, leaving two new trees which are the new district groupings. With this approach, the districts of multiple voting precincts can be swapped at the same time, unlike with the Flip Markov Chain.

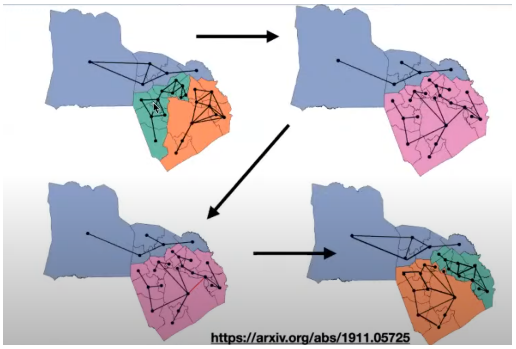

Figure 2: Recombination Approach: The green and orange districts are merged into the purple region. Each point represents a precinct, and the edges connect each precinct into a spanning tree in the purple region. An edge is randomly selected (in red) and is then cut, splitting the spanning tree into two pieces and forming a new redistricting plan.

Although the Recombination Markov Chain is more efficient than the Flip Markov Chain, it also has an underlying drawback. The Recombination Markov Chain shares the same issue with the graph-cut approach as it has not yet been implemented with a score function. However, the simplicity of the new Recombination approach makes implementing the score function more plausible. If the Recombination approach could be used with a score function, then the MCMC algorithm would generate better non-partisan redistricting plans that are more useful for comparisons. As a result, implementing the score function would help give the judicial courts more confidence in how the MCMC quantifies gerrymandering. So implementing the score function with the Recombination Markov Chain would contribute to the development of the latest MCMC technology, making the algorithm more dependable when quantifying gerrymandering.

2 Proposed Solution

We will implement the score function into the Recombination Markov Chain. The Recombination Markov Chain is one of the most efficient MCMC approaches to quantify gerrymandering, and since it is very new, we will be one of the first studies to use this algorithm with a score function. This is an important step for the research area around using the MCMC to quantify gerrymandering because we will improve one of the fastest MCMC methods by optimizing its non-partisan legitimacy. We will improve the algorithm's legitimacy by ensuring that the Recombination Markov Chain will randomly sample redistricting plans that adhere to non-partisan criteria set by the score function. This would make the sampled redistricting plans much more ideal for comparisons with actual government district plans. As a result, the Recombination Markov Chain with the score function would be able to provide better, full-proof evidence against gerrymandering as it would give judicial courts more confidence in the algorithm's implementation. For example, the courts would no longer need to worry about if the randomly generated redistricting plans are biased in favor of the researchers. This is because the score function sets constraints for the MCMC algorithm to adhere to very comprehensive and non-partisan criteria.

In order to implement our score function into the Recombination Markov Chain, we will consider a case study. Our goal will be to develop a small scale Recombination Markov Chain algorithm to quantify the extent of gerrymandering in Utah. Utah has been overlooked by past studies, so this case study will also act as one of the first investigations of gerrymandering in Utah via an MCMC approach. Utah has consistently voted Republican in past elections, giving reason for us to hypothesize that Utah is at least slightly gerrymandered in favor of Republicans.

Utah is ideal for running the Markov Chain because it is a small state with only 4 districts, and it also has a very even, rectangular shape. In order to quantify gerrymandering in Utah, we will first need to collect and clean Utah's past voting precinct data which can likely be found online. The next step is to use the Recombination Markov Chain technique on Utah without the score function or any other constraints. Once we have the algorithm set up, we can begin introducing a way to constrain the Recombination Markov Chain to only select redistricting plans that adhere to the non-partisan criteria set by a score function. An ideal score function for us to use can be very similar to the one from the Mattingly paper [3]. Their score function is a great model because it was able to give enough leeway for their algorithm to sample a sufficiently large and random variation of redistricting plans. Finally, after we have finished implementing the score function with the Recombination Markov chain, we can use the algorithm to generate a collection of random redistricting plans for Utah. These plans will then be used to compare with the real district plans of Utah. This comparison will be done by using past precinct voting data with the new, generated redistricting plans. We can then compare the election results of those generated plans with the real election results, and we observe for outlying or dramatic differences in the election results to help find evidence of gerrymandering in Utah's district plans.

3 Evaluation and Implementation

3.1 Evaluation Plan

To measure the extent of gerrymandering in a Utah election, we will use a z-test to analyze the probability of the Utah election results occurring within the generated redistricting plans (we may also utilize the new concepts of gerrymandering index and representativeness index) [3]. The z-test measures the probability that something will occur in a normal distribution (a bell curve). In this case, we measure the probability that Utah's election plan occurs in our collection of generated redistricting plans. Additionally, we will utilize a third-party non-partisan redistricting plan (although this method has not been finalized) to analyze the extent to which our generated data is non-partisan. For this experiment, we will generate a large number of redistricting plans utilizing MCMC and recombination. This data should for a fairly nice bell-curve in which we can utilize a z-test to calculate the probability of the actual redistricting plan occurring. We will utilize a confidence interval (a threshold of what probability will determine if Utah's election plan is gerrymandered or not) of 0.05 or 0.01 depending on the sample size. However, we will likely just be using 0.05 because anything falling less than 0.05 will have a high probability of gerrymandering. Furthermore, we may also attempt at incorporating the gerrymandering index and representativeness index presented in ”Quantifying Gerrymandering in North Carolina” [3]. These indices (which require further research for proper data analysis) can take our data analysis a step further than only using a z-test.

We will likely utilize either a control case for Utah or a third, non-partisan party (if we find one) to evaluate the success of this experiment. If we utilize the control case, we would likely redistribute the Utah votes to 50% Republican and 50% Democrat and generating numerous redistricting plans on the re-balanced Utah to see if the resulting plot results in the expected 50:50 ratio. If the resulting elected parties are around the expected ratios, then the algorithm likely generates non-partisan redistricting plans. If we utilize a third party generated non-partisan redistricting plan, then we will compare that non-partisan ”control” plan to our generated data to see if the control plan falls within a 95% interval of bell-curve. We will utilize one of these methods in order to ensure that our data is arguably accurate.

3.2 Timeline

(Dates may be shifted around as experiment continues)

Winter Quarter:

(Week 1) Experimentation with MCMC on a grid
Incorporate recombination/spanning trees into the grid

(Week 2) Continue experimentation on grid

Find usable voting data for Utah (as clean as possible)

(Week 3) Clean obtained data
Research/play with implementing MCMC into non-grid graphs

(Week 4) Begin playing with data and constructing MCMC algorithm on Utah

(Week 5) Continue implementation/debugging of MCMC on Utah

(Week 6) Continue implementation/debugging of MCMC on Utah
(hopefully generate some redistricting plans to refine upon)

(Week 7) Continue implementation/debugging of MCMC on Utah
Generate decent sample of redistricting plans

(Week 8) Improve algorithm to take in more voting factors into consideration

(Week 9) Generate a large sample of redistricting plans
Compare generated sample to Utah's voting outcome
Ensure that generated sample is non-partisan

(Week 10) Compile data and begin analysis of data

Spring Quarter:
Mainly reserved for data analysis, paper writing, and presentation preparation.

(Week 1) Analyze extent of non-partisanship in generated data
Analyze Utah voting outcome compared to generated data

(Week 2) Analyze extent of non-partisanship in generated data
Analyze Utah voting outcome compared to generated data
Begin writing paper: Context

(Week 3) Write paper: Introduction
Write paper: Works Cited

(Week 4) Write paper: Methods

(Week 5) Write paper: Analysis

(Week 6) Write paper: Conclusion
Work on presentation

(Week 7) Refine paper
Work on presentation

(Week 8) Refine paper
Work on presentation

(Week 9) Refine paper
Work on presentation

(Week 10) Work on presentation

References

[1] Daryl DeFord, Moon Duchin, and Justin Solomon. Recombination: A family of markov chains for redistricting. *Harvard Data Science Review*, 2021.

[2] Benjamin Fifield, Michael Higgins, Kosuke Imai, and Alexander Tarr. Automated redistricting simulation using markov chain monte carlo. *Journal of Computational and Graphical Statistics*, 2020.

[3] Gregory Herschlag, Han Sung Kang, Justin Luo, Christy Vaughn Graves, Sachet Bangia, Robert Ravier, and Jonathan C. Mattingly. Quantifying gerrymandering in north carolina. 2018.

[4] Jonathan C. Mattingly and Christy Vaughn. Redistricting and the will of the people. 2014.

[5] Ariel D. Procaccia and Jamie Tucker-Foltz. Compact redistricting plans have many spanning trees. 2021.

4 Revision Changes

The most important thing we had to address was what our paper was contributing to. Although our prior proposal was to address gerrymandering in Utah, this was not a good enough contribution to the studies around the MCMC algorithm. Therefore, we chose to make a new implementation of an MCMC algorithm where we use the newest algorithm (Recombination) with an addition of a score function. This proposal is hopefully much more clear for understanding its importance with regards to developing more efficient MCMC algorithms for gerrymandering.

Another problem was that our problem statement and proposed solution were just generally unclear and not well connected. We tried to solve this by cleaning up the problem statement to be more concise and direct with the point we were trying to make, and then we tried to more directly connect the problem statement to the solution.

Yet another large problem was how we presented the technical details in our paper. To solve this, we used more figures and we also organized the flow of technical information. There was also confusing information that was trivial to know in our proposal, so we removed it and instead added in more important context for our paper.