The Mathematics of Gerrymandering

Using computational redistricting methods to assess gerrymandering

Emilia Podgorski

Abstract

Gerrymandering has been an issue in the United States for almost as long as the nation itself has existed. Many researchers have tried to find a mathematical metric to quantify gerrymandering, but the social, political and geographical complexity of the issue has often made it difficult to reduce to a single number. This project instead explores computational methods for redistricting and how they can be used to help identify gerrymandering. In particular, it focuses on Markov Chain Monte Carlo methods, such as the recombination method proposed by DeFord et al.. This method generates a diverse range of potential plans, which offers a comparative context to the issue that other methods for detecting gerrymandering lack. We modify the recombination algorithm to use a spectral cut instead of a spanning tree cut, and assess its effectiveness on a toy example. We then look at a case study of Pennsylvania's congressional district plan from 2011, and use the new modified algorithm to demonstrate how these computational methods can be used to identify plans that may be outliers to help courts reach a decision on gerrymandering.

Contents

1 Introduction

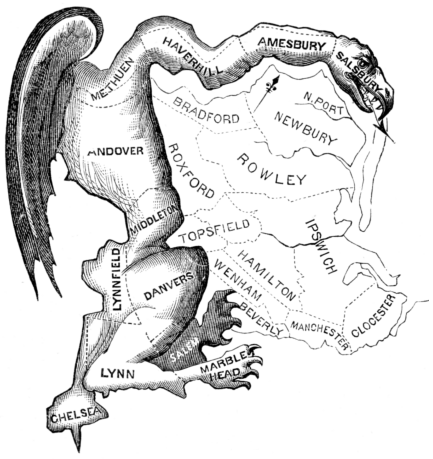


Figure 1: Gerry's Salamander by Elkanah Tisdale (1771-1835) - Originally published in the Boston Centinel, 1812

This project is on the mathematics of gerrymandering, specifically in the US. The word gerrymandering originated in 1812 from "Gerry's Salamander". This was the nickname given to the district of South Essex created by Elbridge Gerry, the governor of Massachusetts. The shape of the district was so irregular that it resembled a salamander, and inspired a political cartoon (seen in Figure 1). While not necessarily the first instance of electoral district manipulation, it is certainly the most famous. This odd-shaped district did work as intended: Though both parties received a roughly equal amount of votes, Gerry's Democratic-Republicans won 29 seats compared to the Federalists 11 seats.[1] Elbridge Gerry himself later went on to become the 5th Vice President of the United States.

The debate around "fair" districting in the US has been around for as long as the nation's independence. In 1787 Madison wrote that electoral districts should be transitable, and a 1901 law declared that they should be made from "compact territory".[2] More recently, in 1986 the Supreme Court ruled that extreme partisan gerrymandering is unconstitutional.[3] Other legislations, such as the Voting Rights Act of 1965 which prohibited racial discrimination in voting, have made gerrymandering into a serious legal issue that requires a clear definition.

Different ways to measure gerrymandering have been proposed, such as using "wasted" votes to calculate an efficiency gap, comparing areas and perimeters with an isoperimetric quotient, and partisan symmetry.[4][5] The main problem with this issue is that each state has their own individual laws surrounding redistricting, and a solution for one state may not work in another. It doesn't appear to be possible to produce a one-size-fits-all solution.

Therefore, this project will focus on computational redistricting, and how it can be used to assess the "fairness" of a districting plan relative to other potential plans. Computing every possible plan would require a lot of time and power, so it is more effective to gather a representative sample using Markov Chain Monte Carlo (MCMC) methods. Many advances have been made in this specific field over the last 10 years.

Section 2 will explore both the history of gerrymandering and the use of maths in the courtroom. Section 3 will give a background on MCMC techniques, and Section 4 will examine the strengths and weaknesses of different proposed metrics for gerrymandering, as well as how they can disagree. Section 5 will investigate existing MCMC methods for redistricting, before a modified version of the recombination algorithm by DeFord et al.[6] is introduced in Section 6. The effectiveness of this new algorithm will then be assessed using a simple 8×8 grid example. Finally, Section 7 will be a case study of Pennsylvania's 2011 congressional district plan to demonstrate how these algorithms can identify potential gerrymanders.

2 History

Ever since Elbridge Gerry's salamander, gerrymandering has been running rampant in the US. In 1853 the Democrats managed to win 10 out of 11 districts in Indiana with just 53% of the vote.[1] This manipulation actually led to a lot of backlash from voters, and ironically they ended up losing most of these districts to the Federalists the next year.[7] Gerrymandering has also been used for racial discrimination. After Black men were given the right to vote in 1870, Southern states gerrymandered districts to dilute their voting power. In 1882 South Carolina created a "boa constrictor" district to pack all the Black voters in, therefore decreasing their voting influence in other districts.[7] South Carolina also attempted to create non-contiguous districts in 1874, but were forbidden by the US House of Representatives.[7]

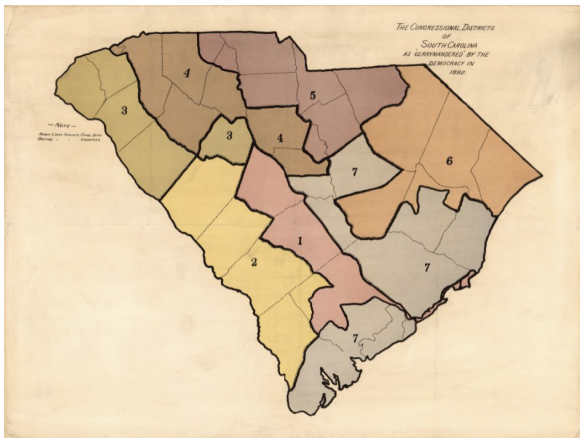


Figure 2: Map of the Congressional Districts of South Carolina from 1882, retrieved from the Library of Congress. District 7 is the "Boa Constrictor" district.

These manipulations of districts showed the need for proper legislation on gerrymandering. By the 1960s many of these Southern states hadn't updated their racially segregated district plans. One of Alabama's rural districts had a population of just 15,000 voters, while another district with a large Black population had 600,000 voters.[1] This disparity led to the Voting Rights Act of 1965. It sought to dismantle some of the obstacles that Black people faced when voting, such as literacy tests and poll taxes.[8] Most importantly for the issue of redistricting, it prohibited the vote dilution of minority groups. This is significant because it provided a concrete law against gerrymandering that has since been used to challenge potentially discriminatory districting plans in court. However, some form of convincing analysis is required to prove the plan is discriminatory.

There is some history of mathematics being used in court. For example, in 2005 James Robbins was accused of selling drugs within 1000 feet of school property. He argued that he was actually 1256 feet of walkable path away from the school, while the prosecution countered that he was 907 feet away "as the crow flies".[9] This is an argument between using the Euclidean and the Manhattan metric to measure distance. The judge ruled against Robbins, which is an example of how mathematical concepts can be used alongside common sense in court. In this case, although the metric for measuring distance hadn't been specified in the law, it was taken as common sense to be the Euclidean metric. He was within an 1000 feet radius of the school. This demonstrates how mathematical arguments don't always hold up on their own, there needs to be an element of human judgement too.

Mathematical methods can help detect gerrymandering. In the case of Bethune-Hill v. Virginia Board of Elections, residents of 12 districts with $\geq 55\%$ Black Voting Age Population (BVAP) voters argued that this concentration of voters was unconstitutional, and diluted their voting power in neighbouring districts.[10] The court originally ruled that is wasn't a gerrymander, but the Supreme Court later ruled that they had "applied the wrong legal standard".[11, pp. 6-7] They argued that it wasn't sufficient to just compare the plan to traditional districting standards, in order to get a full sense of potential gerrymandering the case had to be viewed holistically within the context of the whole state.[11] This shows how the legislation on gerrymandering is quite vague and can be interpreted in different ways. In June 2018, the Eastern District Court of Virginia ruled that 11 of those 12 House districts were indeed unconstitutional.[10] After this ruling, researchers at the

Metric Geometry and Gerrymandering Group analysed these districting plans by comparing them to a large ensemble of alternative plans. Through this method they found the original plan was a statistical outlier. They concluded that BVAP voters were unnecessarily high in 11 of the challenged districts, which decreased their voting power in the 22 neighbouring districts.[10] This shows how mathematical methods can help aid court rulings on gerrymandering. Though there are political and geographical factors that mean gerrymandering can't be fully assessed using maths alone, it's a useful tool that can help identify plans that may be outliers.

3 Mathematical Background

To get an understanding of how computational redistricting methods work, we first need to know the basic concepts behind them. These include Markov chains and Monte Carlo methods, which are combined to create Markov Chain Monte Carlo (MCMC). These are the fundamentals of the algorithms used to collect diverse groups of districting plans.

3.1 Markov Chains

A random walk is a process where random steps are taken in a mathematical space to create a path. It's a stochastic process, which means it evolves randomly over time. The simplest example of this is a random walk on the number line, where at each step the probability of moving either +1 or -1 is 0.5 each. Mathematically, this is described:

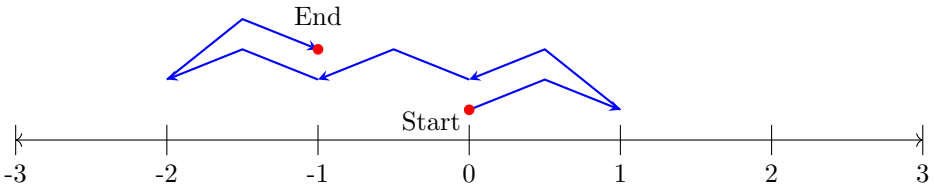
Definition 1 *Let X_1, X_2, ... be independent identically distributed random variables, taking values ± 1 with equal probability. Then the* ***Simple Random Walk*** *S_n is defined as*

$$S_n = \sum_{i=1}^{n} X_i$$

[12]

S_n is the position after n steps of the walk, or the nth state of the system. For example, imagine a person standing on the number line, starting at 0. They decide to flip an even coin, with heads representing a move +1, and tails representing a move -1. Here are the first 5 steps of the walk:

Current State (S_{n-1})	Move (X_n)	New State (S_n)
0	+1	1
1	-1	0
0	-1	-1
-1	-1	-2
-2	+1	-1



So $S_5 = -1$

A Markov Chain can be defined as a random walk without memory.[6] The probability of an event happening depends only on the current state of the system. This means you can make accurate predictions for future outcomes based on the current state, as past outcomes don't affect the probability. In the number line example, the probability of the next state being 0 or -2 is 0.5 either way, while every other state has a probability of 0. This is solely based on the current state being -1. Therefore, it is a Markov chain.

Note: Not every random walk is a Markov Chain. An example of this is a random walk where the probability of the next step depends on the previous **two** steps, rather than just the last one. Take the number line example again, but this time add the condition that if the previous two steps were $+1$ the next step must be -1, and vice versa. If the previous two steps were different then the move is chosen randomly. This random walk isn't a Markov chain because the next step doesn't depend solely on the current state, it also relies on the step before it. Therefore it's an example of a random walk that isn't a Markov chain.

Definition 2 *A sequence of random variables X_1, X_2,... is called a **Markov Chain** if*

$$\mathbb{P}(X_n = x_n : X_1 = x_1, ..., X_{n-1} = x_{n-1}) = \mathbb{P}(X_n = x_n : X_{n-1} = x_{n-1})$$

[12]

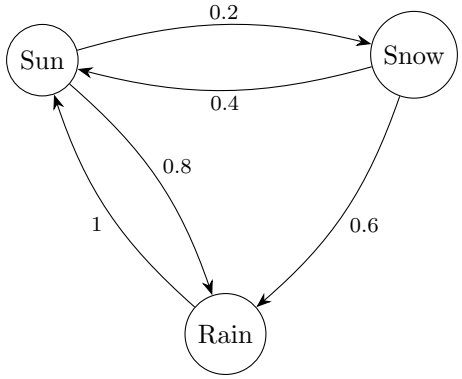
The probability of moving from one state to another is called the transition probability. Given the transition matrix T where T_{ij} is the probability of transitioning from state i to j, and probability vector P where P_i is the current probability of being at state i, then the probability of being at state j in the next step of the Markov Chain is:

$$Q_j = \sum_{i=1}^{n} P_i T_{ij}$$

For a small example of a Markov chain, we can look at a very simplified weather forecast. Here the state space is {Sun, Snow, Rain} with the transition matrix

$$T = \begin{pmatrix} 0 & 0.2 & 0.8 \\ 0.4 & 0 & 0.6 \\ 1 & 0 & 0 \end{pmatrix}$$

$T_{12} = 0.2$ means that the probability of it snowing today given that it was sunny yesterday is 0.2. Note that the values in each row add up to 1. This is because it is the total probability of moving from the current state to another, which always sums to 1. It is also possible to have self loops, when there are for example 2 sunny days in a row. This Markov chain can be represented in a diagram:



In this example we can see that if it was raining yesterday then there's no chance of it snowing today. This is because the probability of the next state depends solely on the current state, and in this markov chain we have $T_{32} = 0$. So there is no chance of moving from a rainy day to a snowy one.

Definition 3 (Key Definitions for Markov Chains)

1. *A Markov Chain is* ***irreducible*** *if each state is (eventually) reachable from every other state in a finite number of steps.*
2. *A Markov Chain is* ***aperiodic*** *if for each state the greatest common divisor of all the possible lengths of loops starting and ending at that state is equal to 1.*
3. *A Markov Chain is* ***ergodic*** *if it's both irreducible and aperiodic. [13]*

The Markov chain in the weather example is irreducible because no matter where you start in the chain it's possible to reach every other state, and aperiodic because each state can return to itself in loops of length 2 or 3. Therefore it is also ergodic. Ergodic Markov chains have a useful property.

Theorem 1 (Fundamental Theorem of Markov Chains) *Ergodic Markov chains have a unique stationary distribution P that satisfies*

$$P_j = \sum_{i=1}^{n} P_i T_{ij}$$

for each state j. [13]

This stationary distribution is the limit of the probability distributions formed by iterating the Markov Chain. This means that, after a certain amount of steps, taking another step of the Markov chain won't change the probabilities of being at each state. The number of steps it takes for a Markov chain to converge to the steady state from an arbitrary starting point is called the mixing time.[6]

An application of Markov chains is in Google's PageRank algorithm. The PageRank algorithm was created by Brin and Page in the late 90s as a response to the rapid growth of the internet.[14] Other search engines such as Yahoo that used "human maintained" lists were effective but expensive and difficult to scale up, while automated machines that matched keywords were ineffective and easy for businesses to manipulate.[14] Therefore, they decided a mathematical method was required to search billions of webpages and return the most relevant results first.

The method involves modelling a random surfer using links to move from page to page. This is done using a Markov chain, where the webpages are nodes and the links are directed edges.[15] Each webpage is assigned a score based on various factors such as the quality and quantity of references to that webpage from other pages.[14] This score is the probability that the random surfer clicks on that page.[15] This method helps the algorithm display the most relevant webpages first, as the more likely a random surfer is to click on a page, the more likely that it will be useful to the current surfer. Ever since this method was introduced Google has far outgrown its competitors, showing the usefulness of mathematical methods.

3.2 Markov Chain Monte Carlo (MCMC)

The Monte Carlo method was first developed by Stan Ulam when he was estimating the probability of winning a solitaire game. Calculating this exact probability would be long and difficult, so he decided to evaluate the probability of winning individual examples of the game, and then used them to find the average probability.[16] The general process of a Monte Carlo method is:

1. Draw an independent sample
2. Calculate some quantity based on this sample
3. Repeat the process on a sufficient number of samples
4. Average the values obtained

[16]

For example, say you want to calculate the probability of winning a simple card game. In this game you draw the top 3 cards from a shuffled deck and win if they're in descending order. Given how many different ways there are to shuffle a deck of cards, the probability of winning is hard to compute directly. Instead, we can use a Monte Carlo method:

1. Shuffle the deck and draw the top 3 cards
2. If these cards are all in descending order you win, if not you lose
3. Repeat many times
4. Average the number of wins over the total number of games played

Repeating the game enough times will give a fairly accurate estimate for the probability of winning the game. This is a much simpler method than calculating the probability of all possible winning combinations. Monte Carlo methods are useful in large problems where it would be computationally intensive to calculate every possible outcome.

In Markov Chain Monte Carlo, this method is combined with Markov chains. Each sample is used to obtain the next sample, which is where the Markov chain comes in.[17] Each sample only depends on the previous one. Here, the goal is to create an ergodic Markov Chain that has a steady state distribution P defined as the distribution we want to sample from.[16] For example, if we wanted to sample from the normal distribution, then we'd aim to create a Markov Chain that converges to the normal distribution. This method is useful because it's possible to sample from a specified distribution without needing to calculate the probabilities of being at each state, as just the transition probabilities are required. These transition probabilities are usually a multiple of the sample distribution.[16]

The mixing time of the Markov chain shows the number of samples needed for good convergence. [16] A Markov chain that has mixed should be uncorrelated to its initial state, so given two mixed Markov chains you should not be able to distinguish their initial states. Once a chain has mixed, increasing the length of the chain won't significantly alter the distribution. This means that a good way to check if a Markov chain has mixed is to compare distributions at different points in the chain, and see if they stop changing after a certain length of time.[10]

The Metropolis-Hastings method can be combined with MCMC to transform samples drawn from a known Markov Chain into samples from a chain with a desired stationary distribution.[16] It requires a scores function s that rates each sample based on some metric, and a proposal function Q which is quite often defined by $Q_{xy} = T_{xy}$. Then, at each step of the chain Q is used to generate a proposed state, and the scores of each state are compared. If the score of the new state is larger then it is automatically accepted and that becomes the next step of the Markov chain. If not, the chain stays at the same state with a probability that's proportional to the ratio of the scores. The steps of this algorithm are given here.

At each step of the Markov Chain $X_1, X_2, ...$ set the current state $X_k = y$ and follow these steps:

1. Use Q_{yz} to generate a proposed state z

2. Compute the acceptance probability

$$\alpha = min(1, \frac{s(z)}{s(y)} \frac{Q_{zy}}{Q_{yz}})$$

3. Pick a number β uniformly on [0,1]

4. Set $X_{k+1} = \begin{cases} z & \beta < \alpha \\ y & otherwise \end{cases}$

[16]

This creates a new ergodic Markov Chain that is reversible, which means $P_i T_{ij} = P_j T_{ji}$. This Markov Chain has a steady state distribution proportional to the scores function s.

The Markov Chain Monte Carlo method can be applied to redistricting. The state space can be defined as the set of districting plans. We want to move about the space without encountering "bad" districting plans. To do this, constraints such as contiguity and population balance can be imposed, which will act as the scores functions. Instead of staying at the same state if the proposed districting plan doesn't satisfy the constraints, which will mean the same plan being sampled twice in a row, for the purpose of generating an ensemble of plans it's better to simply propose a new state and try again until a successful move can be made.

In this case the basic algorithm is:

1. Given the current districting plan, modify it to generate a new plan using some proposal function
2. Evaluate the proposed plan according to the chosen constraints
3. If the proposed plan satisfies every constraint then accept it and continue
4. If not, generate a new plan and evaluate again

This builds a chain of districting plans that satisfy all the imposed constraints. The choice of proposal function and constraints are still to be defined.

4 Gerrymandering Metrics

A significant challenge when it comes to gerrymandering is how to define and measure it. Many potential metrics for quantifying gerrymandering have been proposed, with laws and needs differing by State.

4.1 Efficiency Gap

In 2015, Stephanopoulos and Mcghee proposed using "wasted" votes to calculate an efficiency gap. [18] Wasted votes in a district are either votes from the losing party, or votes over the 50% victory threshold from the winning party. Essentially, these are votes that are not required to win. The number of wasted votes for each party are then compared to calculate the efficiency gap as a percentage of the total votes.[19]

The concept of "packing and cracking" involves distributing voters so the number of wasted votes for one party is as small as possible, thus maximising the amount of districts won with the same number of voters. It involves either "packing" the voters for the opposing party into a few districts, or "cracking" the votes up by distributing them sparsely into many districts.[19] The efficiency gap is designed to identify cases where this manipulation has taken place.

For a simple example of how the efficiency gap can be used, take a simple 7×7 grid where each square denotes a population block voting for either Party A or Party B. The voters are distributed in alternating columns, so Party A has 28 voters and Party B has 21. Figure 3 shows two potential districting plans that both split the grid into 7 equal districts, but achieve vastly different election outcomes. To calculate the efficiency gap for each plan, any vote in a winning district that is over the winning threshold of 4 will be counted as a wasted vote, as well as any vote in a losing district.

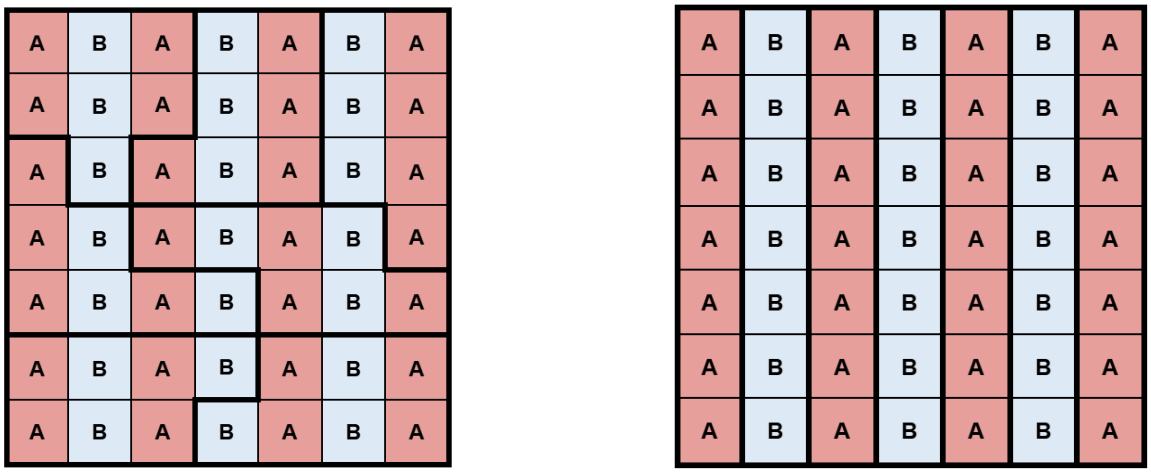


Figure 3: Two potential districting plans for a 7×7 grid with 2 parties. The grid is divided into 7 equal districts outlined in bold. In Plan 1 Party A wins all 7 seats, while in Plan 2 it only wins 4 seats.

Plan 1:

Party	Votes in Losing Districts	Excess Votes in Winning Districts	Total Wasted Votes
A	0	0	0
B	21	0	21

So the total gap in efficiency is $21 - 0 = 21$, which is

$$\frac{21}{49} \times 100 = 42.5\%$$

of the total voters. Plan 1 has an efficiency gap of 42.5% in favour of Party A. Here the efficiency gap has identified the extreme "packing and cracking" in this plan, as the voters for Party B are cracked between districts and can't win a single one.

Plan 2:

Party	Votes in Losing Districts	Excess Votes in Winning Districts	Total Wasted Votes
A	0	12	12
B	0	9	9

So the total gap in efficiency is $12 - 9 = 3$, which is

$$\frac{3}{49} \times 100 = 6.1\%$$

of the total voters. Plan 2 has an efficiency gap of 6.1% in favour of Party B, which is a bit unexpected as Party B won a proportional $\frac{3}{7}$ of the seats with $\frac{3}{7}$ of the voters. This is because parties that have a higher number of voters are more likely to have wasted votes. This is a potential downside of the efficiency gap, as it can sometimes conclude that fair plans are biased against the larger party. However, overall in this example the efficiency gap suggests that Plan 2 is fairer than Plan 1.

Though the efficiency gap appears to be a fairly simple way to measure gerrymandering, it has some drawbacks and can't be relied upon as the sole metric. However, it is a good starting point, and could be used when evaluating ensembles of districting plans to compare efficiency gaps and spot anomalies.

4.2 Isoperimetric Quotient

Other mathematicians have proposed a more geometric approach to measuring gerrymandering, by applying the isoperimetric theorem to districts.[4]

Theorem 2 (Isoperimetric Theorem) *For any planar figure, if A = Area and P = Perimeter then the Isoperimetric Quotient*

$$IQ = \frac{P^2}{4\pi A}$$

is greater than or equal to unity, with equality only if the figure is a circle.

This can be applied to districts by imposing an upper bound on the IQ value allowed for each district, which could prevent irregular shapes and increase compactness. Case calculated that some districts have extremely high isoperimetric quotients, for example the 4th congressional district in Illinois has a boundary perimeter of 116 miles enclosing an area of just 40 miles squared, giving it an IQ over 27.[4] He suggests that an IQ over 5 could indicate gerrymandering. However, Case does note that natural borders can make large perimeters, and therefore high IQ scores, inevitable in states that have lots of lakes and mountains. This bias could skew the results of comparing IQ scores. Therefore, it might not always be the most relevant metric to use.

For an example of the isoperimetric quotient, we can go back to the two districting plans in Figure 3. Each block has a perimeter of 4 units and an area of 1 unit squared. Therefore each district will have an area of 7 units squared. In this example the largest perimeter a district can have is 16, and the smallest perimeter is 12, so the isoperimetric quotients for the districts will be in the range

$$\frac{12^2}{28\pi} = 1.64 < IQ < \frac{16^2}{28\pi} = 2.91$$

In Plan 1, the districts have perimeters $\{12, 12, 12, 12, 12, 12, 14, 14\}$, so the average IQ of the districts is

$$\frac{5(12^2) + 2(14^2)}{28\pi} \div 7 = 1.81$$

In Plan 2, the districts have perimeters $\{16, 16, 16, 16, 16, 16, 16\}$, so the average IQ of the districts is

$$\frac{7(16^2)}{28\pi} \div 7 = 2.91$$

The districts in Plan 1 have a lower average IQ, which suggests that Plan 1 is fairer than Plan 2. This directly contradicts the conclusion reached using the efficiency gap, which shows that different metrics detect gerrymandering differently. This reinforces the need for human judgement to assess gerrymandering, as the metrics don't always agree. It also suggests that one metric alone isn't enough to convincingly detect a gerrymander.

It's worth noting that an alternative version of this metric, called the Polsby-Popper score, is also often used in gerrymandering problems. This is simply defined as the inverse of the isoperimetric quotient. Scores range from 0 to 1, with 0 being the least compact and 1 being the most.[20]

4.3 Partisan Symmetry

A slightly more complicated metric for measuring gerrymandering in partisan symmetry. Partisan symmetry is the principle that a certain share of the votes should translate to a certain share of the seats, regardless of the party.[5] For example, if 55% of the votes earned one party 60% of the seats, then 55% of the vote should also win the other party 60% of the seats. This can be visualised on a seats-votes plot, with the proportion of votes won on the x-axis and the proportion of seats won on the y-axis. One election is then represented by just a point on the graph, and multiple simulations are then plotted using uniform partisan swing.[5] This involves gradually shifting the vote share in each district by the same amount and plotting the results on the graph.

The graph being symmetric around the centre point indicates high partisan symmetry, so an asymmetrical seats-votes curve could indicate gerrymandering. This can be measured by the vertical displacement of the curve from the centre, which is the partisan bias, or the horizontal displacement, which is the mean-median score.[5] The mean-median score is calculated by subtracting the mean number of votes for a party from the median.[21] It shows how much less than 50% of the votes that one party can get while still receiving 50% of the seats. A positive score denotes a bias towards the party being plotted, a negative score denotes a bias towards the other party.[21]

Now we can return to the examples in Figure 3 and calculate the mean-median score for each plan. We'll be looking at the vote share for Party A.

Plan 1: Vote share for Party A = $\{\frac{4}{7}, \frac{4}{7}, \frac{4}{7}, \frac{4}{7}, \frac{4}{7}, \frac{4}{7}, \frac{4}{7}\}$

$$v_{med} = v_4 = \frac{4}{7}$$

$$\overline{v} = \frac{28}{49} = \frac{4}{7}$$

$$\Rightarrow v_{mm} = v_{med} - \overline{v} = \frac{4}{7} - \frac{4}{7} = 0$$

Plan 2: Vote share for Party A = $\{0, 0, 0, 1, 1, 1, 1\}$

$$v_{med} = v_4 = 1$$

$$\overline{v} = \frac{28}{49} = \frac{4}{7}$$

$$\Rightarrow v_{mm} = v_{med} - \overline{v} = 1 - \frac{4}{7} = \frac{3}{7}$$

So Plan 1 has a mean-median score of 0 even though Party A won all of the seats with just $\frac{4}{7}$ of the votes. This is because the vote distribution is completely uniform across all states, so the distribution of vote shares is symmetrical, even though it isn't entirely fair. This is a drawback of the mean-median score, but is unlikely to happen in real life. Plan 2 has a mean-median score of $\frac{3}{7}$ in favour of Party A. In this plan the distribution of vote shares isn't symmetrical at all.

A critique of partisan symmetry in general is that populations themselves can be clustered asymmetrically.[21] For example, Democrats tend to naturally cluster to big cities while Republicans are more spread out in rural areas. In this case, even a completely politically neutral plan could still lack partisan symmetry, so it isn't always the most accurate way to measure gerrymandering. Though there are clear problems with the mean-median score, it can provide useful information about the distribution of voters.

4.4 Conclusion

There are many other metrics for measuring gerrymandering that have been proposed, such as competitive metrics designed to ensure that each district has a fair, competitive voting environment. One of these metrics is the Cook Partisan Voting Index (CPVI), which compares the vote share of a district to the national average, and describes a district as competitive if it's within 5% either way.[22] DeFord et al. analysed various competitive metrics and found that applying universal competitive metrics across the nation, or even statewide, is risky due to political geography.[22] However, they also found that it can still provide useful insight when formulating or comparing redistricting simulations.

Similarly, though many methods for quantifying gerrymandering have been proposed, this section shows that it's hard to find one that satisfies the needs of every single state. Some states do have criteria specified in their districting laws, for example Missouri's 2018 constitutional amendment requires that the efficiency gap of a proposed district be as close to zero as possible.[22] However, overall there is no one metric that has been agreed upon to quantify gerrymandering across the whole United States, and it's unlikely one will be found.

This is where computational sampling comes in. Algorithms can take laws from each individual state into account when sampling plans, which makes them fairly versatile. These samples can then be used to get an idea of a range of possible districting plans, which can be compared to the proposed plan to identify if it's a statistical outlier. This method allows for the different needs of each state, and doesn't rely too heavily on a single metric to quantify gerrymandering.

This section also highlights the different conclusions that metrics can reach. The efficiency gap identified Plan 2 as being the fairest, while the isoperimetric quotient and the mean-median score identified Plan 1 as the fairest. In the case of this example the result from the efficiency gap is probably the most useful, as Plan 1 was a clear case of "packing" voters for Party B into many districts to completely dilute their voting power. The compactness of the plan and the voter distribution weren't as important in this scenario. This shows that the context of the problem is important. Metrics used for analysis must be chosen carefully, as they have different uses and different strengths. A level of human judgement is crucial for identifying gerrymandering.

5 Existing Redistricting Algorithms

In recent years a lot of advances have been made in the field of computational redistricting, especially with respect to MCMC methods. The algorithms all start by viewing redistricting as a graph cutting problem. A state can be represented by a dual graph, where each population unit is a node and edges denote contiguity. Then districts are created by removing edges to partition the graph. Two different methods to do this, "flip" and "recombination", are explored in this section.

5.1 Flip Methods

Flip methods have this name because they involve "flipping" nodes between districts. A flip method was used by Fifield et al. in their article on automated redistricting.[23] This section will explore the algorithm they created.

The first version of this article, released in 2014, was one of the first to propose using Markov Chain Monte Carlo in redistricting simulations. Previous algorithms just used Monte Carlo methods. This algorithm uses a modified version of the Swendsen-Wang method with a Metropolis-Hastings step to obtain a representative sample of valid plans from the Gibbs distribution. The flip step works by performing a breadth-first search on the boundary nodes and randomly choosing some of them for the proposed swap.

The Swendsen-Wang algorithm is a Monte Carlo method used for simulations of large systems. It works by clustering variables based on interactions.[24] This was adapted by Barbu and Zhu to

incorporate a Metropolis-Hastings step, which ensures the Markov chain converges to a specified distribution.[25] The algorithm was further adapted in this paper to include a contiguity restraint, so it rejects a swap that would lead to a non-contiguous district.

They chose the following Gibbs distribution as the target stationary distribution of the Markov chain:

$$f_\beta(\pi) = \frac{1}{z(\beta)}(-\beta \sum_{V_l \in \pi} |\sum_{i \in V_l} \frac{p_i}{\overline{p}} - 1|)$$

Here π is the partition, V_l are the nodes in π, p_i is the population in each node and $\overline{p}$ is the target population. β is called the inverse temperature, and $z(\beta)$ is the normalising function. This distribution includes the equal population constraint, increasing the likelihood that the chain samples plans with equal populations. The distribution is incorporated by using the unnormalised form as the scores function in the acceptance probability.

The resulting chain has a long mixing time, which means it has to be run for a long time to gather a representative sample of plans. A solution proposed in this paper is to use parallel tempering. This is where r copies of the algorithm are run at r different inverse temperatures β, and after a certain number of iterations two adjacent chains swap temperatures. These chains are randomly chosen using a Metropolis criterion. This helps the chains travel through a wider range of districting plans.

A weakness of this algorithm is that it can only contain one constraint at a time. The method of "flipping" nodes has also been criticised by Duchin et al., as it can take a long time to reach a plan significantly different from the initial plan. In a simple 100×100 grid toy example they demonstrated that even after 1,000,000 steps it didn't produce very diverse results.[6] This poor mixing time led to an alternative partitioning approach being proposed.

5.2 Recombination Methods

Instead of "flipping" nodes in the graph, Duchin et al. proposed an alternative method of "recombination", which involves combining two districts into a subgraph and then randomly repartitioning them.[6] The algorithm uses a spanning tree method to repartition the subgraph. A spanning tree is a graph with n vertices, $n-1$ edges and no cycles.

In this algorithm a spanning tree is drawn on the merged districts, which is then repartitioned by cutting an edge. The two merged districts are chosen at random and the spanning tree is drawn using a modified version of Wilson's algorithm, which takes all the possible spanning trees on the subgraph and uniformly samples one. An edge is then cut to ensure both districts satisfy population constraints. If this isn't possible then a new spanning tree is drawn, and if multiple cuts could be made then one is chosen uniformly.

Instead of the Gibbs distribution targeted in the flip algorithm, here constraints are incorporated by targeting the spanning tree distribution. This is because choosing a uniform distribution to target often results in non-compact districts. The Gibbs distribution would need to take many samples in order to avoid this issue, giving it a very long run time. In contrast, spanning tree weighting favours compact districts as it tends to find clusters in the graph, and can work with fast algorithms.

In the 100×100 grid study mentioned in the previous section, two initial seeds were created. In one there were 10 horizontal districts, and in the other there were 10 vertical districts. After just 10,000 recombination steps the two seeds gave similar results and were uncorrelated to their initial seeds, suggesting they had mixed. This demonstrates that recombination methods converge to a steady state distribution much faster than flip methods.

This method was used by the researchers to analyse the districting plans of Virginia, mentioned earlier in this paper as being ruled to be unconstitutional due to the "packing" of BVAP. The algorithm was run for 20,000 steps with 100 neutral initial seeds. They winnowed the results to just include districts with $\leq$60% BVAP and plotted the results for each district on a boxplot. This

helped to visualise the general distribution of BVAP in the ensemble of plans generated. Plotting the plans deemed unconstitutional on this graph showed that all the 11 districts were outliers. This demonstrates that ensemble methods can be useful for identifying plans that are statistically unlikely, and therefore perhaps examples of gerrymandering.

Overall recombination appears to be the superior method for gathering a representative sample of plans. Though a recombination step takes about 1000 times longer to compute than a flip step, in just a matter of hours it was able to produce a large number of potential districting plans.[10] The researchers who developed this algorithm recommend looking at other balanced bipartitioning methods to replace spanning trees, subjecting them to similar tests of quality. This will be the focus of the remainder of this project.

5.3 Limitations of the Recombination Method

From now on the recombination algorithm will be referred to as ReCom. It's worth noting that ReCom is not a traditional MCMC method as it does not have a Metropolis-Hastings step. This is because the Markov chain isn't reversible, so it can't actually be used to sample from a specified distribution.

ReCom was modified by Autrey et al. to make the algorithm reversible.[26] They did this by expanding the state space from the space of all graph partitions to the space of all spanning forests. If Q is the proposal function proposing a move from one state to the next, T and π are the current spanning forest and graph partition respectively, and T' and π' are the proposed spanning forest and graph partition, then $Q(T, T')$ and $Q(T', T)$ are a lot easier to calculate than $Q(\pi, \pi')$ and $Q(\pi', \pi)$. The overall method for the proposal algorithm is broadly similar to the original method, but this change makes the Markov chain reversible. This reversibility allowed them to introduce a Metropolis-Hastings step.

As described before, the Metropolis-Hastings step introduces an acceptance probability to produce a new Markov chain that converges to a desired distribution P. In this paper the acceptance probability is written as:

$$A(T, T') = min(1, \frac{P(T')}{P(T)} \frac{Q(T', T)}{Q(T, T')})$$

So each move is accepted with a probability $A(T, T')$ and rejected with probability $1 - A(T, T')$. This Metropolis-Hastings step allows you to sample from a specified probability distribution, which is a property that the original ReCom algorithm lacks.

The original research group did tackle this problem by creating their own reversible recombination chain, RevReCom.[27] The original method does come close to approximating the spanning tree distribution, but this is only provably true for 2 districts. Once there are more than 3 districts there is no closed form of the distribution. However, by introducing reversibility through adding more rejection conditions they found that the chain did converge to the target spanning tree distribution as desired. They also found that the results using the RevReCom Markov chain were broadly similar to the results obtained from the ReCom chain. This suggests that, for the purposes of this project, this limitation can be overlooked.

6 Spectral Recombination

The main steps of the ReCom Markov chain are first merging two districts into a subgraph, and then repartitioning this subgraph into two new districts. The partitioning method used in the original paper is to draw a spanning tree and uniformly sample a random eligible edge to delete. The rest of this paper will adapt the algorithm to use a different partitioning method and explore how this affects the results.

6.1 Spectral Clustering

Spectral techniques were first used to partition graphs by Donath and Hoffman in 1972, and built on by Fiedler in 1975.[28] Spectral clustering is a graph bisection problem, where the goal is to split a graph $G(V, E)$ in two while minimizing the cut edges.[28]

To find this cut we first need to find the Laplacian matrix. Let A be the adjacency matrix of the graph where

$$a_{ij} = \begin{cases} w & \text{if there's an edge between nodes } i \text{ and } j \text{ with weight } w \\ 0 & \text{otherwise} \end{cases}$$

Let D be the diagonal degree matrix where d_{ii} is the sum of the i-th row of A. Then the unnormalised Laplacian matrix L can be defined as:

$$L = D - A$$

[28]
There are different versions of the normalised Laplacian, but for the purposes of this paper we'll calculate it as

$$L_{norm} = I - D^{-1/2} A D^{-1/2}$$

The goal of this method is to partition V into disjoint sets U and its complement $\overline{U}$. For notation let:

$$|U| = \text{the number of vertices in } U$$

$$\text{vol}(U) = \sum_{i \in A} d_i, \text{which is the the weight of all edges attached to nodes in } U$$

$$\delta(U) = \text{the set of all edges that have just one end in } U, \text{ so the other is in } \overline{U}$$

.
The algorithm aims to minimise either the Ratio Cut (Hagen and Kahng 1992):

$$\frac{\text{vol}(\delta(U))}{|U| \cdot |\overline{U}|}$$

when the Laplacian is unnormalised or the Normalised Cut (Shi and Malik 2000):

$$\frac{\text{vol}(\delta(U))}{\text{vol}(U)\text{vol}(\overline{U})}$$

when the Laplacian is normalised.[28][29] The difference is that the Ratio Cut measures the subset by the amount of nodes, while the Normalised Cut measures the subset by the weights of the edges. Both of these methods could have advantages in redistricting. The Ratio Cut could help ensure balanced nodes, while the Normalised Cut could ensure balanced weighting based on any desired metric.

The eigenvector z_2 corresponding to the second smallest eigenvalue λ_2 of L is known as the Fiedler vector, which is the solution to the Ratio Cut optimisation problem.[28] For the Normalised Cut optimisation problem, the Fiedler vector is the second smallest eigenvector of L_{norm}.[28] There are a few different ways this can be used to partition the graph. One method is finding the median value $\overline{m}$ of the vector z_2 and then assigning the nodes with corresponding values in the Fiedler vector

$\leq \overline{m}$ to U and the rest to $\overline{U}$.[30] Another method is assigning the nodes to either U or $\overline{U}$ based on their corresponding sign in the Fiedler vector.[29]

Spectral clustering has many properties that suggest it could be useful for redistricting. The first is that it aims to minimise the number of cut edges. This is likely to favour both compactness and contiguity. It also naturally divides the graph into balanced partitions, which can help to ensure population balance. The weights on the edges between the nodes can be adjusted to represent various metrics depending on the researcher's goal. For instance, political factors, such as the levels of BVAP voters, or simple geographic factors, such as mountains dividing districts.

These properties suggest that spectral clustering could be an appropriate graph partitioning method to implement in place of the spanning tree bipartitioning method.

6.2 Toy Example

The code in this section utilises the GerryChain Python package developed by the researchers at the Metric Geometry and Gerrymandering Group (MGGG) in 2018. The package can be found here: `https://github.com/mggg/GerryChain`[31]

To first get an understanding of how spectral clustering will partition the graphs differently to the spanning tree method used in the original ReCom algorithm, we can look at a small toy example. Take an 8×8 grid with 8 vertical districts and run the Markov chain for 40 steps. The underlying graph called "Gridlandia" (displayed in Figure 4) and the code for the original ReCom chain were taken from the GerryChain documentation.[32] The Spectral ReCom chain was created by adapting the spectral cut algorithm included in the GerryChain python package to incorporate a constraint for balanced populations. A loop was added to ensure that if the algorithm produced an invalid cut, such as one that didn't have balanced populations, then it would select another random pair of districts and try again. Like the original algorithm, the spectral algorithm generates random weights between 0 and 1 on the graph edges because there are no specified constraints other than contiguity and balanced population at this moment. When further constraints are incorporated, these weights can be updated accordingly. The laplacian was chosen to be normalised.

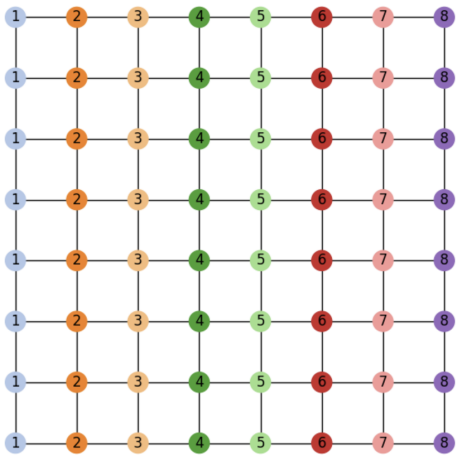


Figure 4: "Gridlandia", an 8×8 grid made up of 8 vertical districts with 8 nodes of equal population

There are two elements of randomness in this algorithm. The first is the random weighting assigned to the graph edges at each step of the Markov chain in order to help the algorithm to find more diverse ensembles. The second is the choice of which districts to merge and split at each step. In order to make this code reproducible, the random seed 2024 was chosen to ensure it makes the same random decisions every time. The results obtained at steps 0, 20 and 40 of the Markov chains

in both the original ReCom and the Spectral ReCom algorithms can be seen in Figures 5 and 6.

In Figure 5 it can be seen that the original ReCom algorithm produces fairly diverse plans, with the grid changing quite a lot between steps 20 and 40. In contrast, Figure 4 shows that the actual layout of the districts produced by the Spectral ReCom algorithm doesn't change much between steps 20 and 40. This lack of diversity in the districting plans is a weakness, as the ultimate goal is to produce a diverse ensemble of potential districting plans for comparison. However, these districts are generally more compact and less straggly than the ones produced by the original algorithm, which is a desirable trait in redistricting. Many states specify that compactness should be prioritised when possible.

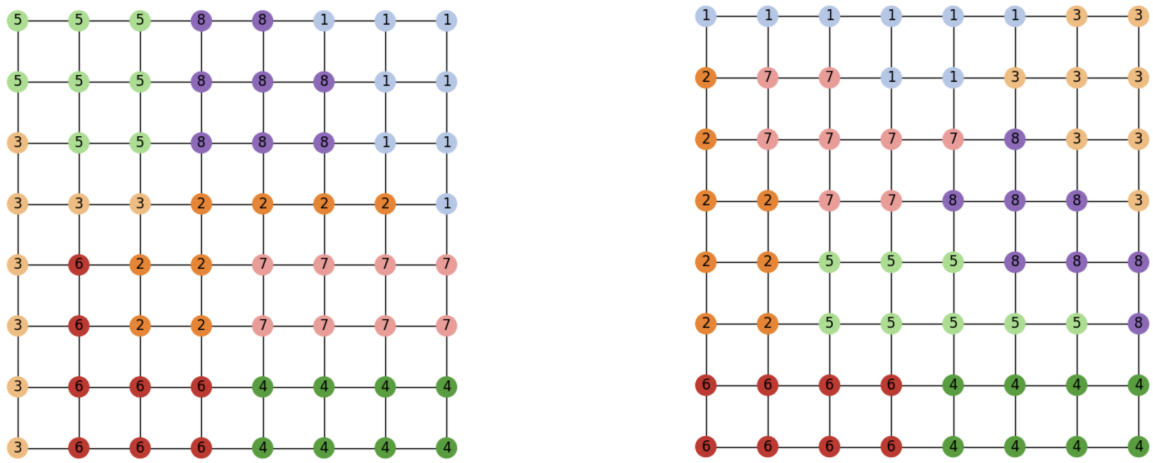


Figure 5: Results from performing the original ReCom algorithm on the toy grid at A) Step 20 and B) Step 40 of the Markov chain

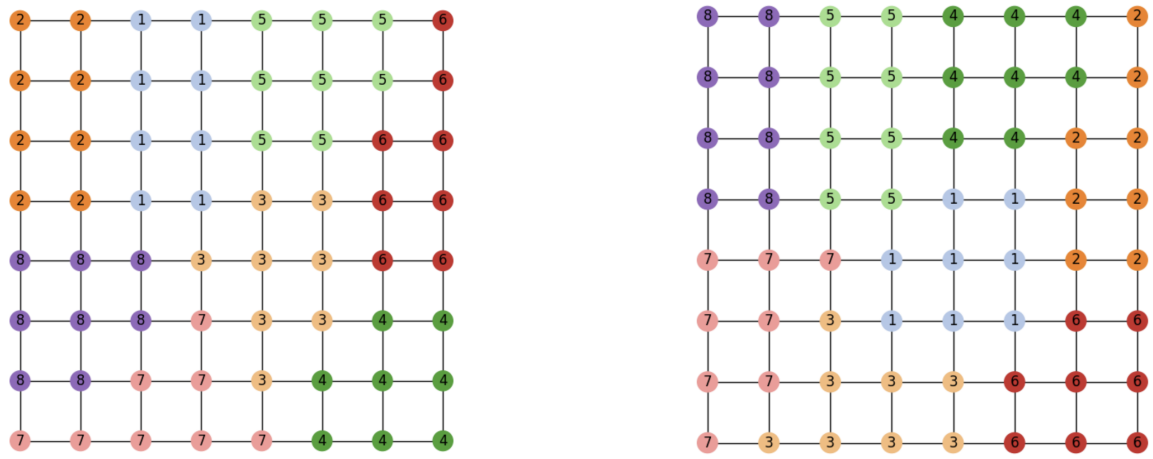


Figure 6: Results from performing the Spectral ReCom algorithm on the toy grid at A) Step 20 and B) Step 40 of the Markov chain

Overall, the new Spectral ReCom algorithm appears to favour compactness at the expense of diversity. This is due to the nature of the spectral cut, which tends to find compact clusters by

minimising the Normalised Cut.

6.3 Toy Example with Underlying Geography

In real world scenarios there will be geographical factors such as rivers and mountains that we may want to avoid splitting between districts. To simulate this, a "region surcharge" can be added to the toy grid. This increases the importance of the graph edges based on a specified factor, which is equivalent to adding weighting to the edges.

In the example given in the GerryChain documentation, the people of Gerrymandria decide they would like to keep municipalities together in the same districts if possible, but more importantly they'd like to keep water districts together.[32] The graphs of these underlying geographies are shown in Figure 7.

In the original ReCom algorithm they added a region surcharge that gave a 0.2 weighting to municipality boundaries and 0.8 weighting to the water district boundaries. This works by increasing the weight of edges that pass between these regions, ensuring that the algorithm prioritises keeping connected nodes together. Spectral partitioning finds natural clusterings in the graph by ensuring the weights of edges between groups are low and within groups are high. This property suggests it could be useful for finding valid partitions within these regional constraints.

To add a region surcharge to the spectral algorithm, every edge was given a random weight between 0 and 1 like before, and additional weights were then added according to the properties. While the original algorithm added weight to edges that crossed the municipality and water district boundaries, the spectral algorithm required that weight be added to the edges within these boundaries instead. This is because spectral clustering finds groups with high weighting within each group, so it will prioritise keeping nodes connected by edges with high weights together. The Markov chain had to be run a lot longer, for 10,000 steps, in order to ensure it had enough time to adapt to these weightings.

An obstacle found straight away was that the Spectral ReCom algorithm did not always run to completion. When the algorithm was run with random seed 2025, it only made it to step 4164 of the Markov chain. This is likely because the spectral cut is a stronger condition than the spanning tree cut. As the spectral cut relies on the laplacian of the graph, there are fewer options for valid bipartitions within the population constraint on the subgraph. In contrast, there are many spanning trees than can be drawn on the subgraph, so there are many more valid cuts available in the original ReCom chain. It appeared to be a lot harder for Spectral ReCom to find new partitions with both the population and regional constraints.

The limited spectral cuts available is a significant weakness of the adapted algorithm. In a real world scenario the chain will need to have thousands of steps to get a diverse ensemble of districting plans. However, this could just be a side effect of the natural grouping properties of the clustering algorithm, which may be desirable when it comes to redistricting.

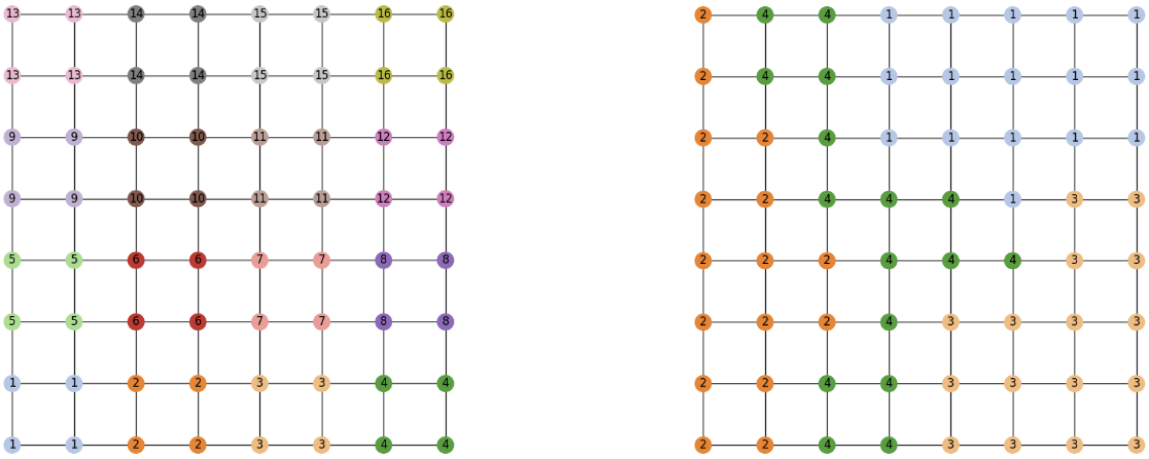


Figure 7: A) Municipalities and B) Water Districts of Gerrymandria

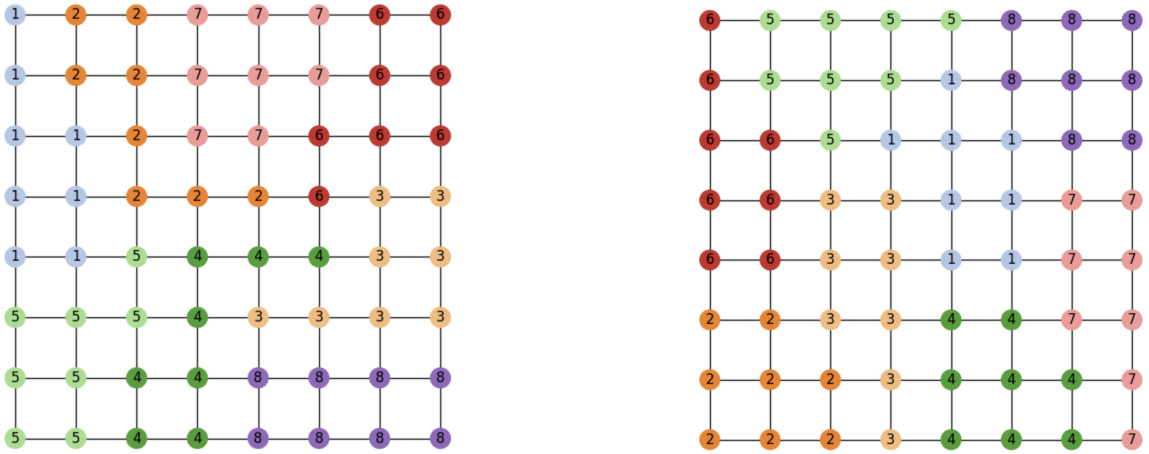


Figure 8: Results after 10,000 steps of the A) Original ReCom algorithm and B) Spectral ReCom algorithm, both with regional surcharges added for municipalities and water districts

In Figure 8 it can be seen that both algorithms adapt to produce districts that prioritise keeping municipalities and water districts together. The original ReCom algorithm does a very good job of this, with every single district being within one water district, while only splitting half the municipalities up. The Spectral ReCom algorithm is a bit less successful. While it manages to keep 9 of the 16 municipalities intact, only half of the districts are contained within one water district. This may be because the irregular shape of the water districts are at odds with the compact nature of the Spectral algorithm. Overall, it can be seen that the original ReCom algorithm is better at adapting to the underlying geography of the graph, though the Spectral algorithm still does a decent job.

7 Case Study: Pennsylvania's Congressional Districts

For an example of how MCMC algorithms, specifically the new Spectral ReCom algorithm, can detect gerrymanders in real life scenarios, we can look at the 2011 Pennsylvania congressional district plan.

In 2011 Pennsylvania had both a Republican governor and a Republican majority in the State Senate.[33] This essentially gave the Republican party full control when creating the new map of congressional districts. The new plan, shown in Figure 9, caused backlash for its Republican bias, with even Republican Senator Mike Folmer saying he suspected it was drawn with the intent to dilute the Democratic vote.[34] Democratic Senator Andrew Dinniman complained that the 7th congressional district, seen in the bottom right of Figure 9, looked like a "three-headed dog".[34]

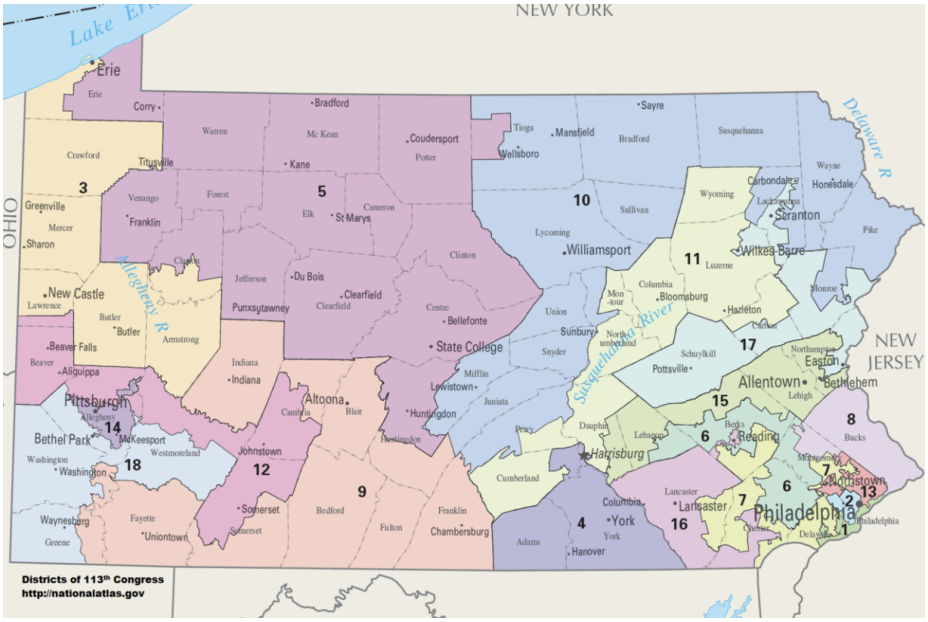


Figure 9: 2011 Congressional District Map of Pennsylvania. By Department of the Interior - National Atlas of the United States

However, despite this backlash the congressional district plan was approved and remained in place for over 6 years. Eventually the League of Women Voters filed a lawsuit on the 14th of June 2017.[35] Though a Pennsylvania trial court initially ruled that it didn't violate state law, on January the 22nd 2018 the Pennsylvania Supreme Court ruled it was a "severe and durable" gerrymander biased towards the Republican party that violated the state constitution.[33]

To assess the bias of the enacted plan, the Spectral ReCom algorithm can be used to generate 1000 potential plans, and then the Democratic vote in each district can be compared using a boxplot. The boxplot created using the original ReCom algorithm was made from code provided in the GerryChain documentation, which was then adapted to incorporate the Spectral ReCom algorithm.[32] The dataset for Pennsylvania was obtained from the GerryChain GitHub page.[31] Data from the 2012 election for State Senate was used to give voting data to each population block. In this election the Democratic candidate Bob Casey Jr. won with 53.69% of the vote compared to the Republican candidates 44.59%.[36] It's worth noting that we are not looking at the number of seats won to assess bias, rather we're looking at whether the Democratic vote in the enacted plan is different to the ensemble of plans.

An important factor to consider is the population constraint. The algorithm contains an epsilon that allows the population in the new districts to deviate slightly from the average, for example

$\epsilon = 0.01$ allows each district to deviate $\pm 1\%$ from the target average population, giving a total population deviation of 2%. This is fine for legislative districts where the population is allowed to deviate by up to 10%, as established in Brown v. Thompson.[37] However, the population constraint is much tighter for congressional districts. In Wesberry v. Sanders in 1964 it was ruled that congressional districts should keep populations as near to equal as possible.[38] A few years later in Kirkpatrick v. Preisler a congressional plan was thrown out because its total population deviation was 5.97%.[39]

The original ReCom algorithm can run this simulation with $\epsilon = 0.01$ in a matter of minutes. However, running the adapted Spectral ReCom algorithm with an ϵ value any less than 0.025 results in the code crashing after only about 20 steps of the Markov chain. This is because the spectral cut is a stronger condition, and the code struggles to find a valid cut that still satisfies the population constraint. For this reason, the population deviation had to be set to $\pm 2.5\%$. This is not ideal for congressional districts and suggests that the Spectral ReCom algorithm may not be very useful in real world scenarios. However, a total population deviation of 5% is less than in the Kirkpatrick v. Preisler case, so it can be argued that for the purposes of this project it is small enough.

Figures 10 and 11 show the graphs obtained from running both the original ReCom and Spectral ReCom algorithms for 1000 steps to obtain 1000 potential districting plans. While this task only took the original algorithm 5 minutes to run, it took the spectral algorithm 780 minutes. This is a significant increase, which suggests that Spectral ReCom is more computationally intensive. This is probably due to the laplacian of the merged subgraph needing to be computed potentially multiple times at each step, and the algorithm needing more attempts to find a cut to satisfy the population constraint. This is a significant downside of Spectral ReCom.

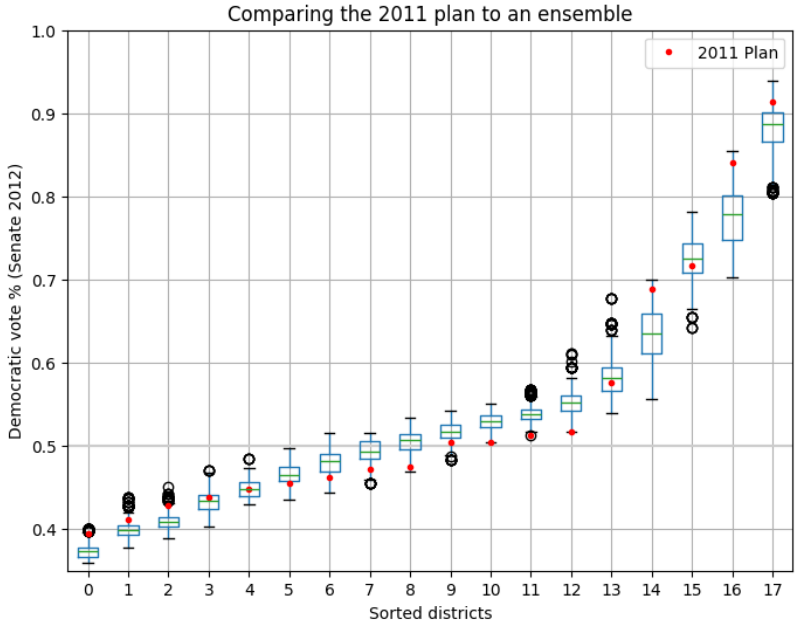


Figure 10: Boxplot created using the original ReCom algorithm. The black circles show outliers and the red dots show the enacted 2011 plan. The boxplot suggests that the Democratic vote is lower in some districts than expected, and higher in others.

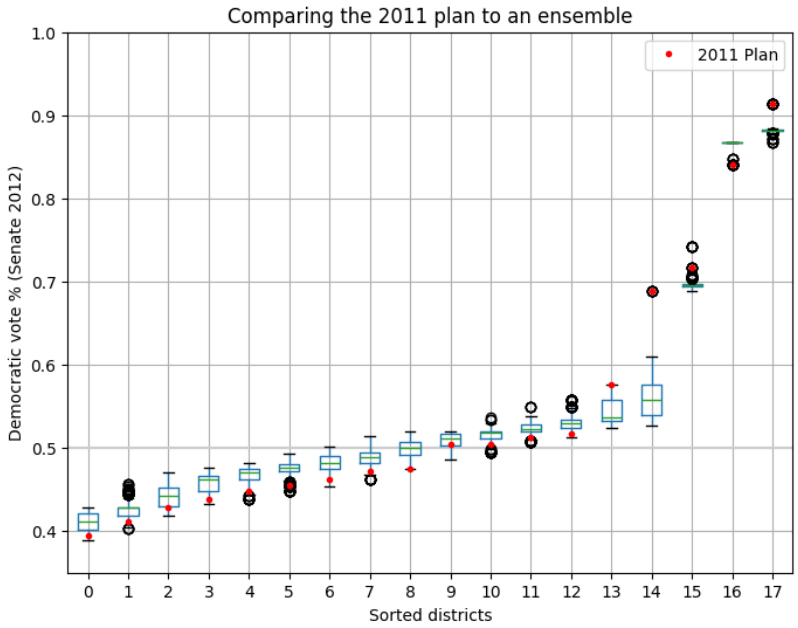


Figure 11: Boxplot created using the Spectral ReCom algorithm. It shows the same general trend as Figure 10, but the boxes are a lot smaller.

Figure 10 displays the boxplot created using the original ReCom algorithm. Here it can be seen that in a lot of the middle districts, especially districts 8, 10, 11 and 12, the enacted plan has a far lower Democratic vote than expected. These districts all have a Democratic vote close to the minimum score out of all 1000 plans, with the plan for district 11 being identified as an outlier. There are also some districts with a higher Democratic vote than expected. These are districts 0, 2, 14 and 16 where the enacted plan has a Democratic vote near the maximum of the ensemble of plans. This suggests that the Democratic vote has been deliberately raised in some districts to decrease the vote in others, as the vote distribution in the enacted plan appears to be statistically unlikely compared to the ensemble of plans. This indicates gerrymandering.

Figure 11 shows the boxplot created using the Spectral ReCom algorithm. Here the boxes are generally smaller, which is likely due to the lack of variety in the ensemble of plans created using this algorithm. The boxes for districts 15, 16 and 17 are extremely small, which suggests that the algorithm didn't affect them much. This could potentially be because it couldn't find a valid cut when merging any of these districts, so it moved onto another pair. This means that the data for the last three districts isn't very helpful.

Overall though the spectral results do show a similar trend to the original. It can once again be seen that the Democratic votes in the middle districts are generally a lot lower than expected, whereas in other districts such as district 14 it is much higher. The results appear more extreme in the spectral version, with hardly any of the districts from the 2011 plan in the interquartile range. This is potentially due to the lack of variety in the districting plans created using the Spectral ReCom algorithm. This means that the range of plans isn't that large, resulting in smaller interquartile ranges. This makes it more likely that the enacted plans will appear as outliers.

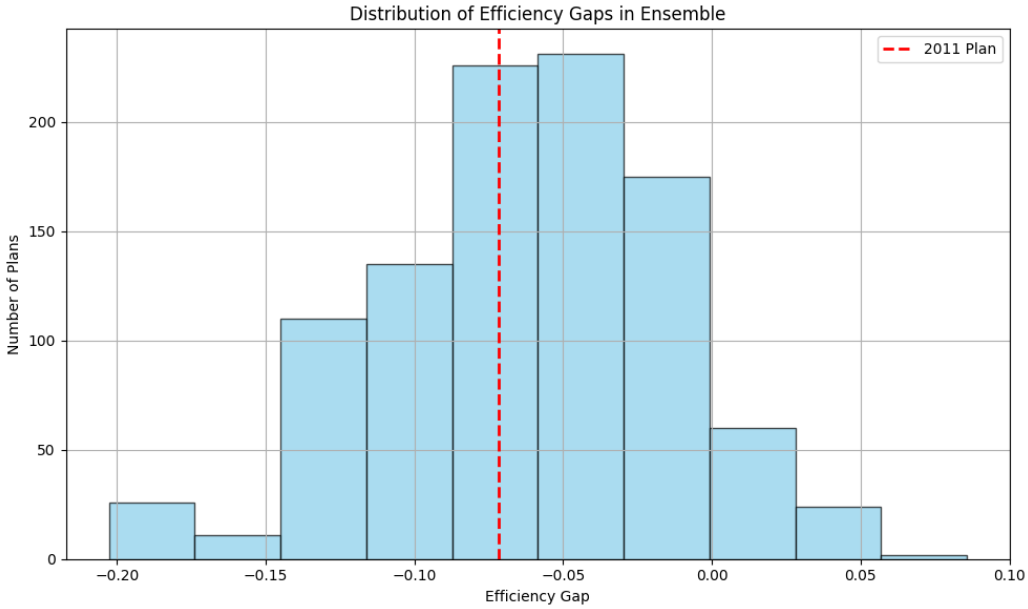


Figure 12: Histogram showing the efficiency gaps of the ensemble of plans. The red line denotes the efficiency gap of the 2011 plan. Most of the plans appear to have a bias towards the Republican party, so the 2011 plan doesn't appear to be an outlier.

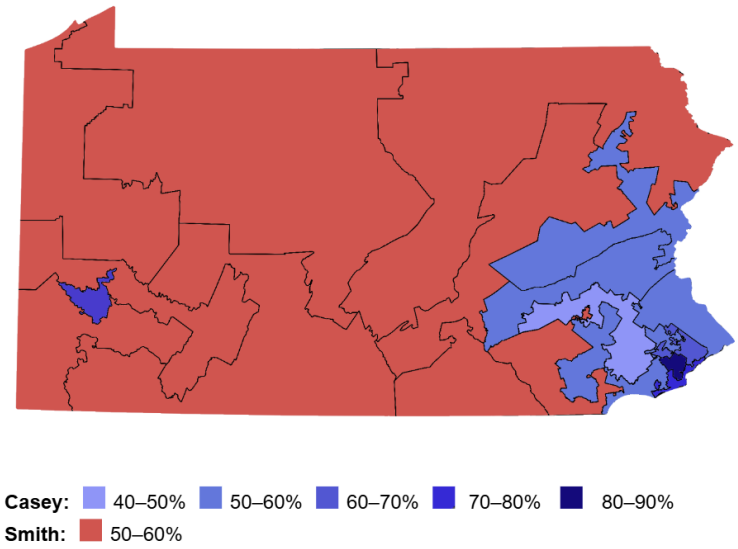


Figure 13: Map of Pennsylvania displaying the vote distribution from the 2012 State Senate election. Casey was the Democratic candidate and Smith was the Republican. Data was obtained from the Pennsylvania election returns website[36]

Figure 12 shows a histogram of the efficiency gaps from the ensemble created using the spectral algorithm. The efficiency gap for the 2011 plan is -0.072 , which is a 7.2% bias towards the Republican party. Here it can be seen that this efficiency gap is broadly what it would be expected to be, which appears to contradict the conclusion that the plan is a gerrymander. However, the majority of the

plans have a negative efficiency gap, showing an overall bias towards the Republican party. This suggests that the bias is almost inevitable due to outside factors. This may be because of voter distribution. As can be seen in the graph in Figure 13, the Republican voters in Pennsylvania tend to be evenly spread out and win their districts by tighter margins with just $50 - 60\%$ of the vote in every winning district. In contrast, the Democrat voters are more concentrated in cities such as Pittsburgh and Philadelphia, winning these districts by landslides. This naturally leads to more Democratic votes being wasted and therefore a larger efficiency gap, regardless of the enacted plan. This suggests that the efficiency gap may not be the most useful metric for assessing gerrymandering in this situation.

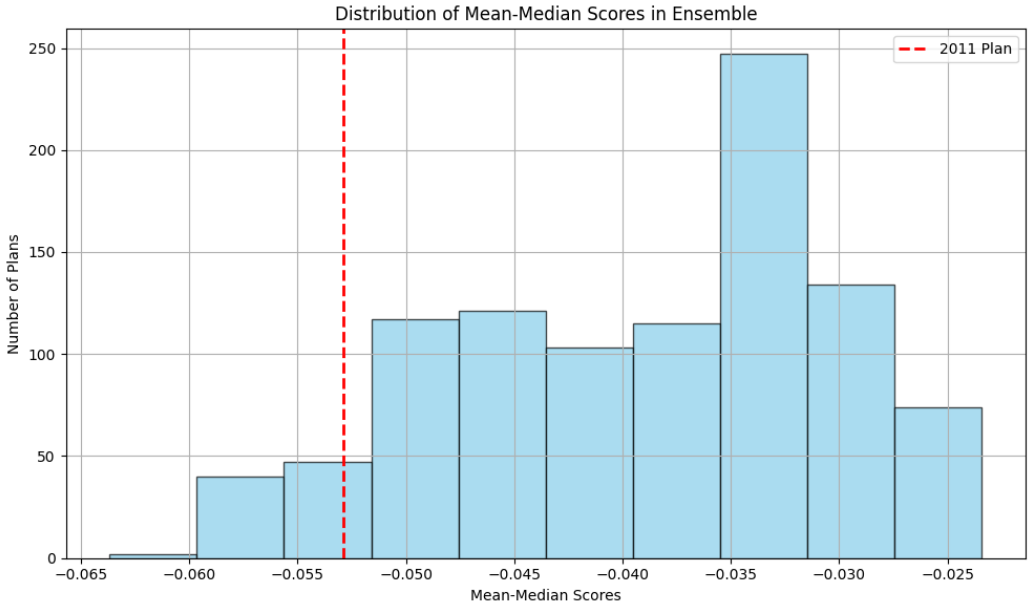


Figure 14: Histogram showing the mean-median scores of the ensemble of plans. The red line denotes the mean-median score of the 2011 plan, which is lower than most.

Figure 13 shows the histogram of the mean-median scores from the ensemble created by the Spectral ReCom algorithm. The scores here are also all negative, likely due to the voter distribution again. However, here the 2011 plan does appear to be an outlier, with its mean-median score of -0.053 being lower than most of the other plans. This means that the median Democratic vote share is 5.3% lower than the mean Democratic vote share, suggesting that Democratic voters are packed into a few districts in order to reduce the Democratic vote in others. This agrees with the results from the boxplots, where it can be seen that the Democratic vote is a lot lower than expected in some districts and higher than others. This suggests that the plan was made deliberately to dilute the Democratic vote, and therefore is a gerrymander.

Figures 12 and 14 reinforce the idea that many of the proposed metrics for measuring gerrymandering don't always agree and will identify different plans as outliers. This shows that, though these metrics can help inform us about different plans, it is important to have an element of human judgement to consider the needs and political geography of each state and decide which metrics to use. In this case study the efficiency gap could not provide much information due to the distribution of voters. However, the mean-median score used alongside the boxplot of votes per district built a convincing argument that the 2011 plan was indeed a gerrymander in favour of the Republican party. This shows the strengths of using MCMC algorithms to build a large ensemble of districting plans to assess gerrymandering. It allows us to get a full overview of what would be expected from

an average districting plan, which can then show if the enacted plan is an outlier.

While the modified Spectral ReCom algorithm is effective at building a large ensemble of districting plans, the lack of diversity and long runtime suggests that the original spanning tree ReCom algorithm is better in most cases. However, in cases where compactness is prioritised the Spectral ReCom algorithm could be useful.

8 Conclusion

The findings of this project reinforce the idea that existing metrics for measuring gerrymandering aren't very useful on their own. As seen in both the toy example in Section 4 and the case study in Section 7, they tend to disagree on whether a plan is a gerrymander and can't take into account the natural political geography of a state. In contrast, MCMC methods generate a large ensemble of plans, allowing for more comparison and analysis. This shows that computational methods give a context to the issue that a single metric can't. An element of human judgement is also necessary to decide which metrics can provide relevant information to each situation.

In Section 6 we modified the ReCom algorithm by DeFord et al. to have a spectral cut instead of a spanning tree cut. Through testing on a toy example we found that, though the modified Spectral ReCom algorithm created very compact districts, the ensemble of plans lacked diversity and had a much longer run time than the original algorithm. The lack of diversity makes it less useful as a tool for comparison, and the long run time could be a significant issue in real world scenarios with more data. Therefore, though the spectral algorithm has its strengths, the original ReCom algorithm is still superior.

Similar issues with the Spectral ReCom algorithm were found in Section 7, but despite these drawbacks it still provided useful information about the enacted plan. Comparing the Democratic vote distribution and the mean-median scores of the ensemble of plans against the enacted plan showed that the 2011 Pennsylvania congressional district plan was potentially an outlier. This further demonstrates the usefulness of computational methods in identifying districting plans that may be outliers, offering context to the issue that other mathematical methods for detecting gerrymandering can't provide. Therefore, these computational methods can be used alongside other evidence to identify gerrymandering.

Overall, computational redistricting is an interesting field with lots of potential. Gerrymandering cases will always need human judgement to concretely identify a gerrymander due to the social complexity of the issue. However, MCMC methods are useful tools that can provide extra information and help decisions be made.

References

[1] Ismar Volić. Gerrymandering Isn't New — But Now We Have a Solution. *Time*, 2024.

[2] Moon Duchin. Geometry v. Gerrymandering. In *The best writing on mathematics 2019*, pages 1–11. Princeton Univ. Press, Princeton, NJ, 2019.

[3] Erica Klarreich. How to Quantify (and Fight) Gerrymandering. *Quanta Magazine*, 2017.

[4] James Case. Flagrant Gerrymandering: Help from the Isoperimetric Theorem? *SIAM News*, 40(9), 2007.

[5] Moon Duchin. Gerrymandering metrics: How to measure? What's the baseline? *arXiv preprint arXiv:1801.02064*, 2018.

[6] Daryl DeFord, Moon Duchin, and Justin Solomon. Recombination: A family of Markov chains for redistricting. *Harvard Data Science Review*, 3(1):3, 2021.

[7] Becky Little. How Gerrymandering Began in the US. *History.com*, 2023.

[8] History.com Editors. Voting Rights Act of 1965. *History.com*, 2023.

[9] Noah Giansiracusa and Cameron Ricciardi. Geometry in the Courtroom. *The American Mathematical Monthly*, 125(10):867–877, 2018.

[10] Daryl DeFord, Moon Duchin, and Justin Solomon. Comparison of districting plans for the Virginia House of Delegates. *MGGG Technical Report*, 12(II), 2018.

[11] The Supreme Court of the United States. Bethune-Hill v. Virginia State Board of Elections, vol. 580. US, 2017.

[12] James R Norris. Markov Chains. Cambridge university press, 1998.

[13] Somenath Biswas. Various proofs of the fundamental theorem of markov chains. *arXiv preprint arXiv:2204.00784*, 2022.

[14] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[15] Rebecca S Wills. Google's pagerank: The math behind the search engine. *Math. Intelligencer*, 28(4):6–11, 2006.

[16] Daryl DeFord. Introduction to Discrete MCMC for Redistricting, 2019. Lecture notes for the 2019 MIT IAP course on Computational Approaches for Political Redistricting.

[17] Don Van Ravenzwaaij, Pete Cassey, and Scott D Brown. A simple introduction to Markov Chain Monte-Carlo sampling. *Psychonomic bulletin & review*, 25(1):143–154, 2018.

[18] Nicholas O Stephanopoulos and Eric M McGhee. Partisan gerrymandering and the efficiency gap. *U. Chi. L. Rev.*, 82:831, 2015.

[19] Patrick Honner. The Math Behind Gerrymandering and Wasted Votes. *Quanta Magazine*, 2017.

[20] Karl-Dieter Crisman and Michael A Jones. The Mathematics of Decisions, Elections, and Games, volume 624. American Mathematical Society, 2014.

[21] Daryl DeFord, Natasha Dhamankar, Moon Duchin, Varun Gupta, Mackenzie McPike, Gabe Schoenbach, and Ki Wan Sim. Implementing partisan symmetry: Problems and paradoxes. *Political Analysis*, 31(3):305–324, 2023.

[22] Daryl DeFord, Moon Duchin, and Justin Solomon. A computational approach to measuring vote elasticity and competitiveness. *Statistics and Public Policy*, 7(1):69–86, 2020.

[23] Benjamin Fifield, Michael Higgins, Kosuke Imai, and Alexander Tarr. Automated redistricting simulation using Markov chain Monte Carlo. *J. Comput. Graph. Statist.*, 29(4):715–728, 2020.

[24] Robert H. Swendsen and Jian-Sheng Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Phys. Rev. Lett.*, 58:86–88, Jan 1987.

[25] Adrian Barbu and Song-Chun Zhu. Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1239–1253, 2005.

[26] Eric Autry, Daniel Carter, Gregory J. Herschlag, Zach Hunter, and Jonathan C. Mattingly. Metropolized Forest Recombination for Monte Carlo Sampling of Graph Partitions. *SIAM Journal on Applied Mathematics*, 83(4):1366–1391, 2023.

[27] Sarah Cannon, Moon Duchin, Dana Randall, and Parker Rule. Spanning tree methods for sampling graph partitions, 2022.

[28] Maxim Naumov and Timothy Moon. Parallel spectral graph partitioning. *Nvidia technical report*, 2016.

[29] Ulrike von Luxburg. A Tutorial on Spectral Clustering, 2007.

[30] Aydın Buluç, Henning Meyerhenke, Ilya Safro, Peter Sanders, and Christian Schulz. *Recent advances in graph partitioning*. Springer, 2016.

[31] Metric Geometry and Gerrymandering Group. GerryChain: A library for building Markov chains to sample redistricting plans, 2018. Accessed: 2025-04-24.

[32] Metric Geometry and Gerrymandering Group. GerryChain Documentation, 2018. Accessed: 2025-04-24.

[33] Jonathan R. Cervas and Bernard Grofman. Tools for identifying partisan gerrymandering with an application to congressional districting in Pennsylvania. *Political Geography*, 76:102069, 2020.

[34] Bram Bumstead. Pennsylvania Senate panel approves new congressional map. *Trib Live*, 2011.

[35] The Supreme Court of Pennsylvania. League of Women Voters et al. v. Commonwealth of Pennsylvania et al. 2018. No. 159 MM, 2017.

[36] Pennsylvania Department of State. 2012 General Election (Official Returns), 2012. Accessed: 2025-04-24.

[37] The Supreme Court of the United States. Brown v. Thomson, 462 U.S. 835, 1983.

[38] The Supreme Court of the United States. Wesberry v. Sanders, vol. 376. US, 1964.

[39] The Supreme Court of the United States. Kirkpatrick v. Preisler, 394 US 526, 1969.