Chapter 17

Random walks and the universe of districting plans

DARYL DEFORD AND MOON DUCHIN

CHAPTER SUMMARY

Random sampling is a key idea across this book, and a leading way to do that is to let a "random walker" loose in your universe to collect samples as they explore. The mathematical framework for this is called Markov chains. This chapter is the place where we dig into Markov chains and MCMC: the motivation, the theory, and the application to redistricting.

1 OVERVIEW: NOT A SOLVED PROBLEM

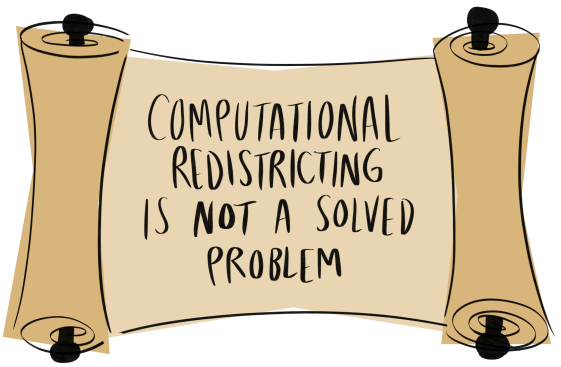
This book has already described many ways in which the modern computing era has revolutionized redistricting: on one hand, an explosion in the sheer amount and diversity of data that map drawers are able to integrate into their methodology; on the other hand, serious algorithmic innovations and expanded computing power for actually constructing plans. It is now easily possible to generate millions of distinct, reasonable plans on a standard laptop in an afternoon, something that would have been unthinkable a few years ago. As access to these data and software has become more widespread, new theoretical developments and applications have changed the way we think about redistricting.

In this chapter we will explore uses of *Markov chains* or random walk methods for generating large collections of districting plans and applications of the resulting ensembles. These techniques have been successfully applied in court cases and legislative reform efforts and are playing an increasingly large role in the design

of both plans and legislation. The underlying mathematical concepts are widely used in many other scientific fields and transferring these techniques to this new setting has led to some great successes in studying redistricting plans.

At a high level, Markov chain Monte Carlo (MCMC) attempts to generate districting plans from a distribution that is "tuned" to satisfy some version of each state's legal criteria, without incorporating explicit partisan biases. The new plans are generated by making iterative changes to a given initial plan while continuing to satisfy the legislative rules. We outline the robust mathematical theory that guarantees that good samples can be constructed, given sufficient time. This gives us an approach to what you might think of as the holy grail for understanding districting plans in context: *baseline ranges* for all kinds of plan metrics that incorporate state rules and voter geography and help us understand the properties of "typical" "reasonable" plans. Back in the 1940s–1960s, when the U.S. courts were trying to figure out how and whether to engage with redistricting, this baseline challenge was laid out by Justice Felix Frankfurter as a prerequisite for thinking clearly about gerrymandering.

Since we're already talking about the holy grail, it's time to introduce our research team's religious mantra:



This is intended both as a reminder and as an exhortation. We *do* have some wisdom about what does and does not work after several years of focused work on this problem, but we do *not* have all the answers about the best ways to apply Markov chain methodologies to our redistricting problems. There are many challenging math problems yet to be tackled—both some relatively low-hanging fruit and some devilishly hard questions—and we hope to provide pointers to researchers looking to enter this area.

Finally, a major goal of this chapter will be to debunk the notion that all computer techniques, or all random map generation techniques, are created equal. In math we like to call a choice "canonical" (echoing religion again) if it is dictated in a standardized and unique way. We'll see that there's very little canon in redistricting and an ineliminable array of modeling choices. We will highlight opportunities for greater community consensus on methods and practices that will promote

more consistent, reliable, and repeatable results. Redistricting is a relatively new application domain for these methods and there are many challenging questions at the boundary of current research in this area.

The remainder of this chapter is organized as follows: First, we provide high-level motivation for the focus on sampling rather than optimization. Next, we describe the basic underlying idea of Markov chains, Monte Carlo, and MCMC. Finally, we connect this methodology to its current state-of-the-art applications to court cases and reform efforts, also highlighting some of the exciting new avenues for future work.

A GLOBAL VIEW OF THE LANDSCAPE OF PLANS

The previous chapter gave a great guided tour from the history to the present day in computational redistricting, from punch-card methods in the 1960s through more modern integer programming or power diagrams. Many of the algorithms developed for redistricting operate by attempting to optimize a particular score or metric, but do not aspire to generate representative samples from the enormous space of possible districting plans. Even the methods that include some *stochasticity* (or random steps) mostly do not provide guarantees about the diversity or distribution of plans that are generated.

The last decade's litigation around partisan gerrymanders has spawned a particular type of counterfactual argument based on the neutrality of random maps. Suppose that a randomized algorithm which is not provided with partisan information, constrained only by (some instantiation of) the traditional districting principles, is shown to never, or at least rarely, generate a map whose partisan measurements are as extreme as those in the challenged plan. We are invited to conclude that the challenged plan is an impermissible partisan gerrymander. In order to justify this argument, we must be persuaded that the sampling methodology is generating *representative* districting plans. To see the perils of mistaking random for representative, imagine that I have a favorite districting plan. I can instruct a computer to select one census unit in the plan that is on the border between districts 1 and 2, and to randomly assign that unit either to district one or district two by a coin flip. I then run this algorithm 100 times and, behold! it gives me back 47 plans with the unit assigned to district 1 and 53 with the unit assigned to district 2. It would be obviously unreasonable to conclude anything at all from this highly specialized collection of 100 plans, even though they have indeed been randomly generated by a computer.

This is where Markov chains enter the scene. The thing that differentiates MCMC methods from other algorithms is the explicit focus on a particular *distribution* over all permissible districting plans. Additionally, the ergodic theorem (Sidebar 17.4) states that if we can generate sufficiently many samples, the distributions of statistics that we are interested in will converge to a stable distribution over the full universe of possibilities. This is what makes it reasonable to describe a given plan as a statistical outlier.

Although the application that motivated much of this research developed in the

adversarial court setting, recent analyses have used the same technique to evaluate and assist reform efforts. Here the question is not, "Was a specific map drawn with improper purpose?" but rather "How would changing the rules change the underlying distribution?" shifting the evaluation from comparing a single map to a distribution to comparing how distributions result from the design of the rules. This evolution has introduced many new research questions that subtly depend on details of the implementations and methodology.

The underlying premise of both of these research directions—outlier analysis and rule design—is that MCMC can be used to discover neutral baselines of arbitrary metrics across the space of districting plans. Even simply comparing these baselines to each other, across elections or states, is already offering new insights into the geospatial structure of American elections and redistricting. It has also guided understanding of the fundamental properties of the metrics that have been proposed in the past as proxies for good redistricting quality.

To begin, we need to address the following questions:

- What is a districting plan?
- How do we know that a districting plan is permissible?
- How do we know that a districting plan is desirable, or even plausible?
- How do we define a distribution that prioritizes plausible or desirable districting plans?
- How can we sample from such a distribution?

We shouldn't expect punchy, universal answers to these questions. Each state has different rules and laws that govern the redistricting process, as well as different political geography that shapes the landscape of possibilities. This chapter will explore how MCMC sets us up for a promising suite of approaches.

2 INTRODUCTION TO MCMC

Let's dive in with a friendly introduction to the ideas and background of Markov chain sampling on discrete state spaces. Applications to political districting have created a renewed interest in these methods among mathematicians, political scientists, geographers, computer scientists, and legal scholars (among others) and this introduction is aimed at presenting the underlying mathematical material in an intuitive fashion for all of those audiences. The goal is to present the key ideas without the need for a significant amount of mathematical background or formalism. Math-ier information will mostly be in sidebars. For additional tools exploring these ideas see the GitHub repository associated with this book (`https://github.com/political-geometry/`).

17.1 EXPECTATION AND SAMPLING

A *probability distribution* is a function that assigns a probability or likelihood to various events. A *random variable* is a variable whose value is determined as the result of a draw from the distribution. We'll focus on the case that there are finitely many possible outcomes, so that the sum of their probabilities is one. We'll call each outcome a *state* and the universe of possible outcomes the *state space*.

An example is rolling a fair die: the state space is $\{1,2,3,4,5,6\}$ and each value has a $1/6$ chance of being on top when the die stops moving. This is an example of a *uniform distribution*, where each outcome has exactly the same probability of occurring. An example of a non-uniform distribution is picking a random letter out of the previous sentence, which gives E and I the highest weight and gives J,Q,Y,Z no weight at all.

The *expected value* of a random variable is a weighted average of the values of the state space: we multiply each value by its probability and add them up. (Notice that the uniform distribution just gives back the usual average.) So for the fair die roll, we get the expectation

$$\mathbb{E}(X) = \frac{1}{6}\cdot 1 + \frac{1}{6}\cdot 2 + \frac{1}{6}\cdot 3 + \frac{1}{6}\cdot 4 + \frac{1}{6}\cdot 5 + \frac{1}{6}\cdot 6 = \frac{21}{6} = 3.5$$

Notice that although we will never actually roll a 3.5 on a six-sided die, it does represent a type of average value: if we rolled the die many times and recorded a large *sample* of random outcomes, the sample average would converge to 3.5.

The same calculation approach applies when the probabilities are not equal, which changes the weights on the values. For example, if we have a loaded die that is weighted so that 3 comes up $3/10$ of the time, 4 and 5 come up $1/10$ of the time each, and 6 comes up half of the time, then the long-term expectation would be $.3(3) + .1(4) + .1(5) + .5(6) = 4.8$.

One of the fundamental results of all of mathematics is the **Central Limit Theorem**. Suppose a random variable is drawn from a probability distribution with true expectation μ and variance var. It does not matter what the shape of that distribution is! If the variable is sampled independently from that distribution, and we let μ_n be the average value of n observations, then the distribution of μ_n converges to a normal with mean μ and variance var/n. This means that if you can only study a random variable by sampling in a "black box" fashion, the experimental evidence helps you to estimate the true expectation. The larger your experiment, the less variance in your estimate.

2.1 MONTE CARLO METHODS

Monte Carlo methods study the aggregate properties of random samples. The origin story for formal Monte Carlo analysis is a deterministic solitaire game played by mathematician Stanislav Ulam while he was sick in bed [1]. ("Deterministic" games are those with no choices to make, such as the game of War—the winner is just determined by the shuffle.) Ulam wanted to figure out how often a randomly shuffled deck would lead to a win. The exact calculation was out of reach, since there are $52! = 80,658,175,170,943,878,571,660,636,856,403,766,975,289,505,$-

440,883,277,824,000,000,000,000 possible shuffles. But once you've analyzed what makes a winning shuffle, you can have a computer repeatedly carry out a sequence of games and see how often you win. This is exactly the type of task that computers are excellent for, since they execute instructions exactly and do not complain of boredom (or of repetitive strain injury).

The same general outline that we applied here is common to most examples of Monte Carlo methods. In sketch:

1. Draw an (independent) sample from the set of all possibilities;
2. Extract some data for each sample;
3. Repeat many times;
4. Average/aggregate the derived data.

Following this procedure offers a way to generate *approximate solutions to difficult problems* by aggregating a large number of *random solutions to easier problems*.

Like many other elite mathematicians in the 1940s, Ulam was working for the war effort, in his case the Manhattan Project in Los Alamos. Ulam's idea was quickly adopted by others in the project, notably John von Neumann, for modeling the behavior of particles released by subatomic processes. Enormous strides in computing power after the war allowed researchers to run a much larger number of trials than would have been possible by hand and provided access to efficient pseudo-random number generation. In the intervening decades, these methods have been applied to problems in physics, chemistry, and computer science as well as in purely mathematical settings.

2.2 DEFINING A MARKOV CHAIN

A Markov chain is a process that moves from state to state in a randomized way in a state space. Its defining property is that the probability of moving to each state at a certain time is determined by your current position. One example is the children's game Snakes and Ladders, where the probability of landing on a particular square on your turn is completely determined by your current square. This kind of process is also the secret sauce in Google's original PageRank algorithm, which works by estimating the *importance* of a website as the probability that a web-surfer would land there after following totally random links for a long time.

Let's build three simple examples to start to understand this. All of them will be random walks on a space with 27 states consisting of the letters of the alphabet plus a space. (For graph representations of these chains, see Figure 2.)

1. **Alphabet Path**: Only allowed moves are from a letter to the ones before or after it in the alphabet, with SPACE after Z. So from A, your next move is definitely B, but from G, you could move to either F or H with equal chances.
2. **Alphabet Cycle**: Same, but now SPACE is also connected to A. Now every state has two "neighbors" and picks one at random.

3. **Keyboard Walk**: starting with any letter, you are equally likely to move to any of its physical neighbors on a standard (QWERTY-style) keyboard. So from H you can transition to any of Y,G,B,N,J, or U with a probability of 1/6 each, while from Q you are equally likely to transition to A or W.

17.2 MONTE CARLO GEOMETRY

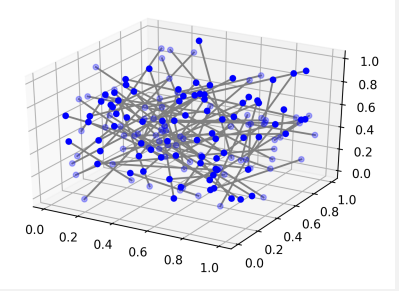
Here is a geometric question that can be tackled with Monte Carlo analysis: What is the expected distance between two points randomly drawn in a unit cube? Although this problem has a mathematical formulation

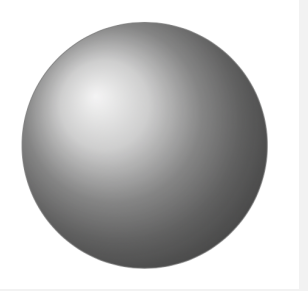
$$\int_0^1\int_0^1\int_0^1\int_0^1\int_0^1\int_0^1 \sqrt{((x_1-y_1)^2+(x_2-y_2)^2+(x_3-y_3)^2}\,dx_1\,dx_2\,dx_3\,dy_1\,dy_2\,dy_3$$

and a mysterious looking exact solution

$$\frac{4+17\sqrt{2}-6\sqrt{3}+21\log(1+\sqrt{2})+42\log(2+\sqrt{3})-7\pi}{105},$$

this is a perfect problem for trying out the Monte Carlo method. If we sample pairs of points with coordinates uniformly random in $[0,1]$, we can report the average. The first run of 1000 trials gave us about .67122, the second run gave .66921 and the third gave .65919. A run of 1,000,000 trials gave .66157. These are not so far off from the theoretical value of .662959... and would continue to improve with longer runs.





Similarly, it may be hard to visualize a ball of radius 1 in five-dimensional space, but it's easy to estimate its volume! I'll just sample n points (x_1,x_2,x_3,x_4,x_5) by randomizing their coordinates in $[0,1]$ and see what proportion of them satisfy $x_1^2+x_2^2+x_3^2+x_4^2+x_5^2\le 1$. This is the part of the ball with positive coordinates. Since the coordinates can have any combination of signs, there are $2^5=32$ similar sections of the ball, so I can multiply the ratio of hits by 32, and voilà! Turns out this volume is about 5.264. It's integration without integrals.[a]

[a] Fun fact! Dimension 5 is the peak volume for the unit ball. The volume decays to zero faster than exponentially in the dimension n, even though the unit n-ball fits snugly in an n-cube whose volume grows exponentially. This seems totally unreasonable until you think about how unlikely it is that $x_1^2+\cdots+x_n^2\le 1$ for large n.

17.3 TEXT GENERATION

An early application of Markov chains was in the analysis of text passages, trying to predict the next letter that would appear in a book written by a given author.[a] Symbols in text are not distributed uniformly: for instance, `q` is almost always followed by `u`, periods are followed by spaces, and the letter `e` is most commonly found at the end of a word. Given a long passage of text, we can compute how often each symbol follows each other symbol and use these proportions to generate new text probabilistically.

This is indeed a Markov chain: the probability for choosing the next letter only depends on the current letter. Let's call this the 1DS chain (one-digit sequences). We can similarly define a 2DS chain that takes into account that `Al` is likely to be followed by `ad`, a 3DS chain that sees that `sec` is frequently followed by `ret` or `ond`, and so on. The longer the strings you consider, the more the output looks like language, at least until you try to figure out what it means. The tradeoff is that the size of the transition matrix grows quickly: if there are n characters in the alphabet, then there are n^2 two-digit sequences, n^3 three-digit sequences, and so on.

Below are some examples generated from letter patterns in the story "Aladdin and the Magic Lamp" from the Arabian Nights.[b] Each of these is a single sample path of the Markov chain induced by the letter sequences. The first line is 50 characters chosen uniformly, for comparison, and 0DS generates letters in proportion to their frequency in the text.

(uniform) ,Kni;;.RgkY:f;;.?ACKKDFtjaBD-vjaIAezAFO-hOzOe?NAm

(0DS) idaleuiupefeibseauitisavisrogeme,aob,aWtosde

(1DS) y mpo fewathe he m, main, wime touliance handddd

(2DS) If ho rembeautil wind was nearsell ith sins. He don the whimsels hed his the my mign for atim, but

(3DS) but powerful not half-circle he great the say woman, and carriage, she sup window." He said feast father; "I am riding that him the laden, while

(4DS) as he cried the palace displeasant stone came to him that would not said: "Where which was very day carry him a rocs egg, and horseback." Then the might fetched a napkin, which were hunting in the

There turns out to be a significant amount of interesting structure in this type of analysis. The transition matrices alone are often enough to distinguish authors from each other, or poetry from prose (see Figure 1).

[a] Similar methods are used for auto–complete functions on smartphones!

[b] All three texts cited here are available from Project Gutenberg (`https://www.gutenberg.org/`).

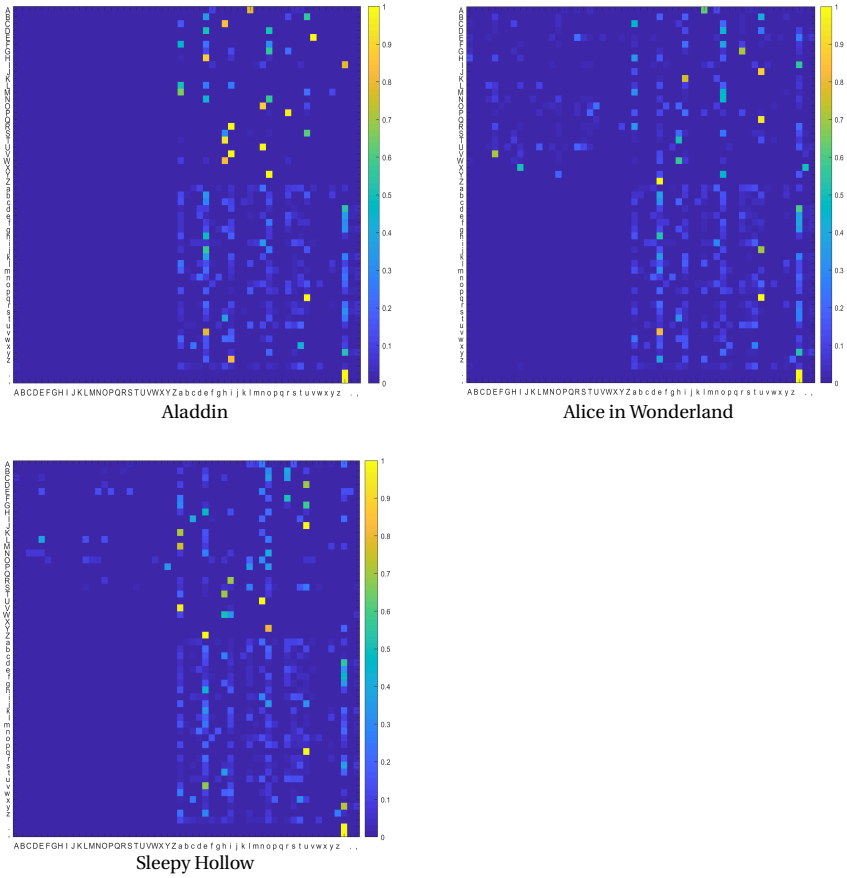


Figure 1: Transition matrices for *Aladdin*, *Alice in Wonderland*, and *Sleepy Hollow*. Each row and column corresponds to a pair of letters, and the brightness of the matrix at that point captures the likelihood of a transition from the first letter to the next in the text.

2.3 APPROACHING A STEADY STATE

We can visualize the random walk on the state space by imagining a person, or an ant, who is crawling from state to state. If the state space is finite, we can use nodes to represent the states and draw edges to represent the possible transitions; this gives us a graph representation of the random walk, as in Figure 2. If the instructions of the random walk amount to, from any node, choosing from among the incident edges with equal probability, then we call it a *simple random walk*—our alphabet path, alphabet cycle, and keyboard walk are all simple in this sense.

The mathematical formalism gets nicer and more unified if we instead consider the evolution of a *probabilistic* position vector. For instance, if the random walk on the keyboard begins at letter Q, then we can record that one step later, its probability vector has $1/2$ weight at W and $1/2$ weight at A.

	initial prob.	step 1	step 2
Q	1	0	$1/4$
W	0	$1/2$	$1/8$
A	0	$1/2$	$1/8$
Z	0	0	$1/8$
S	0	0	$1/4$
E	0	0	$1/8$

In this way, the probabilities keep diffusing through the state space. (At the next step, nonzero probabilities will expand to X, D, and R.)

Thinking about Markov chains in this probabilistic way allows us to study questions about long-term behavior. In general, dynamical systems can have multiple states that are attractors and others that are repellers. But the magic of Markov chains is that there exists a unique attractor—everything is drawn to it. That is, *for a (suitably designed) Markov chain, any initial position will converge to a unique steady state.* This steady state is also called a *stationary distribution.*

If there are finitely many states, say n, then we can formalize this with an $n \times n$ transition matrix M whose (i, j) entry records the probability of transitioning from state i to state j in one step. Let us call its transpose $P = M^\intercal$ the *iteration matrix* of the system. We call it this because it has a nice property: for a position vector v, the matrix product Pv records the probability of being at each position one step later in the process. So if v is your initial position, then $P^N v$ is a complete description of your position at time N.

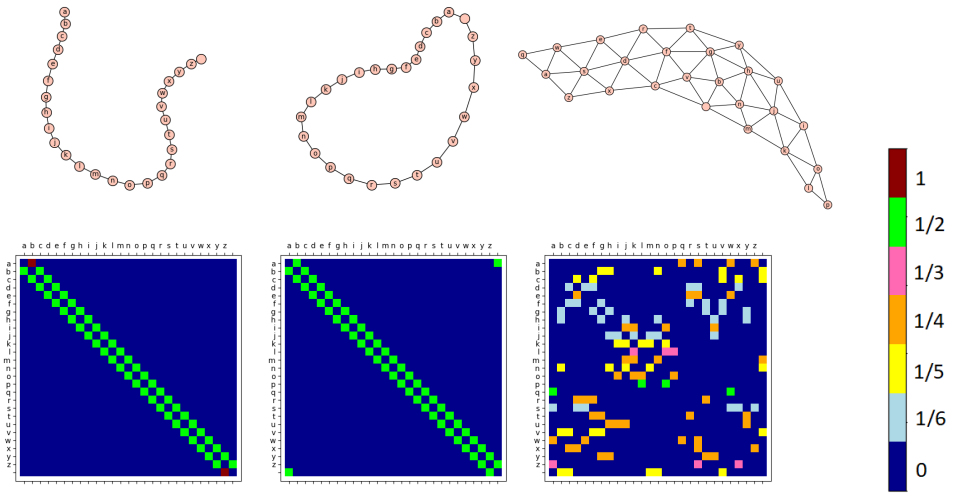


Figure 2: The top row shows the state space as 27 nodes in a graph, with edges for allowed transitions. The bottom row shows the transition matrices in visual format, allowing you to scan the likelihood of going from a letter to any other latter in one step for each of the chains.

17.4 THE FUNDAMENTAL THEOREM OF MARKOV CHAINS

We need to define a few properties of Markov chains to state the fundamental theorem and some surrounding facts. The first adjective we will consider is *periodic*. The period of a Markov chain is the greatest common divisor of all cycle lengths (paths that start and end at the same state). A chain is said to be *aperiodic* if the period is one. Looking at our example chains, we can see that the keyboard is aperiodic (Q–W–Q has length two and Q–W–A–Q has length three, and these have no nontrivial common divisors), and the alphabet cycle is aperiodic (A–B–A has length two and the full tour around the alphabet has length 27), but the alphabet path is periodic because any path starting and ending at the same letter has even length. A common trick to make a walk aperiodic is to add a small probability of remaining in place. This is picturesquely called a "lazy" random walk.

Next, a Markov chain is called *irreducible* if each state can be reached from any other state in a finite number of steps. All of the examples that we have encountered so far have this property. A link-following random walk on the internet does not have this property because some sites have no outgoing links.[a] Markov chains that are both aperiodic and irreducible are called *ergodic*.

A Markov chain is *reversible* if it satisfies a symmetry condition known as "detailed balance." This condition states that in the steady state, the probability of being at state i and transitioning to state j is equal to the probability of being at state j and transitioning to state i. In mathematical notation, if w represents the steady state vector and P the iteration matrix, this condition reads

$$w_i P_{ij} = w_j P_{ji} \quad \forall i, j.$$

The Aladdin text chain in Sidebar 17.3 is an example of a nonreversible chain, since the probability of transitioning from l to A is zero, while the string Aladdin itself shows that the reverse probability is nonzero. Reversible chains have many nice properties and this symmetry condition means that the steady-state distributions are particularly easy to analyze.

Finally, a quick note about measuring success. To say how close one probability distribution is to another, a natural notion is the *total variation* distance between the two measures. Given two distribution vectors u and w, the total variation distance between them is $d_{\mathrm{TV}}(u, w) = \frac{1}{2}\sum_i |u_i - w_i|$. This is just adding up the differences in weight over each state in the state space, normalized so that the distance between any two measures is always between zero and one.

Fundamental Theorem of Markov Chains:

1. Any ergodic Markov chain has a unique stationary distribution. That is, if the iteration matrix is P, then there exists a unique probability vector w (entries summing to 1) such that $Pw = w$.
2. For any probability vector v, its iterates converge to w. That is, $d_{\mathrm{TV}}(P^N v, w) \to 0$ as $N \to \infty$.
3. Every Markov chain can be represented by a random walk on a graph—possibly

> a directed graph with weights on the edges. For the case of simple random walk on a finite graph, the stationary probability of being at position i is proportional to the *degree* of vertex i. That is, where d_i is the number of edges leading to vertex i and $D = \sum_i d_i$, the steady state vector has coordinates $w_i = d_i/D$.

This explains why the steady-state probability is uniform for the alphabet cycle, while the endpoint vertices have half the long-term weight of the others in the alphabet path walk (Figure 2). Note also that it easily follows from the Fundamental Theorem that all simple random walks on undirected graphs (where from each vertex you choose an incident edge with equal probability) are reversible.

Building on this theory, the Markov chain Central Limit Theorem and its various refinements guarantee that given any real-valued function F on our state space, we can estimate its statistics over the state space as a whole by simply collecting samples from a random walk and averaging the values of F on the states in the sample.

This is the sense in which the Markov chain theory is so well suited to redistricting. For years, it has been a burning question to find the normal range of metrics in nongerrymandered plans. If we can find a Markov chain with a suitable steady state, we can use samples to estimate these baselines.

The amount of time that it takes to be guaranteed that $P^N v$ is within a prescribed (total variation) distance of the steady-state w is called the *mixing time* of the Markov chain. There is very beautiful theory when it comes to mixing times, but it is almost never possible to bound mixing times in scientific applications.

To read more, see Levin et al., Aldous and Fill, and Geyer [2, 3, 4].

[a] To make a walk on a finite state space irreducible, one hack is to add a small probability of teleporting anywhere at each step. PageRank works this way.

We can use the three simple chains we introduced in the previous section (the path, cycle, and keyboard walks). Instead of considering a particular sequence of visits to individual states, we instead use the equation above to compute the exact probabilities of arriving at each of the other states. We find that even though the three chains are defined on the same state space, their steady states are different! In the keyboard walk, some states are weighted three times as high as others in the long term, while in the alphabet cycle all states are equally weighted.

2.4 BASELINES WITH MARKOV CHAINS

We can test out the main theorem by seeing how well the Markov chain approximates a numerical "summary score" of the state space. We'll look at two functions from the state space to the real numbers (also called *functionals*).

- **Ascending:** Score is based on position in alphabet.

$$A \mapsto 1,\ B \mapsto 2, \ldots,\ Z \mapsto 26,\ \text{SPACE} \mapsto 27$$

- **Vowel-weighted:** Assign 1 to each consonant, 100 to each vowel, and 50 to Y.

Table 17.1 below compares the theoretical expected values to the estimates obtained from each Markov chain with increasing sample length. All of these runs do a decent job,[1] but we can observe that some seem to converge faster than others, and the rate can depend on the score we choose!

Walk	Score	Experimental				Exact
		2k steps	10k steps	50k steps	100k steps	
Path	Ascending	15.75 (12.5%)	13.99 (0.07%)	14.04 (0.3%)	14.07 (0.5%)	14
	Vowel-Weighted	18.69 (6.6%)	19.70 (1.6%)	19.29 (3.6%)	20.03 (0.07%)	20.02
Cycle	Ascending	14.5 (3.5%)	14.32 (2.3%)	14.1 (0.7%)	13.88 (0.8%)	14
	Vowel-Weighted	21.36 (1.0%)	21.76 (2.9%)	20.97 (0.8%)	21.22 (0.4%)	21.15
Keyboard	Ascending	13.12 (1.2%)	13.34 (0.4%)	13.32 (0.2%)	13.30 (0.05%)	13.292
	Vowel-Weighted	21.02 (9.9%)	19.36 (1.2%)	19.53 (2.1%)	18.91 (1.2%)	19.13

Table 17.1: Experimental comparison for estimating scores. These are independent single runs beginning at the letter A and collecting every score visited by the random walker. Percent error in parentheses.

It is very rare to have rigorous control of the run design needed to get an estimator with provable guarantees. Instead, scientific applications typically use heuristic methods to determine whether or not an estimation has converged.[2]

This example also motivates the use of a common strategy called burning and subsampling. Frequently, practitioners will set a parameter b called "burn-in time" and a second value s called a "sub-sampling parameter." Then, instead of collecting every observed state, a sampling ensemble will be created by collecting states visited at time $b, b+s, b+2s$, and so on. If s is roughly the mixing time of the chain, then these samples will be approximately independent draws from the stationary distribution.[3] For chains where neighbors tend to have similar scores, like the ascending chain, this helps to counter the *auto-correlation*, or degree of similarity from one step to the next, which makes a full sample change its average value more slowly.

We also see a hugely important fact illustrated here that is worth emphasizing: the ground truth itself—what is the average value of the score over the whole universe of possibilities?—depends not only on the state space but also on the probability distribution, because it is a *weighted average*. So, if we're working with the stationary distribution for the keyboard walk, D is weighted three times as much as Q. This will be important below when we turn to redistricting.

[1] If you are following the details, the path walk is periodic. Probabilities proportional to degree is one stationary distribution. Exercise: find them all!

[2] A primary example, sometimes called the *multi-start heuristic*, is to start the chain from different initial states and check that the sample distributions agree. Of course, this can't ensure that you're getting the right answer, but if your multi-start experiment fails, then you can be sure that you're not running long enough.

[3] The use of burn-in in particular is somewhat controversial; see for instance Geyer [4].

2.5 TARGETING A DISTRIBUTION

So far, our examples begin with a process governed by some transition probabilities, then run until they approach stationarity. But in most MCMC applications, we start with a specific distribution we are trying to sample from, then create an irreducible, aperiodic Markov chain designed to target it. The same property that made problems tractable for Monte Carlo analysis—the relative ease of evaluating the properties of a sample rather than the whole—also turns out to be useful for drawing from a target distribution. We can design an appropriate Markov chain knowing only local comparisons for the target.

This was the key idea that was exploited by Metropolis and coauthors in 1953 [5]. As with early Monte Carlo techniques, the original application was to statistical mechanics of atomic particles. This idea was further developed by Hastings [6] and others and has come to be one of the most fundamental computational tools in all of computer science and statistics. In 2000, the IEEE described Metropolis-style MCMC sampling as one of the top 10 most important algorithms of the twentieth century [7].

One situation that calls for this kind of maneuver is when we have a score that makes us regard some states as "better" than others, and we want to prescribe a distribution that prioritizes or up-weights the higher-scoring states. This turns out to be a common situation in physics and Bayesian statistics. We essentially use the score to start with one Markov chain, then design a cleverly weighted coin and use a coin flip to accept or reject each proposed move. When the weighting is just so, it pulls the first Markov chain away from its own steady state and toward the desired distribution.

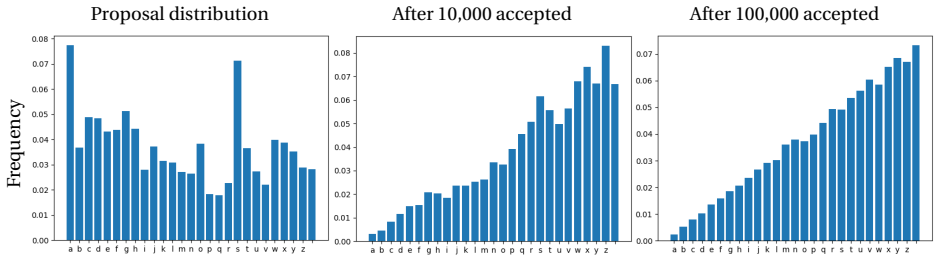


Figure 3: In this run, the proposals are generated according to the keyboard walk, but re-weighted in the Metropolis style to target the ascending distribution. The Metropolis rule is successfully pulling the distribution away from its stationary tendency (left) and toward the ascending shape (right).

This is well illustrated by Figure 3. On its own, the keyboard walk would approach the distribution on the left, but instead it is converging toward the ascending shape. In order to achieve this, many proposed transitions away from the high-scoring letters are rejected, while lower-scoring letters are more readily left behind. Our fidelity to the target distribution improves with longer runs.

17.5 METROPOLIS–HASTINGS

Begin with a score function s on our state space Ω, so that $s : \Omega \to \mathbb{R}$. For instance, the ascending score function has $s(A) = 1$, $s(B) = 2$, etc. We want to sample from the distribution where the states are weighted in proportion to their scores. For example, if we target the ascending scores, then the letter J (score 10) should be twice as likely as E (score 5). For any state $y \in \Omega$, we should therefore assign it probability $\mathbb{P}(y) = \frac{s(y)}{\sum_{x \in \Omega} s(x)}$. When Ω is too large to construct entirely, we won't be able to compute this denominator. However, notice that we can compute *ratios* of probabilities, since the denominators cancel:

$$\frac{\mathbb{P}(z)}{\mathbb{P}(y)} = \frac{s(z)}{s(y)}.$$

And that's good enough for the re-weighting we need.

We perform the Metropolis–Hastings procedure by beginning with a Markov chain to propose steps, and then using the score function to decide whether to accept them. We use the ratio of the new score to the old score to decide. It is this possibility of remaining in place that transforms the stationary distribution to our desired values.

More formally, we follow this sequence of steps:

1. From an initial state y, generate a proposed state z according to the Markov chain with transition matrix M;
2. Accept z with probability $\alpha = \min\left(1, \frac{s(z)}{s(y)} \frac{M_{zy}}{M_{yz}}\right)$;
3. The next state is z if it was accepted and remains y if not. Repeat.

As you can see, a proposed move to z is likely to be accepted if $s(z) > s(y)$, and unlikely if $s(z)$ is significantly lower. This new Markov chain—which has all the possible transitions of the M chain but re-weighted—is ergodic and reversible with steady-state distribution proportional to s.

2.6 TEMPERATURE VARIATION: EXPLORE AND EXPLOIT

In physics, we often have systems that can be modeled with simple (but very large) state spaces, where we want to explore high-energy and low-energy configurations. For instance we can try to understand magnetic systems, or the chemical structure of glass. Randomized models like the one we describe here have been so successful that they've birthed a whole field, called *statistical physics*.

We'll use the classic *Ising model* to illustrate, with math details set aside to the sidebar. We'll describe system configurations as states σ (appearing as red/blue patterns in our pictures), then define a score $s(\sigma)$ called an "energy" which distinguishes between chaotic and clustered states.

Let's imagine that we want a random sample of clustered states—those where red

cells are likely to have other red cells as their neighbors and blue cells are likely to have other blues. And suppose we've set up the energy-based score $s(\sigma)$ to reward this with a higher score for clustered states. How do we sample? For starters, we can use a weighted Metropolis run as described in the last section to try to up-weight states based on s. But this won't work well off the shelf. The Markov chain can get stuck in meta-stable configurations (local optima) that are well-separated from other configurations with similar energy scores, or may even reach global optima that are difficult to escape, which makes it hard to see the diversity of clustered configurations.

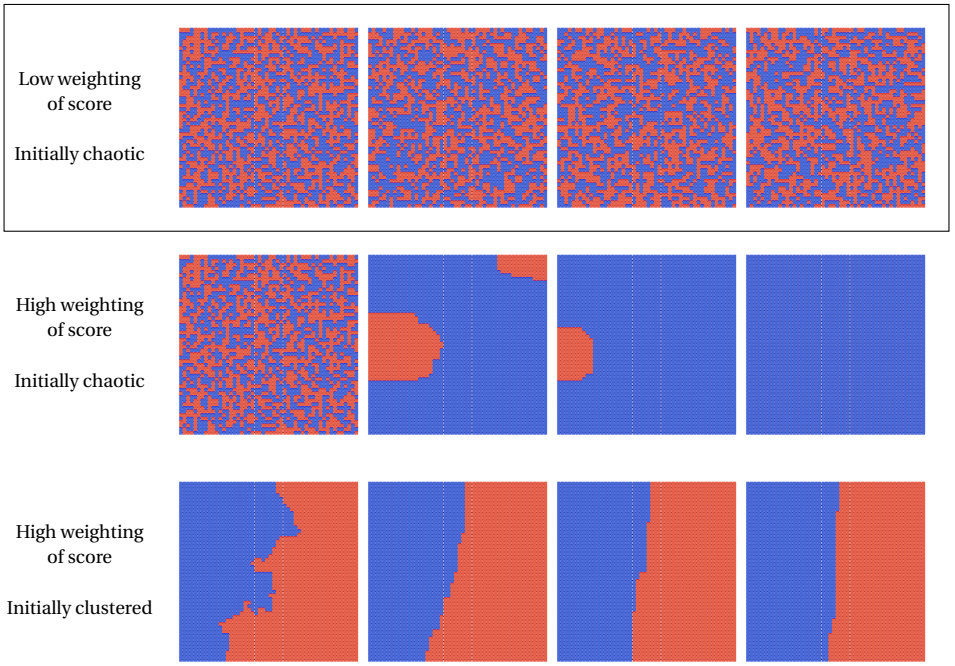


Figure 4: Exploring clustered configurations in a grid with three runs, each at fixed temperature (which controls how the clustering score is weighted during the run). Snapshots are 250,000 steps apart, reading from left to right. These are not efficient strategies for producing a diversity of clustered states.

We see some of the problems in Figure 4. If we run without weighting by the score (first run), clustered configurations are so unlikely that we would not expect to find them at random. But if we run with a high preference for clustering (second and third runs), we will have trouble finding diversity. Notice in particular that the all-red and the all-blue state are both globally optimal for clustering (all neighbors have matching colors), but it would take a truly enormous number of unlikely steps to travel from one to the other while penalizing neighbor differences at every proposed shift. So the second run is stuck at a global optimum, while the third one is stuck in a meta-stable local optimum.

17.6 ISING MODEL: SPECS

The Ising model of ferromagnetism is a mathematical abstraction of a physical system: a network of 'sites,' each of which can be in one of two 'spin states,' usually represented with labels in $\{\pm 1\}$, corresponding to the red and blue colors in the figures here. This is one of the most commonly studied models in statistical physics and also one of the big success stories in the field of MCMC sampling.

Most commonly, the sites are arranged in an $n \times n$ grid; we will denote the assignment of a sign to each node by σ so that σ_i is the spin of node i. Then the 2^{n^2} possible configurations σ make up the states in the state space we will study. Viewing the spin states as magnetic poles that interact with their neighboring sites, we can define an expression called a *Hamiltonian* that represents the total energy of the configuration of spins. In a simple setup this might be written

$$H(\sigma) = -J \sum_{i \sim j} \sigma_i \sigma_j,$$

where $i \sim j$ means that the nodes are adjacent in the grid and J is a term for the interaction strength, here assumed to be constant over the grid. This is designed to distinguish between various kinds of spatial arrangement: if the σ_i are random, the sum will have many positive and many negative terms and will often cancel down to near zero. If the $+1$ and the -1 nodes are highly clustered, most terms in the sum will be $+1$; if they are in a checkerboard pattern the terms will be -1.

The goal is then to sample from a probability distribution over the states given by setting $\mathbb{P}_\beta(\sigma)$ proportional to $e^{-\beta H(\sigma)}$, where $\beta \geq 0$ is a parameter called the inverse temperature. For high values of β (low temperature), this will put a lot of weight on the configurations σ with a large negative $H(\sigma)$, which corresponds to clustering when $J > 0$; on the other hand, for β near zero (high temperature), the probability will be near-uniform.The sum $Z(\beta) = \sum_\sigma e^{-\beta H(\sigma)}$ is sometimes called the *partition function*, which then allows us to write $\mathbb{P}_\beta(\sigma) = \frac{e^{-\beta H(\sigma)}}{Z(\beta)}$.

In his Ph.D. thesis, Ernst Ising solved the one-dimensional model (i.e., on a path graph) exactly, showing that correlations between spins decay exponentially with the distance between the sites. In higher dimensions, we get a more interesting model, finding a sharp *phase transition* as the inverse temperature value β varies. That is, at a critical value of β, we observe a sudden shift between complex, disorganized states for small β and structured, clustered states for large β. Exact solutions are not known for these cases but this is a perfect setting for exploring with MCMC.

Directly sampling from $\mathbb{P}_\beta$ is challenging for high β, despite the fact that it is easy at $\beta = 0$ by assigning the spin at each site uniformly. Instead, we will start with an arbitrary assignment and use MCMC to construct samples from the desired distribution. To move between states, we define transitions where at each step we propose to flip the assignment of a single, randomly chosen node to the opposite sign—this is called *Glauber dynamics*. Following the Metropolis–Hastings procedure with fixed β, we would accept a proposed transition from σ to τ with probability equal to $e^{-\beta(H(\tau)-H(\sigma))}$. The examples in this section illustrate that we get superior results with temperature variation than by running either hot (here, at inverse temperature $\beta = .1$) or cool ($\beta = 3$) alone.

A technique called *simulated annealing* was developed to deal with this sort of phenomenon—its name is motivated by the physical process of heating and re-cooling metal to change its structure. Following this analogy, we will introduce a *temperature* parameter such that steps are more wild and random at high temperature, then settle into aggressive score optimization at low temperature. In the *cooling* regime, we pay more attention to the relative differences between plans, demanding that almost every accepted state must have a higher score than the previous one. Sometimes this distinction in behavior is referred to as "explore/exploit" since running hot lets us *explore* the state space more freely, while cooling then forces us to stay in a smaller neighborhood of the best thing we've found recently, thereby exploiting the high score locally.

The temperature variation over time, also called the *annealing schedule*, is set before the run, prescribing cycles of heating and cooling. Let's see an example of annealing in action, again with the goal of viewing a diversity of clustered states.

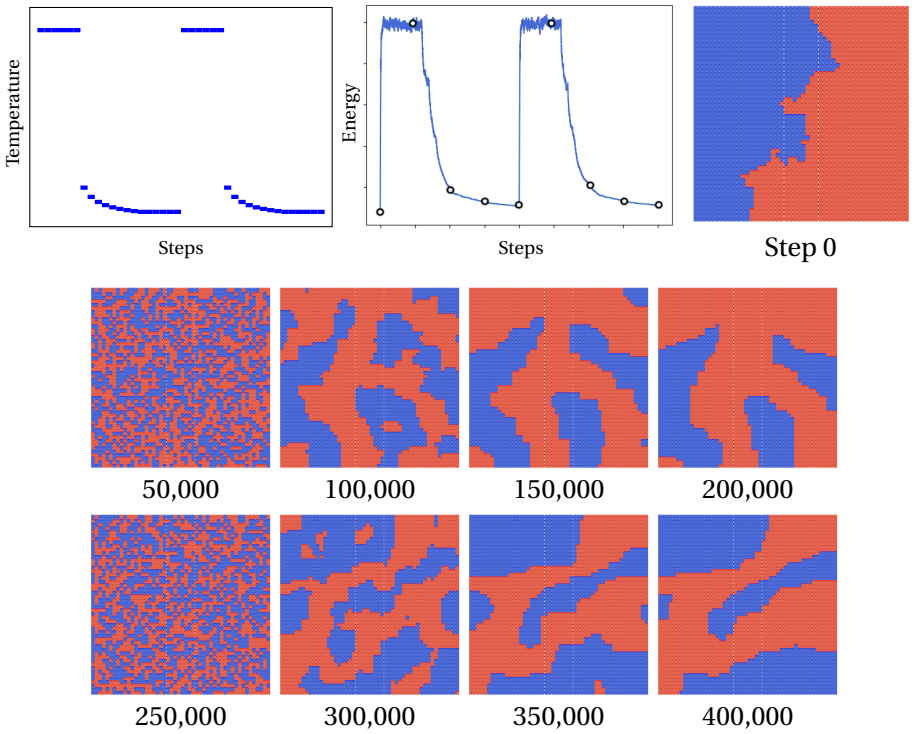


Figure 5: The annealing schedule is shown at the top, with timestamps marked every 50,000 steps and corresponding snapshots below. Each cycle of heating and cooling lets us reset with uniform sampling, then settle into a different clustered configuration by increasingly weighting the score as the temperature drops.

Figure 5 shows annealing performing exactly as advertised. Note that we were able to move between qualitatively distinct configurations in a relatively small number of steps compared to the fixed-temperature runs above.

However, as with many of the methods discussed in this chapter, setting an annealing schedule requires choices and does not come with a canonical strategy. We will return to this idea in the redistricting application below.

3 MCMC FOR REDISTRICTING

We now turn our attention to the application of MCMC methods for political redistricting. First, we describe a formalization of the district-drawing problem in the language of *graph theory*, and then discuss possible formalizations of rules and laws around redistricting. We will introduce several styles of MCMC sampling and survey the state of the art.

3.1 GRAPH PARTITIONS

Although many people think of gerrymandering in terms of lines or curvy boundaries drawn on a map, political redistricting is naturally modeled as a discrete problem where a collection of separate units, like census blocks or voting precincts, are partitioned into districts. This fits the real-world problem, as the Census Bureau reports population and demographic data at the level of census blocks and most states report election results aggregated at the level of precincts. In virtually all cases, you can regard a plan as being built out of census blocks, in the sense that it does not split them.[4] And in many states (like Massachusetts, Louisiana, or Minnesota), state or local redistricting plans are built out of whole precincts. This viewpoint allows us to study redistricting as a discrete problem, using the MCMC tools introduced above.

The object to partition will be a *dual graph* of the chosen units covering the state, which represents each individual unit with a node and places an edge between two nodes if they are adjacent.[5] For instance, Figure 6 shows the counties of Arkansas and the corresponding dual graph. There are 75 counties in Arkansas; on the other hand, there are 186,211 census blocks. In reality, Arkansas's four Congressional districts are built of these smaller pieces. This gives a sense of the gigantic scale of the computational problem and why sampling-based procedures have become so important in this area.

As we've seen, the first step toward defining a Markov chain is to identify the state space. Once we've fixed a dual graph, we'll let the states in our state space be districting plans, i.e., partitions of the dual graph. By a partition, we mean a division of the vertices into groups, which in this case are the districts of the plan. So the partition of Arkansas counties in the last picture of Figure 6 is one state in a space consisting of many trillions of possible plans. Our random walks will wander from one plan to the next. In the next section, we will discuss how to decide whether a partition constitutes a *valid* plan.

[4] In fact, the official description of the districts available from the Census Bureau is given by a *block assignment file*, a table mapping the individual census blocks to their assigned districts.

[5] You have to make a decision about whether to include corner adjacencies. And when you're doing this on real data, you also have to make decisions about what counts as being adjacent across water—for instance, what are islands next to?

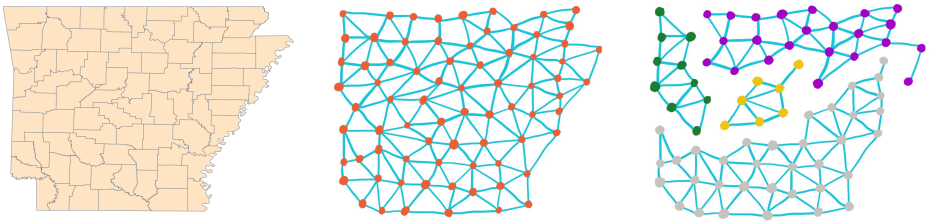


Figure 6: The 75 counties of Arkansas (left), the corresponding dual graph (center), and a districting plan (right).

3.2 DEFINING VALID PLANS

Clearly, many graph partitions do not correspond to reasonable districting plans. Unfortunately, there are far more poorly behaved partitions than reasonable ones. In order to address this issue, we need to enforce some constraints that cut down our state space. The purpose of this section is not to describe the perfect state space for all redistricting problems—no such thing exists!—but rather to highlight the decisions that must be made so that the Markov chain samples are producing reasonable plans for comparisons.

We begin by discussing some commonly enforced *traditional districting principles*. (See Sidebar 0.2.) We'll touch on contiguity, population balance, compactness, county splitting, communities of interest (COI), and the Voting Rights Act (VRA).

To get an algorithm to take an idea into account, you must *operationalize* it, or render it in a formulation that can be handled by a computer. This is one of the steps that is easy to take for granted when making a mathematical model, but the devil is often in these details. Let's just give some examples of operationalizing the rules, which we can illustrate on our Arkansas plan from above.

- Contiguity: the pieces of the partition (the districts) are connected subgraphs.
- Population balance: for some threshold ϵ, each district has a total population between $(1-\epsilon)$ and $(1+\epsilon)$ times the ideal size. (For instance if we set $\epsilon = .05$, then the top to bottom deviation is no more than 10% of ideal.)
- Compactness: the number of edges that were cut to break up the graph into pieces is no more than a threshold.
- County splitting: no more than a threshold number of counties is split between multiple districts.

For most of these, our Arkansas plan from Figure 6 sails past the validity check: the districts are connected, the number of cut edges is just 44 (which is pretty good for this particular dual graph), and the number of split counties is zero (since the building blocks are counties).[6] But the population balance is not great: since it's

[6]Unfortunately, this is not always compatible with prioritizing the preservation of city boundaries, because plenty of cities, including in Arkansas, belong to more than one county.

made out of relatively large pieces (counties), each district is within 10% of ideal but not close to usual Congressional standards of balance.

The COI and VRA criteria are quite a bit harder to handle. One main obstacle to operationalizing COI is that almost no states have a concrete process for official recognition of qualifying communities. If you had those, with shapefiles that tell you their boundaries, then you could handle them with splitting rules like for counties or cities. But another fundamental obstacle is that it's not even clear if most places would prefer to handle COI quantitatively or qualitatively in the first place. (See Chapter 12.)

As for the VRA, the law around its invocation is so complex that it's fairly daunting to incorporate into a mathematical model. In particular, it is a widespread misunderstanding that the VRA requires a certain number of majority-minority districts; instead, it calls for the creation of districts in which minority communities have an opportunity to elect candidates of choice. This can often be roughly gauged by the share of population that belongs to a minority group, but this is not enough to ensure compliance. Nonetheless, several approaches are possible. One thing to keep in mind is that Markov chains will collect plans whose principal intended use is for *comparison*, not for enactment. If you have a quantitative approach to estimating whether a district will be effective, then a reasonable strategy would be to use the number of effective districts as a VRA validity proxy. We'll discuss an approach to gauging effective districts below in Section 3.4.

3.3 FLIP CHAINS FOR REDISTRICTING

In our introduction to Markov chains using the letters of the alphabet, we specified the chain by defining the transition probabilities between each pair of letters. Unfortunately, there are far too many partitions of a state-sized dual graph for us to attempt to compute or store all of the necessary probabilities. Instead, we can specify a Markov chain by describing a set of *elementary moves* that we can apply to a given state in order to generate proposed neighbors.

One theme we will encounter is that even when it is easy to describe a move, it is costly to computations that depend on all the neighbors and the neighbors' neighbors. For instance, let's do a very natural move: start with a plan (say the four-district Arkansas plan in Figure 6), pick a random node, and try to reassign that node a random color. For each of the 75 nodes, there are three new colors to try, so that's $75 \cdot 3$ possibilities. Most of the proposed changes will break contiguity. In an 18-district Congressional plan for Pennsylvania, built out of precincts, there are $9000 \cdot 17$ naive neighbors. What we'll see is that it's easier to try a move and then check validity rather than pre-computing the valid neighbors from each position and choosing among them. Trying and sometimes failing is called *rejection sampling*, and it can be quite efficient as long as the check is quick and the rejection rate is not too high.

DEFINING A FLIP

The very most natural thing to do, especially considering the great successes in the Ising model, is to flip a single node at a time. We'll call this a *flip* walk. This type of proposal has some computational advantages, in that it is easy to iteratively update the computations of score functions and to keep track of the set of nodes that can potentially be changed at each step while preserving contiguity. Figure 7 shows an example of this proposal, where one of the nodes on the boundary between two districts changes its assignment.[7]

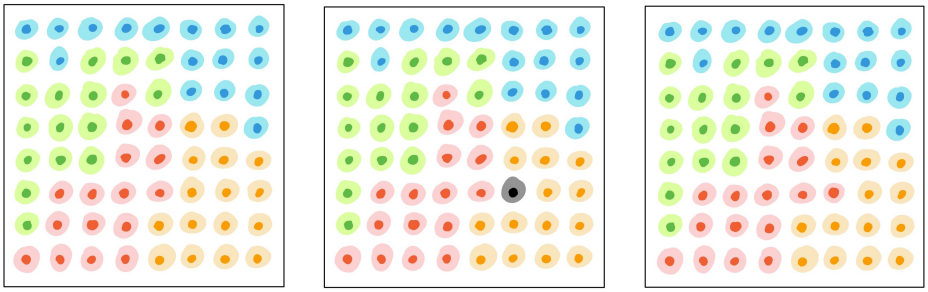


Figure 7: The basic flip step: one unit flips from one district to another.

Even this simple-sounding flip proposal can be implemented in subtly different ways. One option would be to select an edge whose endpoints are in different districts and randomly change the assignment of one endpoint (also at random) to match the other. Alternatively, a boundary node could be chosen and its assignment changed to match one of its neighbors at random. Another version might instead keep track of all of the (node, district) pairs that could be changed to remain contiguous and sample uniformly from that set. Exercise for the reader: confirm that these proposals do not have the same steady-state distribution!

Depending on the formulation of the state space, implementations of the flip proposal can suffer from pathological behavior without careful tuning [8] and in particular have a preference for non-compact plans. Additionally, as only a single node changes assignment at each step, Markov chains using this proposal can require an enormous number of steps to construct approximately uncorrelated samples. In Najt et al. [9], explicit families of graphs were constructed where this proposal exhibits slow mixing. This does not mean that the proposal is wrong for all applications, simply that care must be taken in choosing the sampling methodology and the parameters of the walk to generate useful samples.

TARGETING AND ACCELERATING FLIP CHAINS

Because simply running a flip chain on its own would take astronomically long to converge, and would draw from an undesirable distribution, it seems very natural

[7] One subtlety: if we re-assign a boundary node, we can be sure that the new district that node joins is connected, but it may disconnect its old district by its removal. This makes it slightly trickier to count the neighboring partitions.

to use some of the techniques from the last section to *target* a different distribution of your choice, and to *accelerate* the progress.

At first, it may seem reasonable to target the uniform distribution, equally weighting all plans that pass validity checks. But there are major reasons not to do this. First, this distribution is even more undesirable than one that flip began with, in terms of being overpowered by the least compact plans. Second, the complexity obstructions that suggest very slow convergence for flip chains also apply to the uniform distribution.[8]

Instead, inspired once again by the physics examples, we can target a distribution that is proportional to some score of quality.

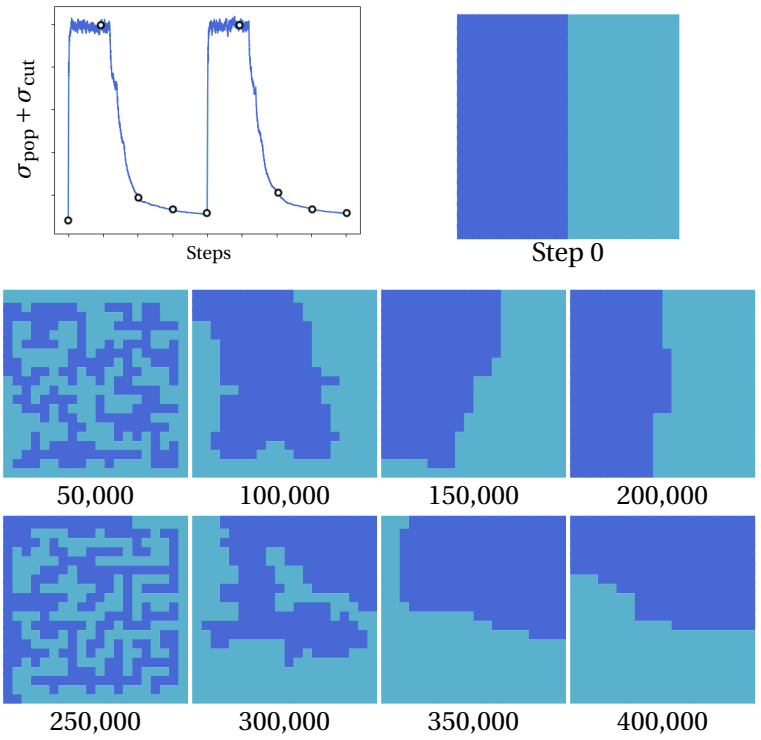


Figure 8: A success for simulated annealing: we sample contiguous two-district plans for a 20×20 grid using a flip walk, with a score $s(P)$ that combines population balance and compactness. The annealing schedule is reflected in the energy trace shown at the top, with timestamps marked every 50,000 steps and corresponding snapshots below. The high-energy states look "fractal," but this level of cooling is successful at recovering short boundaries. Even better, we are sometimes able to traverse from one side of the state space to the other, moving from the vertical split to the horizontal split.

Choosing an appropriate score function for plans, like defining the state space and selecting a proposal distribution, is not a problem with a definitively correct answer. We've already seen that operationalizing the criteria is slippery and subtle.

[8] Some authors, like Fifield et al. [10], first collect a sample with one method and then try to re-weight it after the fact to approximate the uniform distribution. This does not circumvent the complexity obstructions and is likely instead to give poor summary statistics.

To make matters worse, we will now need to combine all of those elements into a single numerical value to serve as the "energy function," summarizing all of the relevant properties of a given plan. This is usually done by a linear combination of several metrics. But choosing the coefficients requires not only deciding on relative importance, but also contending with different units of measurement. How much additional leeway should we permit a plan in population balance in order to make it more compact? A responsible modeler will not only justify these choices, but will also offer a robustness analysis showing that the answers produced by the model are not very sensitive to these decisions.

Let's test out the physics approach in a simple districting application that partitions graphs into $k = 2$ parts. As an example of an energy function we will consider both population balance and compactness. Given a partition $P = (A, B)$ into districts A and B, we set $\sigma_{\text{pop}}(P) = \big||A| - |B|\big|$ to be the difference in the sizes of the districts. (In a grid, the size of a district is just the number of nodes; in a dual graph, it is the population.) Sampling proportional to $e^{-\sigma_{\text{pop}}(P)}$ means that a plan with exactly balanced populations should be drawn with approximately $e^{10} \approx 22{,}026$ times the chance as a plan where one district has 10 more nodes than the other. Next, we'll use the number of cut edges as a compactness proxy. Given a partition $P = (A, B)$ we set $\sigma_{\text{cut}}(P)$ to be the number of cut edges and then can combine the scores into the score function

$$s(P) = e^{-\left(\sigma_{\text{pop}}(P) + \sigma_{\text{cut}}(P)\right)}.$$

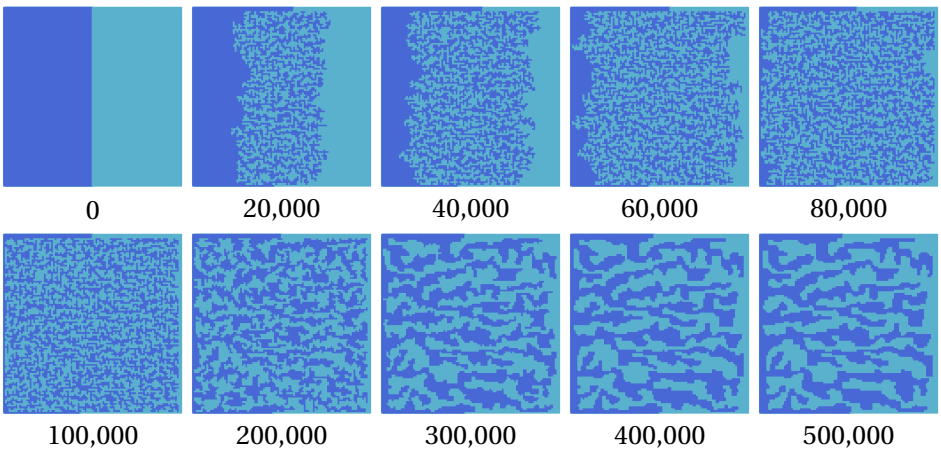


Figure 9: On the 100×100 grid, it is much less obvious how to design a successful annealing cycle. Even after heating and cooling, the boundary assignments have barely changed, and further cooling may not succeed in a reasonable time. A different energy function might be needed.

This example highlights some of the difficulties in making principled decisions about this type of sampling. In a real-world redistricting scenario, designing useful score functions is not a simple task.

Next, having selected a distribution that we would *like* to converge to, we are left with the problem of how to make sure that we actually get there, overcoming any bottlenecks in the state space. So it is natural to try temperature variations to

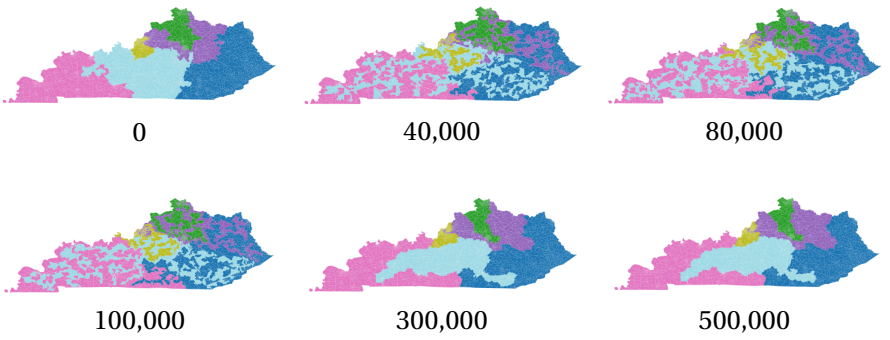


Figure 10: Kentucky's 3,285 block groups tell a similar story to the large grid in Figure 9. This time the cooling has nearly succeeded in returning to a plan as compact as the original, but we can see that annealing has failed to effect a major change. The boundary assignments are still stubbornly persistent.

achieve diversity and accelerate convergence.

Temperature strategies for redistricting flip chains turn out to encounter major problems that were not present in the motivating examples from statistical physics. Like before, the space of plans is huge. But this time, unlike the physics examples, we will need to get lucky enough to select a huge number of changes in a particular order to avoid breaking contiguity—and this is true at every temperature. Second, since the flip procedure itself inclines toward a distribution that is highly noncompact, any compactness preference we implement will be fighting directly against the tendencies of the proposal distribution. Finally, adding cutoff constraints to limit the noncompactness or the population deviation can disconnect the state space entirely: if the limits are too tight, it is easy to construct examples of partitions that cannot be reached from each other using this procedure.

Figures 8, 9, and 10 show annealing runs on a small grid, a larger grid, and the block groups of Kentucky. (For more extensive examples, see DeFord et al. and Najt et al. [8, 9].) Naive annealing was quite successful on the 20×20, but the problems described above began to have real bite once the number of units got to the thousands. One lesson to draw is that it is dangerous to validate techniques on small examples only, since much of the difficulty only kicks in at scale.

CASE STUDY: FLIP WALKS IN NORTH CAROLINA

To see how all these techniques can be combined, let's look at the work of Duke mathematician Jonathan Mattingly and his team, the Quantifying Gerrymandering Group. They participated in federal and state litigation in North Carolina as well as providing model analysis on other states such as Wisconsin.

Both federal and state courts have found their methodology to be persuasive and it was part of the basis of the invalidation by the state Supreme Court of the NC legislative plans. It formed a fundamental part of the evidence before the U.S. Supreme Court in Rucho v. Common Cause (2019). Here we focus on the method-

ology as presented in Herschlag et al. [11], which analyzed Congressional districts and parallels Mattingly's expert report in that case.

The dual graph was constructed from 2692 Census Voting Districts (VTDs), which approximate the precincts in the state. The score function has terms that relate to the North-Carolina-specific districting criteria. In particular, NC has a very strong rule requiring the preservation of counties, and it also has a significant Black population, triggering VRA scrutiny for Congressional districts.

The score terms are:

- $\sigma_{\text{pop}} = \sqrt{\sum_i (p_i - I)^2}/I$, where I is the ideal population of a district and p_i is the population of district i. This is zero if every district is exactly the ideal size.
- $\sigma_{\text{compact}} = \sum_i P_i^2/A_i$, where A_i and P_i are the area and perimeter of district i respectively. This is minimized when the districts are nearly round, which would give an ideal value of $4\pi \approx 12.6$ for any particular district.
- $\sigma_{\text{county}} = f(C_2) + M \cdot f(C_3)$, where C_2 is the set of counties belonging to two districts and C_3 is the set of counties belonging to three or more districts, and f is a function rigged to report 0 if and only if the set is empty. The authors say that M is a large constant but do not report its value.[9]
- $\sigma_{\text{VRA}} = \sqrt{\min(0, 44.48 - B_1)} + \sqrt{\min(0, 36.2 - B_2)}$, where B_1 and B_2 are the highest and second-highest percentages of Black population in any district. This score is zero if and only if $B_1 \geq 44.48$ and $B_2 \geq 36.2$, which are values obtained from an existing map that was approved by a court.[10]

Putting it all together, they attempt to sample proportional to the score function

$$s = e^{-\beta \cdot \left(3000\,\sigma_{\text{pop}} + 2.5\,\sigma_{\text{compact}} + .4\,\sigma_{\text{county}} + 800\,\sigma_{\text{VRA}}\right)}.$$

Here we start to see the dizzying array of choices that go into an analysis like this. Why is the population score weighted 1200 times as heavily as the compactness score and 7500 times as heavily as the county score? Let's continue to describe the setup and hold that question for a discussion of the *robustness* of the findings.

Proposal generation. Select a cut edge uniformly; change the assignment of one of its endpoints to match the other with probability $\frac{1}{2}$.

[9] Suppose county i is in two districts and it has s_i share of its geographical units in the district with the largest share. If it is in three or more districts, let s_i be its share of units in the two districts with the largest share. Then, Mattingly's function is $f(C) = |C| \cdot \sum_C \sqrt{1 - s_i}$. Note that $f(C_2) \leq |C_2|^2 \cdot \sqrt{1/2}$ and $f(C_3) \leq |C_3|^2 \cdot \sqrt{1/3}$, with slowly decaying penalties for cutting off smaller pieces. This is related to an entropy score, but with a number of ad hoc customizations. See Chapter 14 for a discussion of how to use entropy to measure county splitting.

[10] To be clear, this is a massive shortcut to the VRA. There is no basis in law for requiring maps to retain the demographic percentages of an existing map. It may nonetheless be passable for a court if the ensemble is used for comparisons only, and regarded as containing plans that took the VRA into account rather than plans that are certified compliant.

Acceptance probability. Automatically reject discontiguous proposals. Accept contiguous proposals with the Metropolis probability associated with the score s.

Annealing schedule. Initialize temperature parameter β at 0 for the first 40,000 steps, then gradually increase to $\beta = 1$ over the next 60,000 steps. Take an additional 20,000 steps with $\beta = 1$ and then add the final map to the ensemble. This means that a total of 120,000 flips have been proposed between maps in the ensemble.

Winnowing. Remove all plans with population deviation greater than 1%, compactness score of any district worse than 60, any county split four or more ways, or African-American population share falling below $B_1 = 40$ or $B_2 = 33.5$. In their experiment, about one-sixth of the generated plans survived the winnowing step.

Using all of these settings, they generated an ensemble of 24,518 North Carolina districting plans made out of VTDs. Then they compared the distribution of partisan statistics over the ensemble to the statistics observed in the enacted plans from 2012 and 2016, using various recent elections for the voting baseline.

They found that the enacted plans display extreme behavior favoring Republicans, whether measured with partisan bias, efficiency gap, the number of seats won by each party, or a variety of new metrics they devise. (See Chapter 2 for an overview of partisan metrics.) By contrast, there is a plan proposed by a bipartisan panel of retired judges, built to model the work of an independent commission. The judges' plan performs well in line with their ensemble of neutrally generated alternatives. (See Figure 12 for some of the Duke output, together with a replication study.)

To account for the dozens of detailed choices that went into this approach, the authors offer several convergence heuristics and sensitivity analyses to argue that the analysis is robust to the arbitrary choices in its setup. For instance, they tried exchanging their Polsby-Popper compactness score for an alternative dispersion-based compactness score or changing the coefficients in the score function and found that their bottom-line results were qualitatively similar. Any approach with so many choices to make must contend with worries about gameability, so a suite of strong robustness checks of this kind is needed to raise our confidence in the reliability and replicability of this kind of analysis.[11]

3.4 RECOMBINATION

We've seen that flip-based walks can be quite powerful in the redistricting application, but that they are subtle to manage in terms of the centrality of user-chosen specifications. Our research group, the Metric Geometry and Gerrymandering Group (MGGG), based at Tufts and MIT, has spent several years refining a markedly different approach, surveyed in DeFord et al. [8]. It revolves around a fundamental graph theory concept called a *spanning tree*.

[11] The Duke team did not publicly share the code used in this study and report, but a second-generation package is available if you'd like to try your hand at sensitivity analysis. You can find materials in Greg Herschlag's git repo at `https://git.math.duke.edu/gitlab/gjh`.

17.7 "CAREFUL CRAFTING": A LOCAL TEST

A completely different—and very elegant—use of Markov chains has also been developed for redistricting applications, proposed by Chikina–Frieze–Pegden (CFP) [12]. This approach was applied by Pegden (and later by Duchin) in the Pennsylvania Supreme Court litigation over the Congressional map. The theory is developed further in Chikina et al. [13], and earlier work revolving around the same ideas can be found in Besag and Clifford [14].

Suppose your state space is Ω and you fix any functional $f : \Omega \to \mathbb{R}$, any reversible Markov chain, and any value $0 < \epsilon < 1$. Now suppose that a sequence of consecutive states visited by the chain is

$$P_0, P_1, P_2, \ldots, P_N.$$

Then we can consider the set of scores $\{f(P_i)\}$ observed over that sample. The theorem states that the probability that $f(P_0)$ is in the most extreme ϵ fraction of the $\{f(P_i)\}$ is at most $\sqrt{2\epsilon}$. Notice that this result does not require any statement about the convergence of the chain, only that the proposal is reversible. It applies to very short runs as well as long runs, but provides a weaker conclusion. And it also does not require ergodicity—the state space need not be connected, and the theorem applies just as much when the sample is collected from a small connected component, but again with a possibly weaker conclusion.

This suggests a rigorous gerrymandering test, which is very powerful because it does not require a demonstration of ergodicity or convergence. Start a Markov chain at a given plan and let it run for a large number of steps. If the initial state scores worse than the vast bulk of observed variants, then you can be extremely confident that it was not chosen from the stationary distribution!

For the Pennsylvania example described in the CFP paper, the dual graph is made out of the approximately 9000 precincts in Pennsylvania and the Markov chain is a "lazy" flip run designed to have a uniform steady state. In Pegden's report, he didn't use scores for weighting but instead thresholded the various measurable criteria (county splitting, population deviation, compactness) to be in a reasonable range, which amounts to a uniform walk on a restricted state space. Unlike the score-based approach above, the chain is not up-weighting plans that are better aligned with the districting principles. It means all allowable plans are equally likely in the steady state.

This is exactly what is needed to apply the CFP theorem and test. In the paper, the null hypothesis that the enacted plan was drawn from the uniform distribution is rejected with p-values between .0001 and .0000001 depending on the chosen partisan metric, using chains of approximately a trillion steps. And Pegden's expert report found even more eye-popping p-values in litigation. That means you can be very confident that the enacted plan wasn't chosen uniformly at random because its partisan behavior is a great deal more Republican-favoring than the other plans that were found by the chain.

But so what? After all, the Republican legislators never claimed that they were choosing a plan blindly from all the possibilities, and humans are terrible at imitating uniform distributions even when they try. In order for this to have strong persuasive power, you'd want to be sure that this test doesn't merely tell you that a plan was made

by people rather than a computer! In Duchin's use of the CFP test while consulting for the Pennsylvania governor, she included evidence that a plan constructed by the governor's map-making team (without her involvement) *passed* the test—it had partisan properties typical of the observations over long chains. On the other hand, a new compact plan created by members of the legislature failed just as badly as the original it was vying to replace.

Doing this kind of double-check—making sure you have not inadvertently set up a test that only computers can pass—is essential for rolling out this test on a wider scale, as is more study of its gameability.[a] Nonetheless, this way of arguing that a plan has been "carefully crafted" to be much more favorable to the party that controlled the process than a great bulk of similar plans has an unmistakable appeal, and has been found persuasive by the court in Pennsylvania, and later in North Carolina.

[a] This is particularly true since later improvements [13] have massively strengthened the sensitivity of the test by upgrading from a p-value on the order of $\sqrt{\epsilon}$ to one on the order of ϵ, heightening worries about false positives.

A recombination step will typically change the assignment of many nodes at once, which allows for far quicker traversal of the state space. At each step, two districts are merged, forming a graph of twice the desired size. Next, a spanning tree is chosen for that graph. Then, we seek an edge in the tree that we'll call a *balance edge*: when we cut it, the two new pieces that are formed should have equal size. Replacing the two merged districts with these two formed by cutting the tree gives us a new districting plan. This process is cartooned in Figure 11.

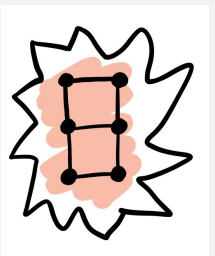
As with the flip walk, recombination admits many variants. For example, there are multiple ways to construct random spanning trees of the subgraph and multiple ways to seek and select a balance edge to cut. We could also merge more than two districts at a time (though at the possible cost of more complexity in the partition step). In the bigger picture, there is no need to use spanning trees at all, as any method for partitioning a merged subgraph could be used to generate the next plan. Nonetheless, we will stick with two-district-at-a-time spanning-tree-based methods for the rest of this exposition, and we will call that Markov chain ReCom.

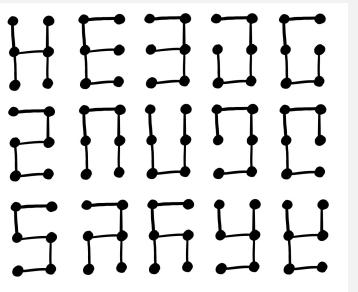
ReCom has three major features that differentiate it from flip chains. One is that there is far less autocorrelation from one plan to the next, which promotes faster convergence and avoids some of the rigidity that made physics-motivated techniques less effective for flip steps. Second, it scales well with the size of the graph being partitioned, because its spanning tree step has polynomial complexity and the number of steps needed to touch all districts is proportional to the number of districts, not the number of units. Finally, it does not need careful weighting to obtain reasonably compact plans.

17.8 SPANNING TREES

As we've seen, a graph is a collection of vertices, together with edges that join some of them pairwise. A *tree* is just a graph with no cycles: there is no edge path that starts and ends at the same vertex without backtracking. And so for any graph, you can create a *spanning tree*—a tree that covers all of the vertices—just by removing edges that appear in cycles until there are none left.

To illustrate this, consider the 3×2 grid graph. It has seven edges, and you can make a spanning tree by removing any two of them while being careful not to disconnect the graph. There are exactly fifteen ways to do this:





For a 3×3 grid graph, there are 192 possible spanning trees and for a 4×4 there are 100,352—the count grows fast! Let's define $\text{sp}(G)$ to be the number of spanning trees of a graph G. There is a beautiful little formula that counts them for you. It is attributable to the physicist Gustav Kirchhoff, as part of his study of electrical circuits in the nineteenth century. First, form the $n \times n$ *graph Laplacian* L by putting the n vertex degrees on the diagonal and subtracting off the adjacency matrix of the graph. This matrix L encodes all sorts of fundamental information about our graph. If G is connected, then $\text{sp}(G)$ is just $1/n$ times the product of the nonzero eigenvalues of L. You can compute this straight from L by eliminating any row and corresponding column to form an $(n-1) \times (n-1)$ minor, then taking its determinant! For our 3×2 example, we get

$$L = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 & 0 \\ -1 & 0 & 3 & -1 & -1 & 0 \\ 0 & 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & 0 & -1 & -1 & 2 \end{pmatrix}; \qquad \text{sp}(G) = \det \begin{pmatrix} 2 & 0 & -1 & 0 & 0 \\ 0 & 3 & -1 & -1 & 0 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & -1 & -1 & 2 \end{pmatrix} = 15.$$

For us, spanning trees will be a crucial device for partitioning because of a key feature: *if you cut any single edge of a tree, you have divided the graph into exactly two parts*.

One last thing to know about spanning trees before we move on: there are remarkably efficient algorithms for generating them randomly! In particular, Wilson's algorithm (based on loop-erased random walk) can be used to get near-uniform sampling of all the spanning trees of G in polynomial time. So when we need to find a spanning tree as a step in a recombination algorithm, we can usually do it fairly fast, even for large graphs.

Now let's consider what the spanning tree count $\mathrm{sp}(G)$ can be said to measure about a graph G. One thing to note is that $\mathrm{sp}(G) = 1$ if and only if G is a tree. This means that a path has only one spanning tree, no matter how long it is. But on the other hand, the number of spanning trees of a grid-graph grows explosively. It's not hard to convince yourself that the spanning tree count is greater when a graph is "plumper," and that it's reduced dramatically by "tentacles" or "necks." From this point of view, it's reasonable to treat $\mathrm{sp}(G)$ as a compactness score for the graph!

To read more about the basics of spanning trees, most introductory texts in computer science or combinatorics cover properties and elementary algorithms, such as Cameron [15]. To read more about what spanning trees might have to do with compactness, check out Duchin and Tenner [16].

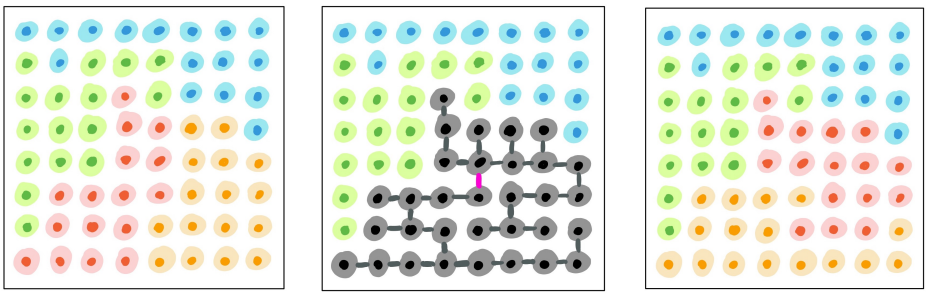


Figure 11: The basic recombination step: two districts are merged, a spanning tree is chosen, a balance edge is selected and cut, leaving two new districts.

Recently, Cannon et al. introduced a reversible variant of ReCom with a prescribed stationary distribution [17]—that is, we know exactly how much some plans are weighted relative to the others. It's even very easy to write down that stationary distribution in closed form. Recall from Sidebar 17.8 that $\mathrm{sp}(G)$ is the number of spanning trees of a graph G, which can be regarded as a kind of compactness score for the graph. Suppose a districting plan P is composed of districts $P_1, \ldots, P_k$. Cannon et al. [17] show that the stationary distribution of reversible ReCom puts a weight on P that is precisely proportional to $\prod_{i=1}^{k} \mathrm{sp}(P_i)$, the product of the spanning tree counts of its districts. That means that plans are naturally weighted by compactness!

Even though "regular" ReCom does not have exactly this stationary distribution, it draws a similar distribution of plans so that it creates compact ensembles without any tuning, and case studies have found extremely fast convergence in summary statistics.

RECOM CASE STUDIES

We will briefly describe two case studies with ReCom chains: a run on Congressional districts in North Carolina and another on state House districts in Virginia. The North Carolina study was focused on the partisan gerrymandering case Rucho v. Common Cause, intended to demonstrate whether different Markov chain meth-

ods might hope to give similar answers. And the Virginia study was directed at analyzing a racial gerrymandering case, Bethune-Hill vs. Virginia State Board of Elections (2019), which revolved around the manipulation of Black population in the state's legislative districts.

Let's start with North Carolina. There, we convened a team of mathematicians and law scholars to write a "friend of the court" brief aimed at helping the Supreme Court to understand the recent Markov chain breakthroughs.[12]

We aimed to see how a partisan-neutral ensemble of compact, contiguous, population-balanced plans would compare to the carefully tuned Duke output. To do this, we used precincts as the building blocks and ran a ReCom chain for 100,000 steps, allowing maximal population deviation of 2% from ideal.[13] That's it! In a few hours on a standard laptop, we get a large and diverse collection of plans.

Figure 12 shows that this very simple run gave outputs that are remarkably consonant with the Duke ensemble. In both, at least 50% of plans have Democrats winning districts indexed 9 through 13, which means 5 seats out of 13, and roughly 25% of plans have Democrats winning 6 seats. By contrast, the plans enacted in 2012 and 2016 had only 3 seats for Democrats in this vote pattern, which is in line with the notoriously brazen assertion by David Lewis, that he had commissioned map locking in a 10–3 Republican advantage only because he couldn't find a way to get an 11th seat.

We highlight this comparison because it is encouraging to see that two very different Markov chain methods give harmonious answers. To see more about North Carolina ensembles and the effects of layering in various districting criteria, visit the GitHub repo for this chapter [18].

Next we turn to Virginia's 100-seat House of Delegates, the subject of the long-running lawsuit Bethune-Hill v. Virginia State Board of Elections. Blocks, rather than precincts, were the natural choice of geographic units to analyze Virginia, because their House plans do not in fact keep precincts whole (and are only allowed 1% population deviation)—just as importantly, vote totals at the precinct level are less salient in a racial gerrymandering case. Thus, the dual graph of Virginia has 285,762 nodes, which is far out of range to expect good performance from a flip chain.

In this study, we used a ReCom ensemble to shed some light on VRA litigation. As we've heard, VRA law is very tricky, because it centers on the creation of *effective* districts for the minority group, in this case Black voters, to elect candidates of choice. To do that requires an estimate of voters' preferences by racial group, which

[12] Fully, it's the *Amicus brief of mathematicians, law professors, and students in support of appellees and affirmance*. As a historical note, we believe it to be the first Mathematicians' Brief (so named) for the Court. There was a Statisticians' Brief in Gonzalez v. Planned Parenthood (2007) arguing that the government had misrepresented p-values in its arguments about late-term abortion. And there was a Computer Scientists' Brief in Lotus v. Borland (1996) weighing in on whether certain elements of computer interfaces were more like languages or functions, for copyright purposes.

[13] For partisan cases, we consider it highly valuable to use precise cast vote totals, which means that precincts are the smallest usable unit. Since those are bigger than blocks, we will typically allow 1–2% population deviation in order to have the Markov chain move efficiently. A professional mapmaker can easily refine such a plan to zero balance.

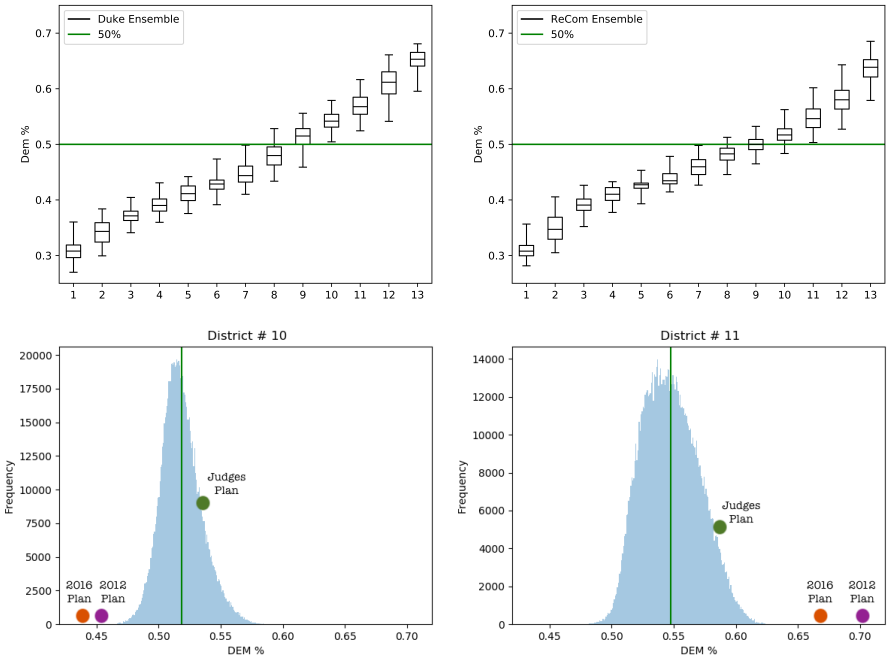


Figure 12: Images from the Mathematicians' Brief. Fixing a vote pattern (Senate 2016), we order the districts in each plan from the smallest (1) to the highest (13) Democratic share of the two-party vote. The boxes show the 25th–75th percentile vote share observed in the ensemble, and the whiskers show 1st–99th percentile. Top row shows the Duke ensemble compared to an untuned ReCom ensemble: not identical, but substantially similar. Bottom row highlights the districts indexed 10 and 11, and shows blatant packing and cracking in the legislatures' plans compared to the ReCom ensemble, or the bipartisan judges' plan. (The ensemble mean is marked with a line in the bottom row.)

is not immediate in a system with a secret ballot. The state of the art for racially polarized voting analysis is a method called "ecological inference" or EI; Bethune-Hill plaintiffs' expert Max Palmer performed EI in all of the house districts of the state, finding that every challenged district was expected to favor a candidate of choice for the Black community in a general election as long as its electorate was at least 45% Black by voting age population (BVAP) [19]. In fact, by Palmer's methods, just 37-38% BVAP would suffice in all but one district. None of the districts requires 55% BVAP, so we treated the range of 38–55% BVAP as a critical one for plausibly effective districts.

How many districts can be simultaneously created that are over 50% BVAP? How about over 37% BVAP? We created an ensemble of alternative plans that are compact, contiguous, and population-balanced to within the 1% deviation limit prescribed in Virginia law. We found that just taking 20,000 accepted ReCom steps and recording every single plan that was encountered gave statistics that were consistent across different starting points or further run lengthening.

We found that hundreds of plans in our race-neutral ensemble had 15 districts in

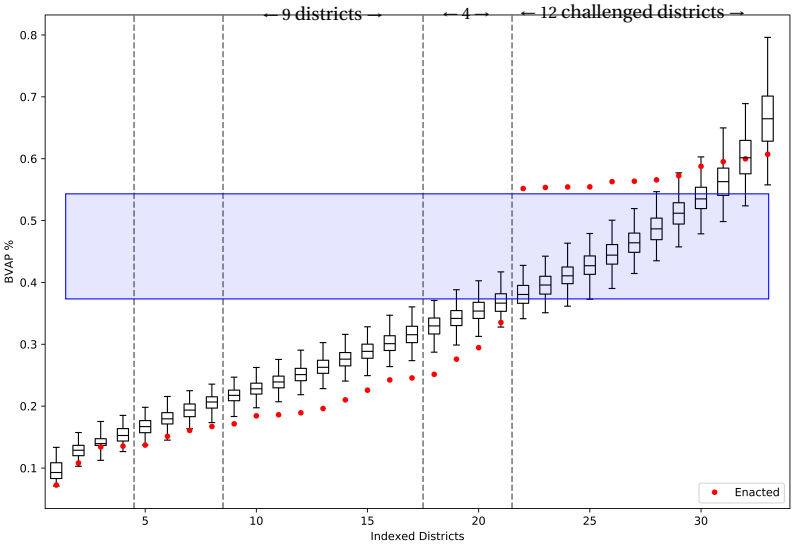
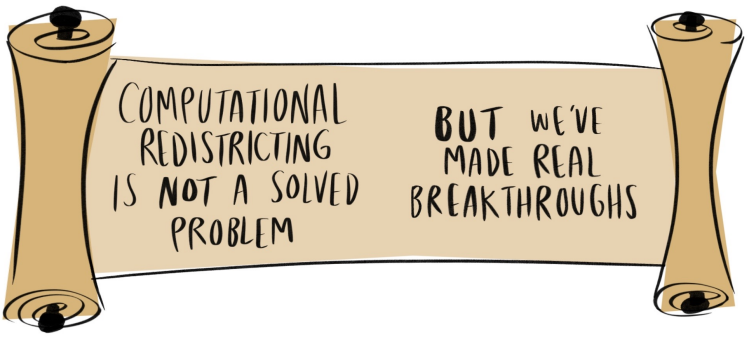


Figure 13: Black voting age percentage (BVAP) by district in an ensemble covering the region of Virginia affected by the Bethune-Hill lawsuit. Red shows the levels in the challenged plan. Boxes show the 25th–75th percentile of the ensemble and whiskers show 1st–99th percentile, as before. We see that packing in the 12 challenged districts has led to cracking in the next 4, and even the following 9.

our plausible range of effectiveness, as opposed to 12 districts in the Republican legislators' plan.... and 13 in the Democratic counter-proposal. So *both* sides are leaving opportunity districts on the table. More than that, the analysis reveals the costs of packing in the plan that was challenged by the lawsuit: the elevation of Black population in the first 12 districts is balanced out by depressed Black population that is not evenly distributed over the other districts, but instead concentrated in the ones with the highest prospects for Black voting power, alone or in coalition with other groups (see Figure 13).

These examples illustrate that ReCom lets you generate a large collection of plans with far less user choice (parameter-tuning, temperature manipulation). It puts Felix Frankfurter's haunting challenge—finding the neutral baseline—within reach of laptop computing.



3.5 SURVEY OF OTHER SAMPLING APPROACHES

There are numerous other district-generation methods out there with various pros and cons. In particular, three political scientists have developed notable sampling methods, which we'll briefly describe.

Jowei Chen (University of Michigan) uses an agglomerative algorithm that is based on iterative merging (see Chapter 16), and has made numerous court appearances based on generated ensembles, typically containing 100–200 maps.[14] His method is intuitive to describe: growing and merging regions amoeba-style until they cover the space with the correct number of districts. These techniques are likely to be useful in the future for finding plans that can be used as starting points in various ways: for initializing a random walk, or as a jumping-off point for the deliberative work of a commission. On the other hand, the method comes with no control or description of the sample distribution, and so provides no grounding for statistical claims. Due to the high rejection rate, agglomerative techniques have difficulty generating diverse ensembles and sometimes have difficulty generating large ensembles at all. Given these various limitations, agglomerative methods are likely to have worryingly high false-positive rates if used for outlier analysis.

Wendy Cho leads a team at the University of Illinois, with supercomputing expert Yan Liu as a major collaborator. Theirs is an evolutionary algorithm that uses a flip step most of the time, then occasionally applies a crossover step based on the common refinement of two partitions. The main upside is that, as discussed in Chapter 16, evolutionary algorithms can do an excellent job of heuristic optimization—that is, they can find "good" plans—and managing multiple populations lets you successfully take advantage of many computing cores in parallel, so the algorithm runs very fast. On the other hand, you lose touch with the theory guaranteeing convergence to a steady state, so it's not clear how the quickly proliferating plans are distributed. Furthermore, the parallelization is carefully engineered for the Blue Waters computing environment (a research supercomputer at University of

[14] Even if drawing from a well-justified probability distribution, this very small sample sizes make unlikely events look impossible. For instance, if something occurs 1% of the time, there is a greater than one in three chance that it will be entirely absent from a collection of 100 maps. ($.99^{100} \approx .366$)

Illinois Urbana-Champaign) and the code is not public, making it quite hard to draw comparisons between this and other methods.

Kosuke Imai is a political scientist and statistician at Harvard. His team, like Duke's, does physics-inspired MCMC with temperature variation (in their case, a technique called parallel tempering). Their proposal flips a few nodes at a time rather than one, providing a modest acceleration. They have made a major and commendable effort to provide benchmarking for all the methods in the redistricting community by developing some (very) small datasets with complete enumeration.[15] Equally commendable: they make their code publicly available!

4 EXPLORING WITH ENSEMBLES

4.1 NOT JUST FOR LITIGATION!

The method of ensembles has many applications for redistricting analysis and reform beyond the adversarial setting of challenging plans in court.

Criteria tradeoffs. The Virginia study described above mainly relied on "vanilla" ensembles, made only with compactness, contiguity, and population balance. In DeFord and Duchin [21], we layer in many other criteria—tighter population balance, preference for keeping cities and counties intact, attention to voting rights share—to see how they interact. We found no basis for some "folk knowledge" that was circulating as Virginia considered a constitutional amendment for redistricting reform, such as the idea that requiring higher compactness would hurt minority representation or that keeping cities intact would favor Republicans. In fact, the preservation of cities and counties had the effect of narrowing observed partisan outcomes, reducing the frequency of maps that had the greatest advantage for either party.

Partisan metrics. Sometimes metrics are designed to measure one thing, but end up being sensitive to factors other than the ones that are advertised. For instance, Chapter 2 shows that the efficiency gap, advertised as a measure of packing and cracking, actually only depends on how many seats are won by each side. You can similarly study other metrics like partisan symmetry scores to see how they behave when tested on real data and many thousands of plausible districting plans [22]. We show that the partisan symmetry standard has many bugs in practice, including systematically reporting advantage for the wrong party in some realistic cases. (We dub this the "Utah Paradox"!)

Nesting. Alaska law requires the 40 House districts in their state legislature to nest 2-to-1 within the 20 Senate districts. (Nine other states had similar nesting rules in the last redistricting cycle.) Suppose you were handed the current House map and required to come up with a pairing of adjacent House districts. We found

[15] Of particular mathematical interest is their approach to approximate enumeration using a data structure called Zero-suppressed Binary Decision Diagram [20].

that there are 108,765 possible matchings in Alaska, which is an imperceptible sliver compared with the usual size of a redistricting problem. Nevertheless, we found that the ability to choose the matching gives almost as much control of the partisan outcome as if one were drawing the map from scratch [23].

Competitiveness. Here, ensemble methods are used to study a range of possible rules for promoting competitiveness in districting. Using two methods—winnowing to the most competitive plans in a neutral ensemble and hill-climbing with flip steps to preferentially create more competitive plans—we show that quantitatively prescriptive language that has been appearing in recent reform measures may be ineffective or generate unintended effects in the future [24].[16]

Least change. What if you have a reason to want a map that makes the least change from a previous one, such as under a rule favoring the preservation of district cores? Mattingly et al. [11] consider this with an experiment using local sampling. Short runs are carried out, rejecting proposals that have more than 40 nodes assigned to different districts than in the initial plan. In Chapter 14 an approach to this, by introducing an entropy-based metric on the space of plans, is also given.

Coalitions and alternative voting systems. Finally, what if you are not sure your community is well served by districts at all? We use single-member districts by law at the Congressional level, but counties and cities have much more latitude to design a system of election. MGGG has carried out tailored studies that look at Asian communities in Santa Clara, CA [26], Latino and Asian communities in Lowell, MA [27], and Black and Latino communities in Chicago, IL [28]. In all three cases, we recommended serious consideration for ranked choice voting in multi-member districts (see Chapter 20 and Chapter 21), finding that it performs at least as well as the districting options found by algorithmic means, but with enhanced opportunities for coalitional representation.

4.2 AN INVITATION

Beyond the Markov chain methods already developed, there is still significant room for creativity in designing elementary moves on graph partitions. New methods should strive for easy implementation, low rejection rate, adaptability to varied districting criteria, and of course theoretical properties like provable ergodicity or reversibility. Although probably extremely difficult, it would also be very valuable to prove mixing time bounds for any of the graph partition methods that are currently in use, where many problems are even open for grid graphs. Lower bounds are useful because they warn of likely failure of certain approaches at large scale; upper bounds would give better statistical guarantees.

With respect to the basic spanning tree ReCom proposal, there are many natural questions about the combinatorics. (What share of trees have a cut that partitions

[16] The districts found by very short and simple hill-climbing runs were similar to extremely competitive plans hand-made for the 538 Atlas of Redistricting project [25].

the nodes equally or near-equally? What kinds of planar graphs have the most spanning trees? and so on.)

Often, algorithms that have poor worst-case performance can still be efficient on nice classes of graphs. Moving beyond grid graphs and other lattices, it would be fundamentally interesting to understand the properties of the graphs realized as dual to Census and precinct geography.[17] This would be useful not only for complexity analysis of algorithms, but also to answer basic intriguing questions like whether different states and places have any artifacts of planning policy visible in the graphs themselves—or even within a state, whether cities are detectable from the abstract dual graph alone.

There would be immediate applications for efficient multi-resolution partitioning methods that start with larger units before refining with smaller sub-units. Prioritizing preservation of counties, cities, or COI could benefit from multi-resolution mapmaking, and a block-level tuning step could provide population balance at the end of a mapmaking process rather than foregrounding it at the beginning.

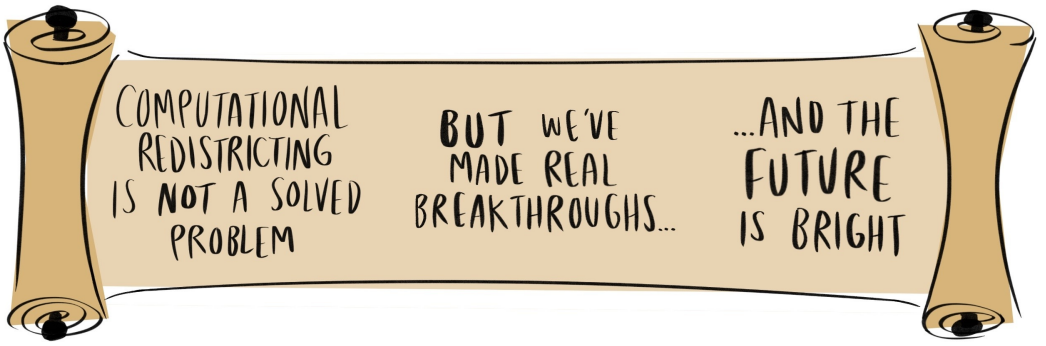
Last but not least, stability and robustness—demonstrating that consonant results are obtained within and across techniques as user choices vary—are paramount concerns as the toolkit continues to expand. As we have seen, there are many setup decisions that must be made in order to design a sampling algorithm for a specific case or state. It is essential to measure the sensitivity of results to model design, tuning, and data perturbation. Determining conditions for reliable and robust findings is perhaps the most important open question in this space.

5 CONCLUSION: STILL NOT A SOLVED PROBLEM

The overarching goal of MCMC methods for redistricting is to generate ensembles of alternative maps that can put a proposed plan in context of the full universe of possibilities. Setting this up requires hard work to certify that we are drawing samples according to a clear rule for weighting some more highly than others. In order to avoid making a test that only computers can pass, we need to know that our ensembles are sufficiently diverse to account for the many ways that benign but unspoken principles can make people's maps different in subtle ways from computer outputs. Ground truth is hard to come by in redistricting, but we should take it where we can, and use it to calibrate our tests.

Redistricting pushes mathematical knowledge to the research frontier, but the scientific consensus is crystallizing around powerful and efficient methods of analysis. Or in other words...

[17] In many cases, the dual graphs are planar and mostly triangulated (as in Figure 6), but when the units are disconnected (as precincts quite often are) the combinatorics can get substantially worse.



REFERENCES

[1] Robert Eckhardt. Stan Ulam, John Von Neumann, and the Monte Carlo Method. *Los Alamos Science*, pages 131–136, 1987.

[2] David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, Providence, R.I, December 2008.

[3] David Aldous and James Allen Fill. Reversible Markov chains and random walks on graphs, 2002. Unfinished monograph, recompiled 2014, available at `http://www.stat.berkeley.edu/~aldous/RWG/book.html`.

[4] Charles J. Geyer. Introduction to Markov chain Monte Carlo. In *Handbook of Markov chain Monte Carlo*, Chapman & Hall/CRC Handb. Mod. Stat. Methods, pages 3–48. CRC Press, Boca Raton, FL, 2011.

[5] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[6] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[7] F. Sullivan and I. Beichl. The Metropolis Algorithm. *Computing in Science & Engineering*, 2(1):65–69, 2000.

[8] Daryl DeFord, Moon Duchin, and Justin Solomon. Recombination: A family of Markov chains for redistricting. *arXiv:1911.05725*, 2019.

[9] Elle Najt, Daryl DeFord, and Justin Solomon. Complexity of sampling connected graph partitions. *arXiv:1908.08881*, 2019.

[10] Benjamin Fifield, Michael Higgins, Kosuke Imai, and Alexander Tarr. A New Automated Redistricting Simulator Using Markov Chain Monte Carlo. *Working paper: Princeton University*, page 55.

[11] Gregory Herschlag, Han Sung Kang, Justin Luo, Christy Vaughn Graves, Sachet Bangia, Robert Ravier, and Jonathan C. Mattingly. Quantifying Gerrymandering in North Carolina. *arXiv:1801.03783 [physics, stat]*, 2018. arXiv: 1801.03783.

[12] Maria Chikina, Alan Frieze, and Wesley Pegden. Assessing significance in a Markov chain without mixing. *Proceedings of the National Academy of Sciences*, 114(11):2860–2864, 2017.

[13] Maria Chikina, Alan Frieze, Jonathan Mattingly, and Wesley Pegden. Practical tests for significance in Markov chains. *arXiv:1904.04052*, 2019.

[14] Julian Besag and Peter Clifford. Generalized Monte Carlo significance tests. *Biometrika*, 76(4):633–642, 1989.

[15] Peter J. Cameron. *Combinatorics: Topics, Techniques, Algorithms*. Cambridge University Press, 1994.

[16] Moon Duchin and Bridget Tenner. Discrete geometry for electoral geography. *arXiv:1808.05860*, 2018.

[17] Sarah Cannon, Moon Duchin, Dana Randall, and Parker Rule. A reversible recombination chain for graph partitions. *preprint*, 2020.

[18] Daryl DeFord and Moon Duchin. Chapter 12 repo. *GitHub repository*, 2021.

[19] Max Palmer. Expert Report for *bethune-hill v. virginia state board of elections*, 2017.

[20] Benjamin Fifield, Kosuke Imai, Jun Kawahara, and Christophe Kenny. The essential role of empirical validation in legislative redistricting simulation. *preprint*, 2019.

[21] Daryl DeFord and Moon Duchin. Redistricting reform in Virginia: Districting criteria in context. *Virginia Policy Review*, 12(2):120–146, 2019.

[22] Daryl DeFord, Natasha Dhamankar, Moon Duchin, Varun Gupta, Mackenzie McPike, Gabe Schoenbach, and Ki Wan Sim. Implementing partisan symmetry: Problems and paradoxes. *Political Analysis*, to appear, 2021.

[23] Sophia Caldera, Daryl DeFord, Moon Duchin, Sam Gutekunst, and Cara Nix. Mathematics of nested districts: The case of Alaska. *Preprint: https://mggg.org/uploads/Alaska.pdf*, 2019.

[24] Daryl DeFord, Moon Duchin, and Justin Solomon. A Computational Approach to Measuring Vote Elasticity and Competitiveness. *Preprint*, pages 1–30, 2019.

[25] Wasserman D. Bycoffe A., Koeze E. and Wolfe J. The atlas of redistricting, 2018. `https://projects.fivethirtyeight.com/redistricting-maps/`.

[26] Mira Bernstein, Moon Duchin, Tommy Ratcliff, and Stephanie Somersille. Study of voting systems for Santa Clara, California. *MGGG Technical Report*, pages 1–16, 2018.

[27] Ruth Buck, Moon Duchin, Dara Gold, and JN Matthews. Community-Centered Redistricting in Lowell, Massachusetts. *MGGG Technical Report*, pages 1–8, 2020.

[28] Hakeem Angulu, Ruth Buck, Daryl DeFord, Moon Duchin, Howard Fain, Max Hully, Maira Khan, Zach Schutzman, and Oliver York. Study of Reform Proposals for Chicago City Council. *MGGG Technical Report*, pages 1–31, 2020.