Empirical Validation of Automated Redistricting Algorithms on the Virginia House of Delegates District Map

Madeleine Goertz

International Community School

AP Research

Randall S. Huberman

May 20th, 2021

Empirical Validation of Automated Redistricting Algorithms on the Virginia House of Delegates District Map

Redistricting

- Explain what districts are
- Explain how the redistricting process generally works

Gerrymandering

- Explain what gerrymandering is.

Ways to combat gerrymandering

Automated Redistricting Algorithms

- Explain what the goals of these things are

Metrics to detect gerrymandering

- Explain what the goals of these are

Overview of Method

- Actually introduce the rest of the paper.

Literature Review

Detection of Gerrymandering

- Explain existing research that focuses on whether or not existing maps are gerrymandered

- This is good, but given the move to nonpartisan commissions/collaborations within the state legislatures, it's necessary to have fair methods to generate the districts in the first place.

- This is why we need.....

Automated Redistricting Algorithms

- Explain purpose of these

- Give some of the first examples

- My research uses three of these, go in depth explaining them.

Markov Chain Monte Carlo

The first automated redistricting algorithm that I'm going to discuss in detail is an implementation of a statistical method known as "Markov Chain Monte Carlo," henceforth MCMC (Fifield, Higgins, et al., 2020).[1] The following overview is meant to provide a high-level understand of how the basic algorithm works. Since the purpose of my research is to compare the *outputs*, that is the redistricting plans, generated by these algorithms, *rather than the algorithms themselves*, a rigorous understanding of the algorithms is not a prerequisite.[2].

---

[1] I will refer to the *redistricting algorithm* that uses Markov Chain Monte Carlo as "MCMC," rather than the *statistical method* Markov Monte Carlo.

[2] Please see the section "The Proposed Methodology" in the paper for an in-depth, mathematically-rigorous explanation. (Fifield, Higgins, et al., 2020, p. 2)

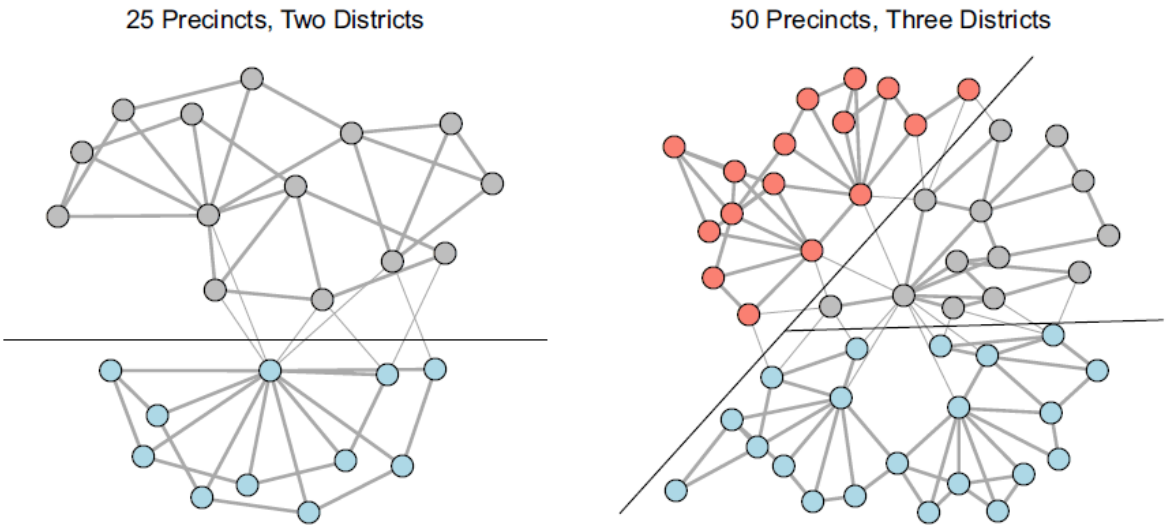

Figure 1

Representation of redistricting as graph cutting. Every node is a precinct, and nodes that share an edge are known to be adjacent precincts. MCMC "cuts away" edges between nodes until islands of districts are formed. (Fifield, Higgins, et al., 2020, p. 3)

MCMC conceptualizes the problem of redistricting precincts as a graph-cutting problem. For the uninitiated, a graph is a network of different interconnected points, where the points are called "nodes" and the lines connecting them are called "edges" (Fifield, Higgins, et al., 2020). MCMC represents every precinct as a node, and it draws an edge between nodes where the corresponding precincts are adjacent. Since the goal of redistricting is to assign every precinct a district, MCMC imagines that edges between nodes are "cut" until "islands" (known as "sub graphs") are formed, which each is disconnected from the rest. The disconnected "sub graphs" then become the districts. Figure 1 provides a nice visualization of this representation with a sample set of 50 precincts. (Fifield, Higgins, et al., 2020)

Now that the representation used by MCMC has been established, we can discuss the specifics of the algorithm. Figure 2 visualizes the steps in this algorithm. The basic MCMC algorithm begins with a valid redistricting plan, such as the one currently in use,

and randomly decides to "turn on" some edges in the graph. Then, the nodes (precincts) that are connected by these "turned on" edges and are located on the boundary of a district are identified. Then, these highlighted graph components are "nominated" for a swap across the district boundary based on some probability, provided that this swap would not break the district into two. Lastly, this new proposed redistricting plan is either accepted or rejected based on some "acceptance probability." This process is repeated as many times as desired. (Fifield, Higgins, et al., 2020).

MCMC in its current form allows for further constraint of this algorithm, particularly with regard to the relative population size and compactness of each district. (Fifield, Higgins, et al., 2020, p. 6) The explanation of this algorithm is beyond the scope of this paper.

Sequential Monte Carlo

The second automated redistricting algorithm that I'm going to discuss is an implementation of a statistical method known as "Sequential Monte Carlo," henceforth SMC (McCartan & Imai, 2020).[3] The following overview is meant to provide a high-level understand of how the basic algorithm works. [4].

Just like MCMC, SMC conceptualizes the electoral map as a mathematical graph with precincts as nodes and edges connecting geographically-adjacent precincts. It also uses this graph-cutting concept, but SMC specifically uses something called a "spanning tree" (see Figure 3), which is a graph that is connected by the minimum number of possible edges[5].

Figure 4 visualizes the iterative splitting procedure used by SMC. First, compute

---

[3] I will refer to the *redistricting algorithm* that uses Sequential Monte Carlo as "SMCMC," rather than the *statistical method* Sequential Monte Carlo.

[4] Please see the section "The Proposed Algorithm" in the paper for an in-depth, mathematically-rigorous explanation. (McCartan & Imai, 2020, p. 13)

[5] Put another way, if any edge is cut from the graph, the graph will be split into two sub graphs.

the deviation from the target precinct population for each subgraph generated by cutting each edge in the spanning tree. Then, randomly cut one of the edges, creating two new subgraphs[6] If the smaller subgraph meets the population and compactness requirements, then it's accepted as the first district, and the splitting procedure is repeated with the other subgraph.

This process generates the possible redistricting plans that satisfy the requirements. For more details, please refer to McCartan and Imai (2020).

Compact Random Seed Growth

The final automated redistricting algorithm that I'm comparing is called Compact Random Seed Growth, henceforth referred to as "CRSG" and was proposed by Chen and Rodden (2013). Its objective is to generate a set of districts that fall within a certain population constraint and are reasonably compact using only the geography and total population of each precinct (Chen & Rodden, 2013). The following is a high-level explanation of the algorithm.[7]

CRSG begins with declaring that every precinct is it's own district. A random precinct is then chosen, and then its geographically-closest[8] neighbor is merged with it, creating one fewer district. This process is repeated until you arrive at the desired number of districts. (Chen & Rodden, 2013, pp. 249–50)

After this procedure, the districts are somewhat compact due to the geographic proximity requirement, but there is no guarantee that the districts are within the required population percentage of each other.

To satisfy the population parity requirements, CRSG does the following. First, it

---

[6] Technically they're spanning trees, which are known as a spanning forest in the plural.

[7] For more details, please refer to Chen and Rodden (2013, pp. 249–50).

[8] The geographically-closest precinct is the neighboring precinct with the smallest distance from its centroid to the seed precinct's centroid

identifies the two adjacent districts that have the greatest difference in total population. Then the precinct in the more-populous district that is furthest from the center of said district is reassigned to the less-populous district.[9] This process is repeated until all of the districts are within some desired percentage of the mean district population.Chen and Rodden (2013, pp. 249–50).

One run of CRSG will produce one set of districts, but separate runs of CRSG with the same input data may produce slightly different districts given the random choice of districts to merge in the first pass of the algorithm.

Empirical Validation

- Usually small-scale validation within the papers that propose these Algorithms

- Needed to see how well the algorithms scale

- For a commission, it's helpful to be able to compare the results of several different leading algorithms.

Evaluation of Redistricting Plans

Katz et al. (2020) brings mathematical rigor to the various proposed metrics for measuring partisan symmetry.

Partisan Symmetry

A legislative is said to have partisan symmetry if both parties can receive m proportion of the overall votes and therefore have n proportion of the seats in the legislative body.An example would be that if Republicans win 60% of the votes but control 65% of the seats, then in a symmetrical system, Democrats should also be able to control 65% of the seats by winning 60% of the votes. Katz et al. (2020)

Partisan symmetry is usually observed by plotting a "seats-votes curve." This plot has V, the proportion of the overall votes won by the party, on the x-axis, and $S(V)$, the

[9] Provided that this reassignment doesn't break either district into parts.

proportion of the seats won by the party, on the y-axis. Figure 5 (Katz et al., 2020, p. 175) illustrates several hypothetical seats-votes curves.

Naturally, it's very rare to observe the necessary electoral outcomes under the same electoral system in order to determine partisan symmetry. (ie., it's very rare for two parties to tie one year, have one win 51% of the total votes the next year, and then win 49% of the votes the following year.)

In practice, one can estimate a seats-votes curve using the principle of uniform partisan swing.

Uniform Partisan Swing

Uniform partisan swing is the principle that when the overall vote share V changes by some amount dV between different elections under the same electoral system, the vote share at the district level also usually changes by the same dV. Katz et al. (2020) Empirically verified this to be true in 646 different elections.

Thus, given a list of vote proportions per district $v_1, v_2, v_3, ...$ from one election, one can adjust each vote proportion by an arbitrarily small dV until the seats share $S(V)$ is covered from 0 to 1. (Katz et al., 2020).

An example of such a curve is shown in Figure 6

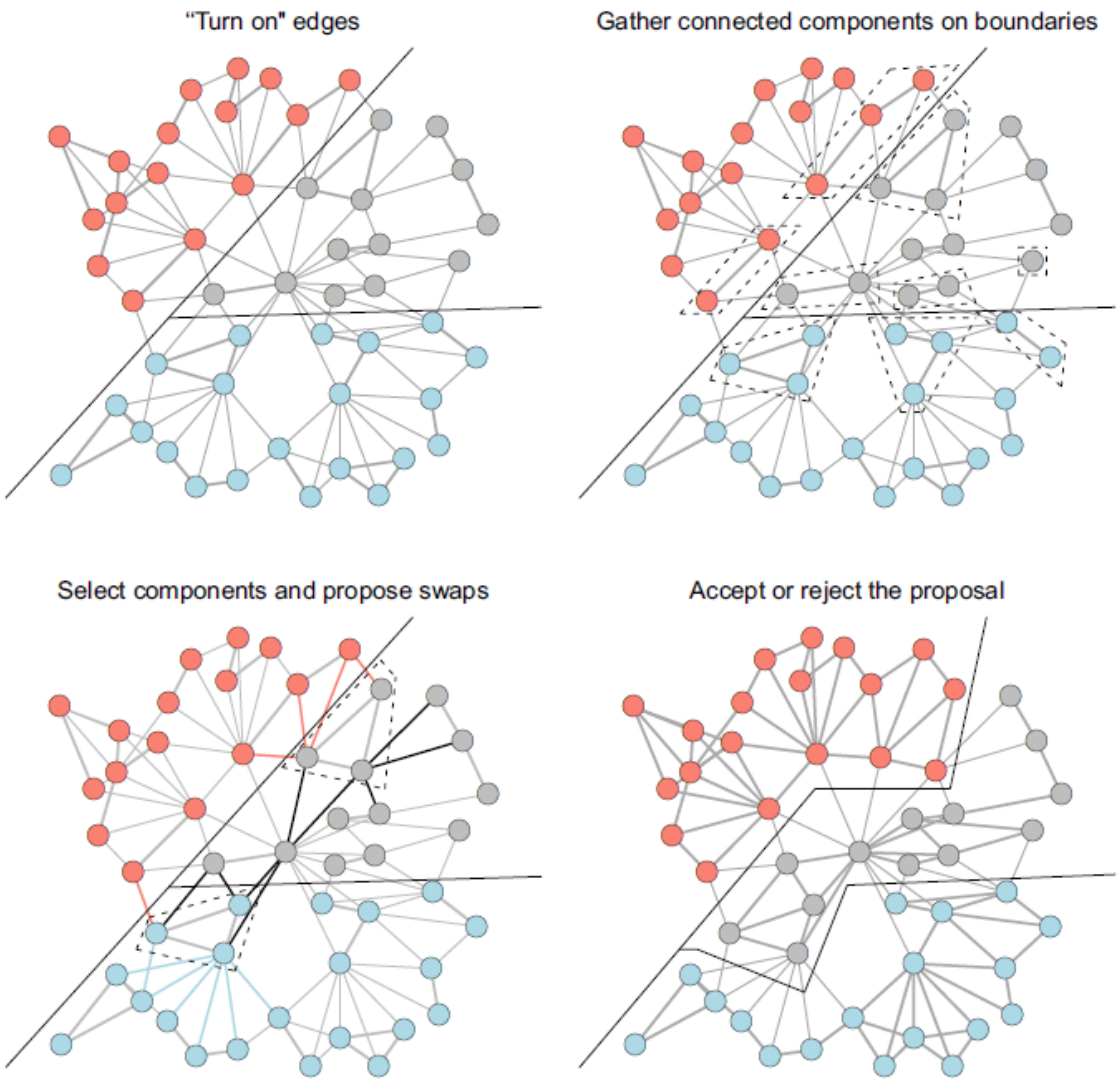

Figure 2

Representation of MCMC algorithm. (Fifield, Higgins, et al., 2020, p. 4)

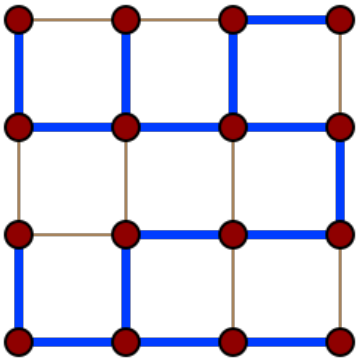

Figure 3

An example spanning tree. The spanning tree consists of all nodes and the blue edges, where the complete graph includes the grid edges as well. (Eppstein, 2007)

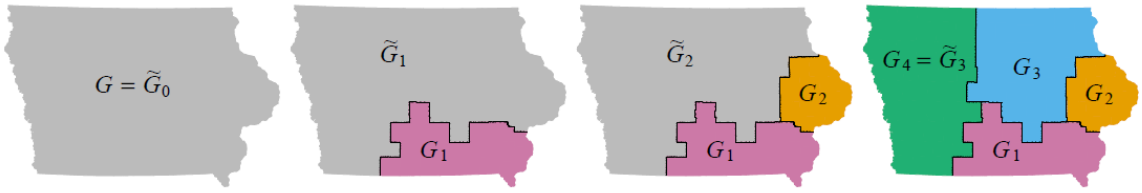

Figure 4

Representation of splitting procedure of SMC. Every node is a precinct, and nodes that share an edge are known to be adjacent precincts. MCMC "cuts away" edges between nodes until islands of districts are formed. (McCartan & Imai, 2020, p. 14)

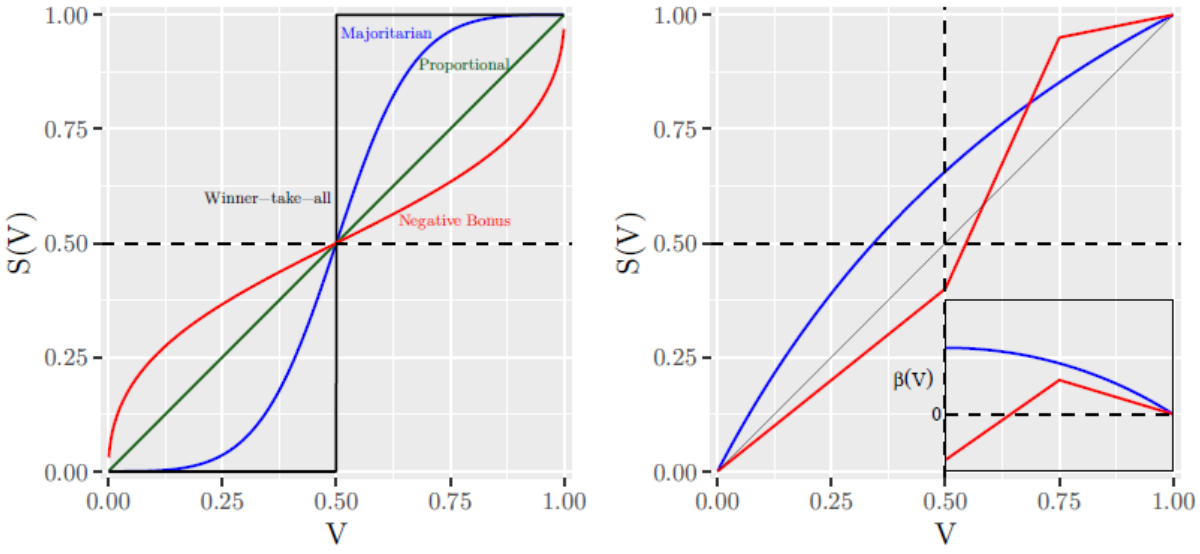

Figure 5

Types of Seats-Votes Curves. Left panel: Symmetric (fair) curves with differing levels of electoral responsiveness. Right panel: Asymmetric (biased) curves, including one consistently biased toward the Democrats (blue) and one with biases favoring different parties depending on V (red); the inset graph is for

(V) for V 2 [0:5; 1] with the vertical axis scaled to be the same as the main plot, and lines color coded to the seats-votes curves. (Katz et al., 2020, p. 175)

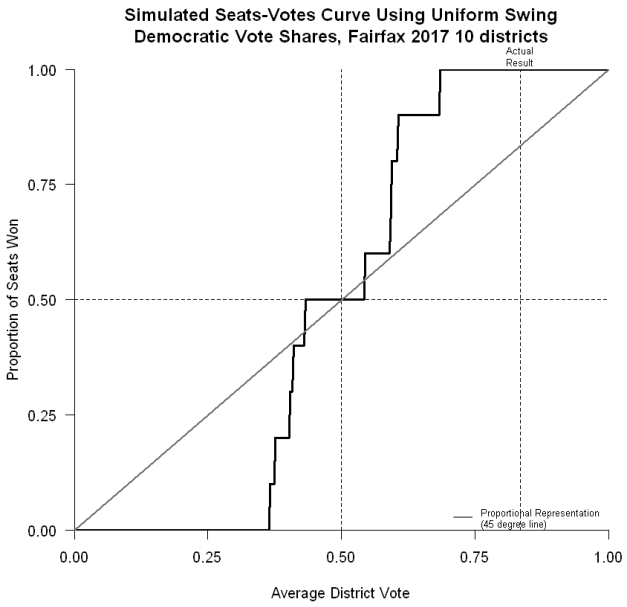

Figure 6

Sample seats-votes curve generated using uniform partisan swing. Ignore title. (Katz et al., 2020, p. 175)

Method

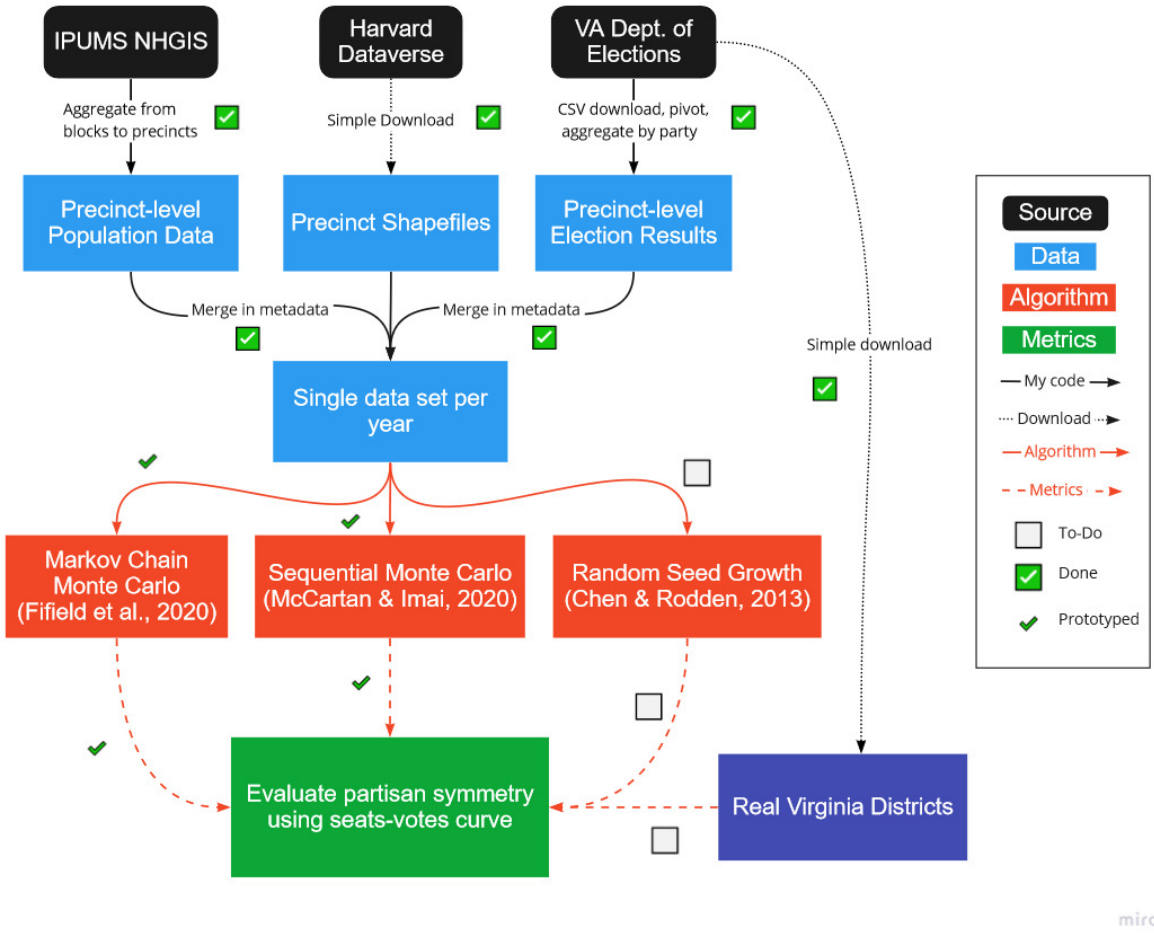

Figure 7

Graphical Overview of project

My research method simulates redistricting the legislative districts for the Virginia House of Delegates using three different algorithms in the years 2015, 2017, and 2019. Figure 7 provides and overview of this process. I begin by explaining my overall research method, explaining how I'm aligned to both the components and principles of experimental designs, and then explaining the data collection and cleaning process.

Choice of Research Method

For this study, I chose to use the experimental design method because it will allow me to isolate the hypothetical impact of the redistricting algorithm from other possible confounding variables. This method also includes the use of a control group, which allows the researcher to establish causation.

Components of Experimental Design

Experimental Units. The experimental units for this study are the complete datasets for each election year in Viriginia.[10] I have one dataset for each of these years: 2015, 2017, 2019. Every row in each dataset corresponds to a precinct, the smallest geographical unit by which votes are tabulated in Virginia. For each precinct, I have the following attributes: total population, population by race, total voting-age population (VAP)(population over the age of 18), VAP by race, total votes for the democratic House of Delegates (HOD) candidate, total votes for the Republican HOD candidate, and the total votes for any other HOD candidate. Additionally, each precinct has a polygon associated with it that represents its geographical shape.

Treatments. The treatments for this study are the three different redistricting algorithm that I'm comparing: Markov chain Monte Carlo (Fifield, Higgins, et al., 2020), Sequential Monte Carlo (McCartan & Imai, 2020), and Random Seed Growth (Chen & Rodden, 2013).[11] I'm using the implementations in the R programming language "redist" package (Fifield, Kenny, et al., 2020). See the Literature Review section for a deeper dive into these algorithms. Broadly, I chose them because they are deterministic. Much of the literature focuses on creating many possible redistricting plans for a commission to choose from, but these three aim to create an "ideal" map.

---

[10] This corresponds to the blue rectangles in Figure 7.

[11] This corresponds to the red rectangles in Figure 7

Response Variables. Broadly, the goal will be to evaluate how "fair" each redistricting plan generated by each algorithm for each year is.[12] Within the literature, partisan symmetry is the primary principle used to evaluate redistricting plans.

Chamber Power Balance. Since the redistricting that's occuring is hypothetical and I have precinct-level election results for each of these years, I can simulate what the power distribution in the VA House of Delegates would be if the proposed redistricting plan had been used.

Control Group. The official VA House of Delegates map used in the years 2015-2019 will serve as the control group for this experiment. I will compute the same metrics for this map as I will for my hypothetical redistricting plans.[13]

Principles of Experimental Design

The primary principles of experimental research design are randomization, replication, and local control. This is how I plan to address them.

Randomization. Every experimental unit will receive each treatment, and every experimental unit can be replicated many times without issue, so thereâĂŹs no error from a lack of randomization. Think of each treatment operating within a separate parallel universe.

Replication. There is no need for me to run my trials several times (run the same algorithm on the same data set several times) because these are deterministic algorithms, and the datasets will be immutable.

Local Control. All of the redistricting will be happening in controlled environments, so there will be no way for lurking variables to creep in and confound my results.

---

[12] This corresponds to the green rectangles in Figure 7.

[13] This corresponds to the purple rectangle in Figure 7.

Data Cleaning

To create my datasets, I cleaned and compiled three different types of data: demographic data, Geographic Information Systems (GIS) data, and election data.[14]

Demographic Data

One required piece of data in order to redistrict is demographic data at the precinct level. This means both the total population and the Voting-Age Population (VAP) broken down into the Non-Hispanic White, Non-Hispanic Black, and Hispanic categories. In order to run the most accurate redistricting simulations, these data needed to be recent for the year being redistricted, meaning I needed different data sets for 2015, 2017, and 2019. Comprehensive population counts are only conducted by the US Census Bureau every 10 years, so I instead used the 5-year American Community Survey results at the block-group level. This is a sample survey, not a population count, but that is offset by the aggregation of sample data over a 5 year period. I downloaded this data from the IPUMS National Historic GIS project (Manson et al., 2020). Using the "maup" Python Library (Hully, n.d.), I disaggregated the data from the block-group level to the block level, prorating the demographic data based on population. This data was then aggregated up to the precinct level.

However, IPUMS did not have VAP by race data, so I downloaded that separately from the US Census Bureau (Bureau, n.d.) and cleaned it in a similar fashion, merging it into my precinct tables.

GIS Data

In order to redistrict, the algorithms need to know the shape and relative location of each precinct. In practice, this means every precinct has a "polygon" associated with it and a Coordinate Reference System that describes where these polygons fall in space. These

[14] This is an explanation of the black and blue rectangles in Figure 7 and the transitions between them.

data tables with the geometry column are known as "shapefiles." I accessed these shapefiles from the Voting and Election Science Team on their Harvard Dataverse (Voting and Election Science Team, 2019a, 2019b, 2019c). I then merged in my precinct-level demographic data tables, so I now have shapefiles with the necessary demographic data. [15]

Election Data

The last necessary component needed to evaluate redistricting plans is the number of votes one by each party in each precinct in each election.[16] The Virginia Department of Elections publishes historical records of every election on their website (Virginia Department of Elections, n.d.), which I cleaned and aggregated to arrive at a party vote count for each precinct for each year. These data were then merged into my precinct shapefiles.

---

[15] Since election administrators are free to change the precincts between elections, precinct shapefiles are unique to both a place and a time. This was the reason that I only ran simulations in the years 2015, 2017, and 2019, since these were the only years for which I was able to find reputable precinct shapefiles.

[16] The algorithms I'm comparing assume a 2 party system, so I only tracked Democratic and Republicn votes won in each election.

References

Bureau, U. C. (n.d.). Citizen Voting Age Population by Race and Ethnicity. Retrieved February 2, 2021, from https://www.census.gov/programs-surveys/decennial-census/about/voting-rights/cvap.html

Chen, J., & Rodden, J. (2013). Unintentional Gerrymandering: Political Geography and Electoral Bias in Legislatures. *Quarterly Journal of Political Science*, *8*(3), 239–269. https://doi.org/10.1561/100.00012033

Eppstein, D. (2007). English: A 4x4 Grid Graph and One of Its Spanning treesFrançais : Une Grille 4x4 et Un de Ses Arbres Couvrants. Retrieved March 1, 2021, from https://commons.wikimedia.org/wiki/File:4x4_grid_spanning_tree.svg

Fifield, B., Kenny, C. T., McCartan, C., Tarr, A., Higgins, M., Kawahara, J., & Imai, K. (2020). Redist: Simulation Methods for Legislative Redistricting. Retrieved January 29, 2021, from https://CRAN.R-project.org/package=redist

Fifield, B., Higgins, M., Imai, K., & Tarr, A. (2020). Automated Redistricting Simulation Using Markov Chain Monte Carlo. *Journal of Computational and Graphical Statistics*, *0*(0), 1–14. https://doi.org/10.1080/10618600.2020.1739532

Hully, M. (n.d.). Maup: The Geospatial Toolkit for Redistricting Data. Retrieved February 18, 2021, from https://github.com/mggg/maup

Katz, J. N., King, G., & Rosenblatt, E. (2020). Theoretical Foundations and Empirical Evaluations of Partisan Fairness in District-Based Democracies. *American Political Science Review*, *114*(1), 164–178. https://doi.org/10.1017/S000305541900056X

Manson, S., Schroeder, J., Van Riper, D., Kugler, T., & Ruggles, S. (2020). National Historical Geographic Information System: Version 15.0. https://doi.org/10.18128/D050.V15.0

McCartan, C., & Imai, K. (2020). Sequential Monte Carlo for Sampling Balanced and Compact Redistricting Plans. *arXiv:2008.06131 [cs, math, stat]*. Retrieved January 18, 2021, from http://arxiv.org/abs/2008.06131

Virginia Department of Elections. (n.d.). Results/Reports. Retrieved January 13, 2021, from https://www.elections.virginia.gov/resultsreports/

Voting and Election Science Team. (2019a). 2015 Precinct-Level Election Results. https://doi.org/10.7910/DVN/KTXHEW

Voting and Election Science Team. (2019b). 2017 Precinct-Level Election Results. https://doi.org/10.7910/DVN/VNJAB1

Voting and Election Science Team. (2019c). 2019 Precinct Shapefiles. https://doi.org/10.7910/DVN/A0VJ3B