Counting Votes, Counting Seats: Fairness in the 2020 U.S. House Elections

ESE 3090 Special Topics in Systems Engineering: Modeling and Design of Social Choice Systems

Emily Zeiberg
Electrical and Systems Engineering
Washington University in St. Louis
St. Louis, Missouri, USA
e.p.zeiberg@wustl.edu

Michael Zaslavskiy
Electrical and Systems Engineering
Washington University in St. Louis
St. Louis, Missouri, USA
m.n.zaslavskiy@wustl.edu

***Abstract*—This document provides an in-depth evaluation of election fairness metrics for the U.S. House of Representatives across all 50 states. Through the development of a MATLAB script to analyze proportionality, partisan symmetry, and efficiency gap fairness, our group utilizes 2020 House of Representatives election data to understand voting fairness among republican, democratic, and swing states. Our paper considers the social importance of redistricting analysis, outlines our design methods for this study, and discusses the implications of our learnings in understanding how our election systems are made to either represent, or misrepresent, our political interests.**

I. Introduction

A. What is "Fairness"?

The question of fairness in redistricting is both philosophical and mathematical. Fairness is ubiquitously defined as the exercise of judgment free from impartiality - though easily theorized, it is very challenging to integriously exercise [1]. In the case of voting, fairness can be interpreted as an electoral outcome that accurately represents the collective political preferences of voters without manipulating election results. Redistricting plays a critical role in this by determining which groups of voters are given more or less decision power. Notoriously, districts come in suspiciously unconventional shapes in an effort to group certain types of voters, leading to a very specific electoral outcome desired by those within power. However, unconventionally-shaped districts can also be used to give minority voters more of a voice, going back to the challenge of measuring fairness and creating a system that accurately represents a population's political preferences.

B. The U.S. House of Representatives

To establish the U.S. House of Representatives, each state is allotted some number of representatives that is proportional to the number of people living in that state. Within a state, voters are divided into groups, called districts, based on where they reside. Each district elects one representative to serve in the U.S. House of Representatives, and ideally, that representative has the same beliefs and political affiliations as the people living in the district they represent. Because populations in each district change overtime, district maps are redrawn every 10 years, following the census, to ensure that the population living in each district is roughly equal. However, the way that each district is drawn greatly affects which representative is elected. This creates opportunities for mapmakers to gerrymander, which is the practice of manipulating districts in a way to improve one party's chance of winning.

The Voting Rights Act of 1965 was established to prevent gerrymandering. This act helps to ensure that everyone, regardless of race or ethnicity, has equal access to voting and that their vote is not diluted in any way [2]. In terms of redistricting, this means that a minority group living in a condensed area should not be divided into multiple districts. This increases the chances that the representative elected from that district represents the ideals of that minority group. If the group had been separated, their votes would have been diluted by the majority group, and the minority group would not be fairly represented in Congress. The challenge of redistricting is further amplified by other general redistricting requirements, such as respecting established borders and creating condense and contiguous districts. Despite all these regulations, gerrymandering still exists and is a prominent problem. Fairness metrics can be used to determine if gerrymandering has occurred, but depending on the way fairness is defined, different conclusions can be reached.

II. Methods and Technical Approach

This section explains our approach to creating our script – from data collection, to the background and implementation of the fairness metrics we utilized, to the visualization techniques of our data.

We discuss the three fairness metrics – proportionality, partisan symmetry, and efficiency gap – that were used to determine whether district maps for the 2020 election were unfair or if there were other factors that led to these results. The definition for each metric came from the Duchin, Walch textbook [3].All data came from the Federal Election Commission website [4].

A. Data Collection

Finding data proved to be one of our greatest challenges. We researched a variety of websites – from state to federal legislatures, until we identified a large excel document on the Federal Election committee. Before finding this document, however, we also attempted to collect data via web scraping from the CNN's 2020 House of Representatives election website.

None of our team members had experience with using Python for web scraping, which posed to be a tough challenge. We first familiarized ourselves with basic web scraping capabilities using an example website and the BeautifulSoup Python library. When we understood how to scrape basic headers, titles, and body text from websites, we felt confident attempting to web scrape from the CNN website. We attempted to first collect all House information for Pennsylvania's 2020 House Election results [5].

An obstacle we encountered was that CNN loads its data via JavaScript, so not all data is in its HTML page. Given these more complicated features, we needed to use Selenium, another Python library with more advanced web scraping tools. To identify specific elements, we had to use the inspect view of the CNN website to identify specific HTML class tags for our Selenium script. This proved to be very arduous and ineffective, provided that sifting through large amounts of div, select, and paragraph HTML tags gave room for human error. Even when human error was circumvented, we ran the risk of not actually finding the text that was displayed because, as mentioned before, rather than being embedded into the website's HTML, it was being loaded through JavaScript, which we didn't know how to access.

In a final effort to web scrape, we found a tool called Octoparse that would work as an artificial intelligence web scraping agent. All we needed to do was identify the moments on the screen the program would need to click on/ record. This also proved futile, however, given that Octoparse wasn't able to recognize a drop-down needed to go through each district's data within a given state. Having given it a chance, we decided that web-scraping would be too challenging to accomplish in such a short period of time, so we moved forward with the large excel document we found on the Federal Election Commission's website.

The following sections discuss each of the fairness measurements used and how they were implemented.

B. Proportionality

Proportionality measures fairness by looking at whether the number of votes a political party received is proportional to the number of seats that the party received. Ideally, if Republicans received 40% of the state's vote, then Republicans should occupy 40% of the state's seats in Congress. This metric is relatively straightforward to calculate. The vote-share for one party is compared to the seat-share of that same party, and if these ratios are within 5% of each other, then the map is considered proportional. We chose a value of 5% to closely abide by an example posed in the Duchin and Walch textbook. The definition of proportionality is summarized in equations 1 and 2.

$$R_{VS} = \frac{V_R}{V_R + V_D} \tag{1}$$

$$R_{SS} = \frac{S_R}{S_R + S_D} \tag{2}$$

R_{VS} is the Republican vote share, V_R is the number of votes received by Republicans, and V_D is the number of votes received by Democrats. Likewise, R_{SS} is the Republican seat share, S_R is the number of seats held by Republicans, and S_D is the number of seats held by Democrats.

At a very high level, we implemented this calculation first by counting total votes and winners for each party within the selected state, then calculating the respective percentages and comparing them to the percentages of elected officials and their correlating political parties.

C. Partisan Symmetry

Partisan symmetry measures voting fairness by assessing whether political parties would receive equivalent seat shares if their vote shares were swapped, ensuring no inherent advantage in electoral outcomes. For example, if Democrats received 53% of votes and secured 72% of House Seats, then when the Republicans were to win 53% of votes, they would also have to secure around 72% of House seats for this map to be considered fair. What matters here is the concept of a "winner's bonus" – that is, though the number of votes to occupied seats is not proportional, the ratio of votes to seats is equivalent for both Democrats and Republican electoral outcomes if either would win, respectively. This means in the long term that as different elections occur, one election may not equally represent all republican or democratic votes, but over multiple elections, their cumulative representation will be equivalent.

Changes in individual district vote shares are modeled using the uniform partisan swing (UPS) model:

$$v_i' = v_i + \delta \quad \text{for } i = 1, \ldots, N \tag{3}$$

where v_i is the original vote-share that a party received in district i, and v_i' is the vote-share of the party in district i after the original vote-share is adjusted by δ. The partisan symmetry score is the calculated by the following:

$$S(V) = 1 - S(1 - V) \tag{4}$$

$$\beta(V) = \frac{S(V) - (1 - S(1 - V))}{2} \tag{5}$$

$$\int_0^1 |\beta(V)|\, dV = \int_0^1 |S(V) - (1 - S(1 - V))|\, dV \tag{6}$$

where S(V) is a function that produces the seat-vote curve, S(1-V) is and 180° rotation of that curve, and $\beta(V)$ is a measure of how asymmetrical S(V) is. An example of the Partisan Symmetry gap for North Carolina from our MATLAB analysis is illustrated below.

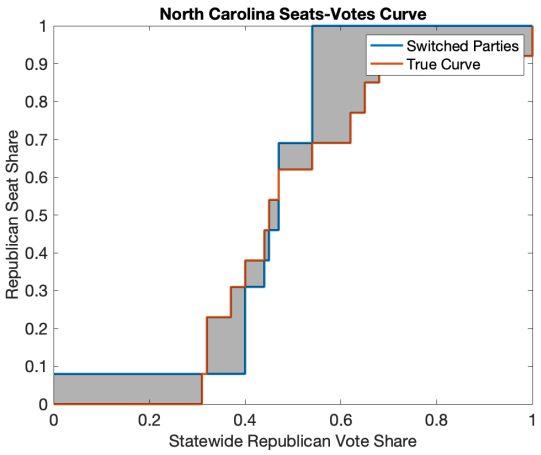

Fig. 1. Example of North Carolina Partisan Symmetry Gap

This map can be interpreted by considering the relationship between both horizontal and vertical axes. For example, when the republicans have 0.4 percent vote share, they have 0.3 percent of seat share. When the Democrats have 0.6 percent of vote share, they have 0.7 percent of seat share. In this type of graph, correlating republican and democratic vote and seat shares should add up to 1. This example is not symmetric, however, given that the ratio of percentage of vote share to seat share is not preserved between republican and democratic parties.

It's important to note that though we produced these graphs to provide a better visual understanding when looking at each individual state, the visual graphics were neither used nor referenced in our later analysis discussed below. This is because the numeric symmetry indices were more important from these graphs in our calculation of partisan symmetry fairness metrics rather than visual lines and shadows.

When identifying a threshold value for determining whether or not a state is symmetric, we observed how different values affected the map output. We noted that maps with a value of 3% or less clearly illustrated characteristics of a symmetric map.

At a very high level, we implemented partisan symmetry first by calculating a party's vote shares within the state, then modeling vote shifts (i.e. how election results would change if the Republican vote share shifted from 0 to 100 percent in 1 percent increments), and finally calculating partisan symmetry utilizing equations 4, 5, and 6 listed above.

D. Efficiency Gap

The efficiency gap uses the number of "wasted" votes to determine if a map is fair or not. A vote is considered wasted if it does not directly lead to that candidate being elected. Since 50% of votes + 1 is needed to elect a candidate, votes for the winning party in excess of this are considered wasted. Also, any votes for the losing candidate are considered wasted. The goal of the efficiency gap metric is to measure if a district is "packed" or "cracked". If a district is "packed", more people with the same political beliefs are kept together, and the elected official has the same beliefs as most of the people living in their district. If a district is "cracked", then that district represents multiple communities, and the representative has the same beliefs as roughly half of the district's population. A low efficiency gap is equivalent to packing, and high efficiency gap is equivalent to cracking. Packing is preferred since one of the goals of drawing district maps is keeping communities of interest together. The efficiency gap can be calculated using equation 7.

$$EG = \frac{|W_B - W_A|}{T} \tag{7}$$

where W_B is the number of wasted votes for party B, W_A is the number of wasted votes for party A, and T is the total number of votes for parties A and B. In the context of packing, the efficiency gap is lower because the additional quantity of votes for the winning party is similar to the overall quantity of votes for the losing party, meaning that most people of the same political interests are in a single district. It's important to recognize, though, that for an efficiency gap to be low, there must be votes of another party – otherwise, the quantity of wasted votes will be very high due to an excess of votes for the winning party. A too high efficiency gap, therefore, not only signifies cracking, but can also mean too many individuals of a single party are within a specific district.

For the purposes of our analysis, we used an efficiency gap threshold of 0.08. As proposed by Stephanopoulos and McGhee [6], if the efficiency gap is less than 0.08, then the map is considered fair, and if it's above this threshold, the map could indicate that gerrymandering occurred.

At a very high level, we implemented efficiency gap calculations by first by performing district-by-district analysis on the Republican and Democratic votes wasted per district. We then utilized equation 7 to calculate the efficiency gap for the the state.

E. Code Logic and Visualization

For our project, we created code to be utilized in two distinct programs: the first to provide detailed metrics for a desired input state, and the second to create larger mappings of fairness measurements for all states across the U.S..

For our first program, we wanted to allow for a user to input a state and receive specific data explaining the fairness measurements for that state. An example of such output for North Carolina is detailed below:

```
Enter state name (with the first letter capitalized): North Carolina
---------------
PROPORTIONALITY
Percent of votes for Republican candidates: 49.72%
Percent of votes for Democratic candidates: 50.28%
Percent of elected representatives that are Republican: 61.54%
Percent of elected representatives that are Democrats: 38.46%
This districting map is not proportional.
---------------
PARTISAN SYMMETRY
Partisan Symmetry score: 0.41
This districting map is not symmetrical.
---------------
EFFICICENCY GAP
Number of wasted votes for the Democratic party: 1622855
Number of wasted votes for the Republican party: 1023071
Efficiency gap: 0.11
This districting map is not efficient.
---------------
```

Fig. 2. Example of North Carolina Fairness Metrics from our First Program

For this program, our logic was fairly straightforward: the user inputs a state, we find all the data required for our fairness metrics calculations associated with that state, and then we compute the proportionality, partisan symmetry, and efficiency gap measurements.

For our second program, we needed to put this main code into a for loop that would iterate through all the states to output a visual diagram of the fairness measurements for the entire country. In this for loop, we filled proportional, partisan symmetry, and efficiency gap arrays with ”1”s at indexes where a state passed the fairness measurements, and ”0”s at indexes where states did not pass those measurements. We then were able to use MATLAB’s mapping library to plot a color-coded version of the U.S. map for each fairness measurement based on the binary values within each of these fairness arrays. Those maps are illustrated in the following section.

Finally, we wanted to see how all of these fairness measurements overlapped to better understand if states with a certain political preference were more ”fair” than other states. To do so, we concatenated the proportionality, partisan symmetry, and efficiency gap binary arrays, and used each row of this new matrix as an RGB value. This assigned eight unique colors to eight unique combinations of fairness for all the states. This overarching, final diagram is pictured in the following section, and allowed us to better analyze how Republican, Democratic, and swing states each generally stacked up against all three fairness measurements.

III. Results and Discussion

A. Collective Fairness Metrics Results

As aforementioned, we focused on understanding the fairness measurements for Democratic, Republican, and swing states in the 2020 election to see if there was some sort of relationship between the political affiliation and district map fairness of each state. The states meeting each requirement for fairness are shown in figures 3, 4, and 5.

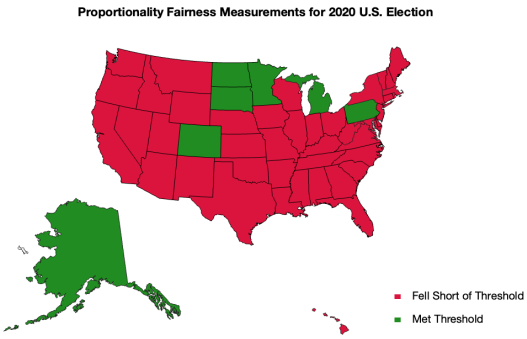

Fig. 3. Proportionality across U.S. States

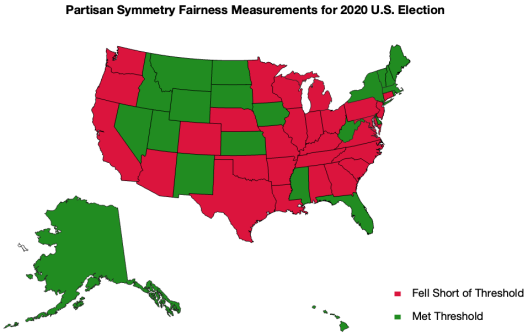

Fig. 4. Partisan Symmetry across U.S. States

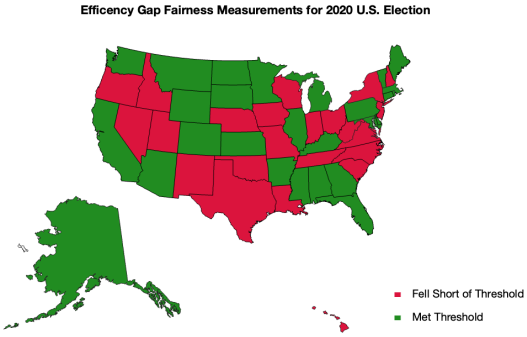

Fig. 5. Efficiency Gap across U.S. States

It is interesting to note how the fairness test with the lowest pass rate is proportionality, with only Colorado, North Dakota, South Dakota, Minnesota, Michigan, Alaska, and

Pennsylvania passing. As the the tests continue to Partisan Symmetry and Efficiency Gap, we notice that more states begin to meet required thresholds. We hypothesized this to occur due to a variety of reasons ranging from geographic voter distribution to the mechanics of the math itself. For example, the natural concentration of Democratic voters in urban areas may result in inadvertent gerrymandering that proportionality won't be able to overlook. Proportionality is also much harder to mathematically achieve because there is a strict standard for what is considered proportional and what isn't, whereas partisan symmetry and efficiency gap measurements prioritize electoral responsiveness, emphasizing relative advantage over exact proportionality.

It was hard to identify what states were more "fair" than others given three separate maps, so we wanted to consolidate these into a single map that would demonstrate the quantity and type of fairness tests that each state passed. As discussed in the methods section, we modified our code to create Figure 6, which shows an overall test fairness summary for each state, where the color represents which fairness metrics that state met the threshold for, if any.

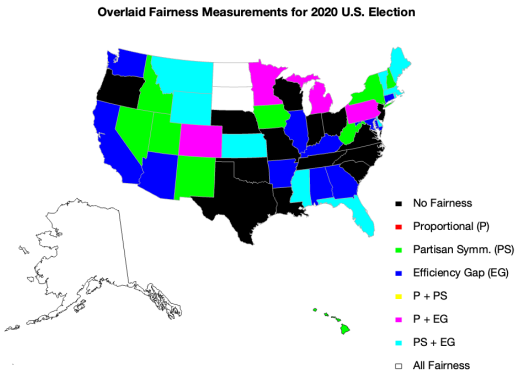

Fig. 6. All Fairness Metrics

This figure demonstrated that no states fell into the category of both proportional and partition symmetry fairness if they didn't also have efficiency gap fairness. The only states that did meet all three fairness metrics – Alaska, North Dakota, and South Dakota – actually do not need to draw district maps because they only have one seat in congress, meaning that the whole state votes on on the same candidates. This was interesting because it almost uncovered an exclusivity between partition symmetry and proportionality except for the case when there was 1 seat being voted for. This also means that no state that draws district maps (for the purpose of electing members of the House of Representatives) satisfies all three fairness metrics. It is also important to note that Delaware, Vermont, and Wyoming also only have one seat in the House of Representatives, but these states do not satisfy proportionality. This again indicates a separation between certain fairness metrics, but doesn't necessarily imply that gerrymandering occurred. In the case of Delaware, Vermont, and Wyoming, the fairness outcome may be a result of the population distribution throughout the state, requiring further analysis to determine the true cause for such fairness outcomes.

We wanted to see if we could somehow further extrapolate our findings. Figure 7 shows a table that indicates what fraction of each type of state (Republican, Democrat, or swing) meets each type of fairness threshold. Note that the rows do not add up to 1 since some states satisfy multiple fairness metrics.

	P	PS	EG	None
Republican	0.13	0.46	0.46	0.46
Democrat	0.11	0.47	0.58	0.16
Swing	0.29	0.14	0.57	0.14

Fig. 7. Fairness Metrics by Type of State

We chose to define an unfair state as a state that does not meet at least 1 of the three fairness metrics. Because each metric defines a different type of fairness, fair vs. unfair can not depend on just on of the metrics. Based on our findings, Republican states are more unfair than any other type of state, with 46% of Republican states not meeting any of the fairness metrics.

While gerrymandering is a possible cause for why district maps for Republican states are more likely to be considered unfair, another possible explanation could be population density. Republican states are typically made up of rural areas, causing the population to be less dense. This can make it challenging to keep communities of interest together since there may not be enough of one community in a specific region to create a majority. Because creating condense districts is also a priority for drawing district plans, if a there is not enough people belonging to one community of interest living close enough together, then it may be impossible to construct a district map where this community of interest is the majority within the district. If there are no majority-minority districts, then minority votes can be diluted by the majority vote.

Proportionality is often the most intuitive fairness metric, but as indicated by Delaware, Vermont, and Wyoming, it can also be the most misleading. If a state is very right or left leaning, it is more likely to not be proportional since there is either not enough of the minority party to be the majority of its district, or the minority party population is too spread out to form a district that mostly consists of the minority party. Delaware and Vermont are very left-leaning states, and Wyoming is a very right-leaning state, so this outcome is consistent with the proposed idea. Additionally, Massachusetts's district maps are usually not proportional, even if no gerrymandering occurs. This is because, as discussed in ESE 3090 Lecture, the Republican population is too spread out in order to form a district that has a Republican-majority. Because of this, Massachusetts typically has all Democratic representatives, even though 100% of the state is not Democratic. This begs the question: which fairness measurement is the most valid?

B. Is "Fairness" more complicated than these tests?

In the case of Massachusetts, it is arguably impossible, given the population density, to draw districts such that they will be considered proportionally "fair". However, that doesn't mean the overall districting and electoral system is unfair, given that it satisfies two of the three measurements in a way that prioritizes electoral responsiveness rather than exact proportionality, as aforementioned.

Given our other findings that proportionality is almost exclusive with partisan symmetry, and that there are no states where proportionality is the only achieved fairness metric, we considered that maybe proportionality isn't the best indicator for measuring the fairness of redistricting. It seemed not just challenging, but almost infeasible of implementing in conjuction with other fairness measurements. After some additional reflection, however, we concluded that this reasoning wasn't accurate: a very small quantity of states achieved proportionality in addition to other fairness metrics because it is so challenging to preserve. Proportionality determines fairness from the perspective of the voters, i.e. there is a 1:1 representation between voters and elected officials. Partisan symmetry and efficiency gap, on the other hand, determine fairness from the perspective of party power and "efficiency", i.e. they only care about how much power the winning party has or just that the winning party has power – they are not as individualized as proportionality. Therefore, proportionality in a way became evident as the most valid fairness measurement because it attempts to directly map percentage of voter categorization into house representation rather than just prioritizing the winning party getting into office/ holding the same power the same way its opposing party would.

Here, we identified that determining the "fairness" of these states is actually a little more complicated than we previously anticipated. For the sake of this project, we weighted each of the fairness metrics – proportionality, partisan symmetry, and efficiency gap - equally in determining the general fairness of each state. However, it may be interesting to vary the weights of different fairness measurements when creating an overall fairness score depending on which best aligns with our definition of "fairness". For example, if we believe that preserving proportionality is more important than a low efficiency gap, maintaining proportionality would benefit a state's overall fairness score more than preserving an efficiency gap fairness metric. To maintain a focused scope of our project, we decided not to pursue creating this new system in our report.

IV. Conclusion

By analyzing three fairness metrics for all U.S. States, we have concluded that Republican states are the most likely to be unfair, where 'unfair' means none of the three fairness thresholds were met. This indicates that many Republican states may have been gerrymandered, but this is not guaranteed since many factors, including ones that map drawers cannot control, affect whether a map is fair or not. This study can be continued by using other methods to determine if gerrymandering occurred. Specifically, a larger sample of district maps for each of the unfair states could be observed, and if the samples often meet at least one of the fairness metrics, then gerrymandering likely occurred. If the sample maps also do not meet any of the three fairness metrics, then the state is likely set up in a way that makes redistricting extra challenging. However, because there are millions of possibilities for how district maps can be drawn for a single state, choosing a 'good' subset of maps to look at is also challenging. The maps cannot be randomly chosen since most district maps are infeasible or illogical to implement. Using a computer program, such as the ReCom algorithm, can help find a 'good' subset of maps to analyze and help determine if gerrymandering occurred when making the 2020 election state's district map. Overall, it is very difficult to determine whether a district map is fair or not, but this distinction must be made in order to ensure that voters are properly represented by their elected officials.

References

[1] Velasquez, Manuel, and Claire Andre. "Justice and Fairness." Markkula Center for Applied Ethics, Santa Clara University, www.scu.edu/ethics/ethics-resources/ethical-decision-making/justice-and-fairness/. Accessed 11 Dec. 2024.

[2] "Voting Rights Act (1965)." National Archives and Records Administration, National Archives and Records Administration, www.archives.gov/milestone-documents/voting-rights-act. Accessed 11 Dec. 2024.

[3] Duchin, Moon, and Olivia Walch. "Measuring Partisan Fairness." Political Geometry, Birkhäuser, https://mggg.org/gerrybook.html. Accessed 11 Dec. 2024.

[4] "Election Results and Voting Information." Federal Election Commission, www.fec.gov/introduction-campaign-finance/election-results-and-voting-information/. Accessed 11 Dec. 2024.

[5] "Pennsylvania 2020 House Election Results." CNN, Cable News Network, www.cnn.com/election/2020/results/state/pennsylvania/house/district-1. Accessed 11 Dec. 2024.

[6] Nicholas O. Stephanopoulos and Eric M. McGhee. Partisan gerrymandering and the efficiency gap. University of Chicago Law Review, 82:831, 2015.

V. Detailed Calculations per State

```
%data from https://www.fec.gov/introduction-campaign-finance/election-results-and-voting-information/
opts = detectImportOptions('houseElections2020.csv');
all_results = readtable('houseElections2020.csv', opts);
%% Proportional
clc;
state = input('Enter state name (with the first letter capitalized): ' ,"s");
rep_votes = 0;
dem_votes = 0;
state_results = all_results(string(all_results{:,2})==state,:);

if state == "Minnesota"
    state_results.PARTY(strcmp(string(state_results.PARTY), 'DFL'))= {'D'};
end

num_rep_winners = 0;
num_dem_winners = 0;

for row = 1:size(state_results,1)
    if string(state_results{row,10})=='R' && str2double(string(state_results{row,15}))>0
        rep_votes = rep_votes+str2double(string(state_results{row,15}));
        if string(state_results{row,22})=='W'
            num_rep_winners = num_rep_winners+1;
        end
    elseif string(state_results{row,10})=='D' && str2double(string(state_results{row,15}))>0
        dem_votes = dem_votes+str2double(string(state_results{row,15}));
        if string(state_results{row,22})=='W'
            num_dem_winners = num_dem_winners+1;
        end
    end
end

total_votes = rep_votes+dem_votes;
total_representatives = num_rep_winners + num_dem_winners;

rep_percent = rep_votes/total_votes*100;
dem_percent = dem_votes/total_votes*100;

rep_elec_percent = num_rep_winners/total_representatives*100;
dem_elec_percent = num_dem_winners/total_representatives*100;

fprintf('---------------\n');
fprintf('PROPORTIONALITY\n');
repVotePercentString = 'Percent of votes for Republican candidates: %.2f%%\n';
repWinPercentString = 'Percent of elected representatives that are Republican: %.2f%%\n';
demVotePercentString = 'Percent of votes for Democratic candidates: %.2f%%\n';
demWinPercentString = 'Percent of elected representatives that are Democrats: %.2f%%\n';
notProportional = 'This districting map is not proportional.\n';
proportional = 'This districting map is proportional.\n';

fprintf(repVotePercentString,rep_percent)
fprintf(demVotePercentString,dem_percent)
fprintf(repWinPercentString,rep_elec_percent)
fprintf(demWinPercentString,dem_elec_percent)

if abs(rep_elec_percent-rep_percent) < 5
    fprintf(proportional)
elseif isnan(rep_percent) || isnan(rep_elec_percent)
    disp('neither')
else
    fprintf(notProportional)
end
fprintf('---------------\n');

%% Partisan Symmetry

num_districts = max(state_results{:,3});
different_index = 0;
if num_districts == 0
    num_districts = 1;
    different_index = 1;
end
rep_votes_by_district = [];
```

```
dem_votes_by_district = [];
rep_winners_by_delta_no_repeats = [];
rep_vote_share_no_repeats = [];
num_rep_winners = 0;

for i = 1:num_districts
    district_index = i;
    if different_index == 1
        district_index = i-1;
    end
    district_i = state_results(state_results.DISTRICT == district_index,:);
    district_i_rep_cand = district_i(strcmp(district_i.PARTY,'R'),:);
    district_i_rep_cand = district_i_rep_cand(str2double(district_i_rep_cand.GENERALVOTES) > 0,:);
    district_i_rep_votes = sum(str2double(district_i_rep_cand.GENERALVOTES));

    district_i_dem_cand = district_i(strcmp(district_i.PARTY,'D'),:);
    district_i_dem_cand = district_i_dem_cand(str2double(district_i_dem_cand.GENERALVOTES) > 0,:);
    district_i_dem_votes = sum(str2double(district_i_dem_cand.GENERALVOTES));

    rep_votes_by_district(end+1) = district_i_rep_votes/(district_i_rep_votes+district_i_dem_votes);
    dem_votes_by_district(end+1) = district_i_dem_votes/(district_i_rep_votes+district_i_dem_votes);

    if rep_votes_by_district(i) > 0.5
        num_rep_winners = num_rep_winners+1;
    end
end

model_num_rep_winners = 0;
rep_winners_by_delta = [];
rep_vote_share = [];
last_value = 0;
for i = 0:0.01:1
    delta = i - rep_percent/100;
    model_rep_votes_by_district = rep_votes_by_district+delta;
    model_num_rep_winners = 0;
    for d = 1:num_districts
        if model_rep_votes_by_district(d) > 0.5
            model_num_rep_winners = model_num_rep_winners+1;
        end
    end
    rep_winners_by_delta(end+1) = model_num_rep_winners/num_districts;
    rep_vote_share(end+1) = i;
    rep_winners_by_delta_no_repeats(end+1) = model_num_rep_winners/num_districts;
    rep_vote_share_no_repeats(end+1) = i;
    if last_value ~= rep_winners_by_delta(end)
        rep_winners_by_delta(end+1) = model_num_rep_winners/num_districts;
        rep_winners_by_delta(end-1) = last_value;
        rep_vote_share(end+1) = i;
    end
    last_value = rep_winners_by_delta(end);
end
rep_winners_by_delta = round(rep_winners_by_delta,2);
rep_winners_by_delta_no_repeats = round(rep_winners_by_delta_no_repeats,2);
rep_vote_share = round(rep_vote_share,2);
rep_vote_share_no_repeats = round(rep_vote_share_no_repeats,2);
oneMV = round(1-rep_winners_by_delta,2);
oneMV_no_repeats = round(1-rep_winners_by_delta_no_repeats,2);
S_1mV = [];
S_1mV_no_repeats = [];
for i = 1:length(oneMV)
    S_1mV_index = find(rep_vote_share==oneMV(i));
    S_1mV(end+1) = 1-rep_winners_by_delta(S_1mV_index(1));
end

for i = 1:length(oneMV_no_repeats)
    S_1mV_index_no_repeats = find(rep_vote_share_no_repeats==oneMV_no_repeats(i));
    S_1mV_no_repeats(end+1) = 1-rep_winners_by_delta_no_repeats(S_1mV_index_no_repeats(1));
end


sym_score = 0.5*sum(abs(rep_winners_by_delta_no_repeats-S_1mV_no_repeats)*0.01);

figure();
plot(rep_vote_share, S_1mV, '-', LineWidth=2);
```

```
xlabel('Statewide Republican Vote Share')
ylabel('Republican Seat Share')
title([num2str(state), ' Seats-Votes Curve'])
hold on;
plot(rep_vote_share, rep_winners_by_delta, '-', LineWidth=2);
fill([rep_vote_share, fliplr(rep_vote_share)], ...
     [S_1mV, fliplr(rep_winners_by_delta)], 'k', 'FaceAlpha', 0.3, 'EdgeColor', 'none');
hold off;
fontsize(15, 'pixels');
legend('Switched Parties', 'True Curve');

fprintf('PARTISAN SYMMETRY\n');

sym_score_string = 'Partisan Symmetry score: %.2f\n';
fprintf(sym_score_string,sym_score)

symmetric = 'This districting map is symmetrical.\n';
asymmetric = 'This districting map is not symmetrical.\n';
if sym_score < 0.03
    fprintf(symmetric)
else
    fprintf(asymmetric)
end
fprintf('---------------\n');

%% Efficiency Gap

fprintf('EFFICICENCY GAP\n');
num_districts = max(state_results{:,3});
rep_votes_by_district = [];
dem_votes_by_district = [];
rep_winners_by_delta_no_repeats = [];
rep_vote_share_no_repeats = [];
num_rep_winners = 0;

r_wasted = 0;
d_wasted = 0;

for i = 1:num_districts
    district_index = i;
    if different_index == 1
        district_index = i-1;
    end
    district_i = state_results(state_results.DISTRICT == district_index,:);
    district_i_rep_cand = district_i(strcmp(district_i.PARTY,'R'),:);
    district_i_rep_cand = district_i_rep_cand(str2double(district_i_rep_cand.GENERALVOTES) > 0,:);
    district_i_rep_votes = sum(str2double(district_i_rep_cand.GENERALVOTES));

    district_i_dem_cand = district_i(strcmp(district_i.PARTY,'D'),:);
    district_i_dem_cand = district_i_dem_cand(str2double(district_i_dem_cand.GENERALVOTES) > 0,:);
    district_i_dem_votes = sum(str2double(district_i_dem_cand.GENERALVOTES));

    rep_votes_by_district(end+1) = district_i_rep_votes/(district_i_rep_votes+district_i_dem_votes);
    dem_votes_by_district(end+1) = district_i_dem_votes/(district_i_rep_votes+district_i_dem_votes);

    total_district_i_votes = district_i_rep_votes + district_i_dem_votes;
    threshold_to_win = floor(total_district_i_votes/2)+1;

    if district_i_rep_votes > district_i_dem_votes %republican wins this district
        r_wasted = r_wasted + district_i_rep_votes - threshold_to_win;
        d_wasted = d_wasted + district_i_dem_votes;
    else %democrats win this district
        r_wasted = r_wasted + district_i_rep_votes;
        d_wasted = d_wasted + district_i_dem_votes - threshold_to_win;
    end
end

efficiency_gap = abs((r_wasted - d_wasted) / total_votes);


efficient = 'This districting map is efficient.\n';
not_efficient = 'This districting map is not efficient.\n';


```

```
numDemWasted = 'Number of wasted votes for the Democrati party: %u\n';
numRepWasted = 'Number of wasted votes for the Republican party: %u\n';


fprintf(numDemWasted,d_wasted)
fprintf(numRepWasted,r_wasted)

eff_gap = 'Efficiency gap: %.2f\n';
fprintf(eff_gap,efficiency_gap)

if efficiency_gap < 0.08
    fprintf(efficient);
else
    fprintf(not_efficient)
end


fprintf('---------------\n');
```

VI. Map Generation Code

```
%% Data Reading Input
%data from https://www.fec.gov/introduction-campaign-finance/election-results-and-voting-information/
opts = detectImportOptions('houseElections2020.csv');
all_results = readtable('houseElections2020.csv', opts);

%All states
states = readgeotable("usastatelo.shp");
states = states.Name;

proportionalFair  = zeros(numel(states), 1);
symmetricFair   = zeros(numel(states), 1);
efficientFair   = zeros(numel(states), 1);

%% All Fairness Measurements

for state = 1:numel(states)
    rep_votes = 0;
    dem_votes = 0;
    state_results = all_results(string(all_results{:,2})==states(state),:);

    if states(state) == "Minnesota"
        state_results.PARTY(strcmp(string(state_results.PARTY), 'DFL'))= {'D'};
    end

    num_rep_winners = 0;
    num_dem_winners = 0;

    for row = 1:size(state_results,1)
        if string(state_results{row,10})=='R' && str2double(string(state_results{row,15}))>0
            rep_votes = rep_votes+str2double(string(state_results{row,15}));
            if string(state_results{row,22})=='W'
                num_rep_winners = num_rep_winners+1;
            end
        elseif string(state_results{row,10})=='D' && str2double(string(state_results{row,15}))>0
            dem_votes = dem_votes+str2double(string(state_results{row,15}));
            if string(state_results{row,22})=='W'
                num_dem_winners = num_dem_winners+1;
            end
        end
    end

    total_votes = rep_votes+dem_votes;
    total_representatives = num_rep_winners + num_dem_winners;

    rep_percent = rep_votes/total_votes*100;
    dem_percent = dem_votes/total_votes*100;

    rep_elec_percent = num_rep_winners/total_representatives*100;
    dem_elec_percent = num_dem_winners/total_representatives*100;

    if abs(rep_elec_percent-rep_percent) < 5
        proportionalFair(state) = 1;
    elseif isnan(rep_percent) || isnan(rep_elec_percent)
        proportionalFair(state) = 2;
    else
        proportionalFair(state) = 0;
    end

    %% Partisan Symmetry

    num_districts = max(state_results{:,3});
    different_index = 0;
    if num_districts == 0
        num_districts = 1;
        different_index = 1;
    end
    rep_votes_by_district = [];
    dem_votes_by_district = [];
    rep_winners_by_delta_no_repeats = [];
    rep_vote_share_no_repeats = [];
    num_rep_winners = 0;

```

```
    for i = 1:num_districts
        district_index = i;
        if different_index == 1
            district_index = i-1;
        end
        district_i = state_results(state_results.DISTRICT == district_index,:);
        district_i_rep_cand = district_i(strcmp(district_i.PARTY,'R'),:);
        district_i_rep_cand = district_i_rep_cand(str2double(district_i_rep_cand.GENERALVOTES) > 0,:);
        district_i_rep_votes = sum(str2double(district_i_rep_cand.GENERALVOTES));

        district_i_dem_cand = district_i(strcmp(district_i.PARTY,'D'),:);
        district_i_dem_cand = district_i_dem_cand(str2double(district_i_dem_cand.GENERALVOTES) > 0,:);
        district_i_dem_votes = sum(str2double(district_i_dem_cand.GENERALVOTES));

        rep_votes_by_district(end+1) = district_i_rep_votes/(district_i_rep_votes+district_i_dem_votes);
        dem_votes_by_district(end+1) = district_i_dem_votes/(district_i_rep_votes+district_i_dem_votes);

        if rep_votes_by_district(i) > 0.5
            num_rep_winners = num_rep_winners+1;
        end
    end

    model_num_rep_winners = 0;
    rep_winners_by_delta = [];
    rep_vote_share = [];
    last_value = 0;
    for i = 0:0.01:1
        delta = i - rep_percent/100;
        model_rep_votes_by_district = rep_votes_by_district+delta;
        model_num_rep_winners = 0;
        for d = 1:num_districts
            if model_rep_votes_by_district(d) > 0.5
                model_num_rep_winners = model_num_rep_winners+1;
            end
        end
        rep_winners_by_delta(end+1) = model_num_rep_winners/num_districts;
        rep_vote_share(end+1) = i;
        rep_winners_by_delta_no_repeats(end+1) = model_num_rep_winners/num_districts;
        rep_vote_share_no_repeats(end+1) = i;
        if last_value ~= rep_winners_by_delta(end)
            rep_winners_by_delta(end+1) = model_num_rep_winners/num_districts;
            rep_winners_by_delta(end-1) = last_value;
            rep_vote_share(end+1) = i;
        end
        last_value = rep_winners_by_delta(end);
    end
    rep_winners_by_delta = round(rep_winners_by_delta,2);
    rep_winners_by_delta_no_repeats = round(rep_winners_by_delta_no_repeats,2);
    rep_vote_share = round(rep_vote_share,2);
    rep_vote_share_no_repeats = round(rep_vote_share_no_repeats,2);
    oneMV = round(1-rep_winners_by_delta,2);
    oneMV_no_repeats = round(1-rep_winners_by_delta_no_repeats,2);
    S_1mV = [];
    S_1mV_no_repeats = [];
    for i = 1:length(oneMV)
        S_1mV_index = find(rep_vote_share==oneMV(i));
        S_1mV(end+1) = 1-rep_winners_by_delta(S_1mV_index(1));
    end

    for i = 1:length(oneMV_no_repeats)
        S_1mV_index_no_repeats = find(rep_vote_share_no_repeats==oneMV_no_repeats(i));
        S_1mV_no_repeats(end+1) = 1-rep_winners_by_delta_no_repeats(S_1mV_index_no_repeats(1));
    end


    sym_score = 0.5*sum(abs(rep_winners_by_delta_no_repeats-S_1mV_no_repeats)*0.01);

    if sym_score < 0.03
        symmetricFair(state) = 1;
    else
        symmetricFair(state) = 0;
    end

    %% Efficiency Gap
```

```
147
148     num_districts = max(state_results{:,3});
149     rep_votes_by_district = [];
150     dem_votes_by_district = [];
151     rep_winners_by_delta_no_repeats = [];
152     rep_vote_share_no_repeats = [];
153     num_rep_winners = 0;
154
155     r_wasted = 0;
156     d_wasted = 0;
157
158     for i = 1:num_districts
159         district_index = i;
160         if different_index == 1
161             district_index = i-1;
162         end
163         district_i = state_results(state_results.DISTRICT == district_index,:);
164         district_i_rep_cand = district_i(strcmp(district_i.PARTY,'R'),:);
165         district_i_rep_cand = district_i_rep_cand(str2double(district_i_rep_cand.GENERALVOTES) > 0,:);
166         district_i_rep_votes = sum(str2double(district_i_rep_cand.GENERALVOTES));
167
168         district_i_dem_cand = district_i(strcmp(district_i.PARTY,'D'),:);
169         district_i_dem_cand = district_i_dem_cand(str2double(district_i_dem_cand.GENERALVOTES) > 0,:);
170         district_i_dem_votes = sum(str2double(district_i_dem_cand.GENERALVOTES));
171
172         rep_votes_by_district(end+1) = district_i_rep_votes/(district_i_rep_votes+district_i_dem_votes);
173         dem_votes_by_district(end+1) = district_i_dem_votes/(district_i_rep_votes+district_i_dem_votes);
174
175         total_district_i_votes = district_i_rep_votes + district_i_dem_votes;
176         threshold_to_win = floor(total_district_i_votes/2)+1;
177
178         if district_i_rep_votes > district_i_dem_votes %republican wins this district
179             r_wasted = r_wasted + district_i_rep_votes - threshold_to_win;
180             d_wasted = d_wasted + district_i_dem_votes;
181         else %democrats win this district
182             r_wasted = r_wasted + district_i_rep_votes;
183             d_wasted = d_wasted + district_i_dem_votes - threshold_to_win;
184         end
185     end
186
187     efficiency_gap = abs((r_wasted - d_wasted) / total_votes);
188
189     if efficiency_gap < 0.08
190         efficientFair(state)  = 1;
191     else
192         efficientFair(state) = 0;
193     end
194 end
195
196 %% Individual Fairness Map Drawings
197 graphNum = 0;
198 names = ["Proportionality" "Partisan Symmetry" "Efficency Gap"];
199
200 for measurement = [proportionalFair symmetricFair efficientFair]
201
202     figure()
203     graphNum = graphNum + 1;
204     colors = colorArray(measurement);
205
206     states2 = readgeotable("usastatelo.shp");
207
208     rowAlaska = states2.Name == "Alaska";
209     rowHawaii = states2.Name == "Hawaii";
210
211     statesA   = states2(rowAlaska, :);
212     statesH   = states2(rowHawaii, :);
213
214     h = height(states2);
215     faceColors = makesymbolspec("Polygon",{'INDEX',[1 h],'FaceColor',colors});
216
217     allColors = cell2mat(faceColors.FaceColor(3));
218
219     ax = usamap("all");
220
```

```
    geoshow(ax(1),states2,"DisplayType","polygon","SymbolSpec",faceColors);
    geoshow(ax(2),statesA,"FaceColor", allColors(rowAlaska,:))
    geoshow(ax(3),statesH,"FaceColor", allColors(rowHawaii,:))

    for k = 1:3
        setm(ax(k),"Frame","off","Grid","off",...
          "ParallelLabel","off","MeridianLabel","off")
    end

    completeTitle = strcat(names(graphNum), " Fairness Measurements for 2020 U.S. Election");
    ttl = title(completeTitle);

    annotation('rectangle', [0.75, 0.25, 0.01, 0.01], 'FaceColor', [0.86, 0.08, 0.24], 'EdgeColor', 'none')
    ;
    annotation('textbox', [0.77, 0.26, 0.2, 0.02], 'String', 'Fell Short of Threshold', 'EdgeColor', 'none'
    , 'FontSize', 10);

    annotation('rectangle', [0.75, 0.2, 0.01, 0.01], 'FaceColor', [0.13, 0.55, 0.13], 'EdgeColor', 'none');
    annotation('textbox', [0.77, 0.21, 0.2, 0.02], 'String', 'Met Threshold', 'EdgeColor', 'none', '
    FontSize', 10);
end

%% Aggregate Fairness Measurements
figure()

hold on
colors = [proportionalFair symmetricFair efficientFair];


    states2 = readgeotable("usastatelo.shp");

    rowAlaska = states2.Name == "Alaska";
    rowHawaii = states2.Name == "Hawaii";

    statesA   = states2(rowAlaska, :);
    statesH   = states2(rowHawaii, :);

    h = height(states2);
    faceColors = makesymbolspec("Polygon",{'INDEX',[1 h],'FaceColor',colors});

    allColors = cell2mat(faceColors.FaceColor(3));

    ax = usamap("all");

                h1 = geoshow(ax(1),states2,'DefaultEdgeColor', [.7 .7 .7], "DisplayType","polygon","
    SymbolSpec",faceColors);
    h2 = geoshow(ax(2),statesA,"FaceColor", allColors(rowAlaska,:));
    h3 = geoshow(ax(3),statesH,"FaceColor", allColors(rowHawaii,:));

    for k = 1:3
        setm(ax(k),"Frame","off","Grid","off",...
          "ParallelLabel","off","MeridianLabel","off")
    end

    ttl = title("Overlaid Fairness Measurements for 2020 U.S. Election");

% Adjusted upward placement factor
y_offset = 0.15;

% Black (0,0,0)
annotation('rectangle', [0.75, 0.30+y_offset, 0.01, 0.01], 'FaceColor', [0 0 0], 'EdgeColor', 'none');
annotation('textbox', [0.77, 0.31+y_offset, 0.2, 0.02], 'String', 'No Fairness', 'EdgeColor', 'none', '
    FontSize', 10);

% Red (1,0,0)
annotation('rectangle', [0.75, 0.25+y_offset, 0.01, 0.01], 'FaceColor', [1 0 0], 'EdgeColor', 'none');
annotation('textbox', [0.77, 0.26+y_offset, 0.2, 0.02], 'String', 'Proportional (P)', 'EdgeColor', 'none',
    'FontSize', 10);

% Green (0,1,0)
annotation('rectangle', [0.75, 0.20+y_offset, 0.01, 0.01], 'FaceColor', [0 1 0], 'EdgeColor', 'none');
annotation('textbox', [0.77, 0.21+y_offset, 0.2, 0.02], 'String', 'Partisan Symm. (PS)', 'EdgeColor', 'none
    ', 'FontSize', 10);

```

```
% Blue (0,0,1)
annotation('rectangle', [0.75, 0.15+y_offset, 0.01, 0.01], 'FaceColor', [0 0 1], 'EdgeColor', 'none');
annotation('textbox', [0.77, 0.16+y_offset, 0.2, 0.02], 'String', 'Efficiency Gap (EG)', 'EdgeColor', 'none', 'FontSize', 10);

% Yellow (1,1,0)
annotation('rectangle', [0.75, 0.10+y_offset, 0.01, 0.01], 'FaceColor', [1 1 0], 'EdgeColor', 'none');
annotation('textbox', [0.77, 0.11+y_offset, 0.2, 0.02], 'String', 'P + PS', 'EdgeColor', 'none', 'FontSize', 10);

% Magenta (1,0,1)
annotation('rectangle', [0.75, 0.05+y_offset, 0.01, 0.01], 'FaceColor', [1 0 1], 'EdgeColor', 'none');
annotation('textbox', [0.77, 0.06+y_offset, 0.2, 0.02], 'String', 'P + EG', 'EdgeColor', 'none', 'FontSize', 10);

% Cyan (0,1,1)
annotation('rectangle', [0.75, 0.0+y_offset, 0.01, 0.01], 'FaceColor', [0 1 1], 'EdgeColor', 'none');
annotation('textbox', [0.77, 0.01+y_offset, 0.2, 0.02], 'String', 'PS + EG', 'EdgeColor', 'none', 'FontSize', 10);

% White (1,1,1)
annotation('rectangle', [0.75, -0.05+y_offset, 0.01, 0.01], 'FaceColor', [1 1 1], 'EdgeColor', 'black');
annotation('textbox', [0.77, -0.04+y_offset, 0.2, 0.02], 'String', 'All Fairness', 'EdgeColor', 'none', 'FontSize', 10);

hold off

%% Functions
function colorArray = colorArray(binaryArray)

    % Define color values
    crimson = [0.86, 0.08, 0.24]; % RGB for crimson
    forestGreen = [0.13, 0.55, 0.13]; % RGB for forest green
    black = [0, 0, 0];

    % Preallocate the color array
    colorArray = zeros(length(binaryArray), 3);

    % Assign colors based on binaryArray values
    for i = 1:numel(binaryArray)
        if binaryArray(i) == 0
            colorArray(i, :) = crimson;
        elseif binaryArray(i) == 1
            colorArray(i, :) = forestGreen;
        else
            colorArray(i, :) = black;
        end
    end

end
```

VII. Web Scraping Exploration Code

```python
# scraper-python.py
# To run this script, paste `python scraper-python.py` in the terminal

# import requests
# from bs4 import BeautifulSoup


# def scrape():
#     url = 'https://www.cnn.com/election/2020/results/state/pennsylvania/house/district-1'
#     response = requests.get(url)
#     soup = BeautifulSoup(response.text, 'html.parser')

#     title = soup.select_one('h1').text
#     text = soup.select_one('p').text
#     link = soup.select_one('a').get('href')
#     print(title)
#     print(text)
#     print(link)

# if __name__ == '__main__':
#     scrape()

# scraper-python.py
# To run this script, paste `python scraper-python.py` in the terminal

# hover = driver.find_element(By.CLASS_NAME, 'house-bubblesstyles__OuterHouseCircle-sc-1kq3cgp-3 ijHLKF')
    # actions = ActionChains(driver)
    # actions.move_to_element(hover).perform()
    # time.sleep(3)

    # for choice in options:
    #     ######
    #     break
    # actions = ActionChains(driver)
    # actions.move_to_element(hover).perform()

from selenium import webdriver
from selenium.webdriver import ActionChains
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import Select
import time

def scrape():
    # Path to your WebDriver executable (e.g., chromedriver for Chrome)
    driver = webdriver.Chrome()
    driver.maximize_window
    driver.get('https://www.cnn.com/election/2020/results/state/pennsylvania/house/district-1')

    #prints all states
    counties_list = driver.find_element(By.CLASS_NAME, "hkzbOY")
    print(counties_list.text)

    dropdown = Select(counties_list)

    for option in dropdown.options:
        print(dropdown.options)
        print(f"Selecting option: {option}")
        dropdown.select_by_visible_text(option)
        time.sleep(1)  # Optional: Wait for visual confirmation or page response

    # #hovers over 1 bubble
    # hover = driver.find_element(By.CLASS_NAME, 'dropdown-selectstyles__Select-ut7hny-0')
    # options = hover.text.count('\n')
    # print(options)
    # hover.click()
    # time.sleep(1)
    # print(hover.text)



    # hi = driver.find_elements(By.CSS_SELECTOR, f"option[value]")
    # for h in hi:
```

```
    #     print(h.text)

    for option in range(17):
        hello = driver.find_element(By.CSS_SELECTOR, f"//option[value={option+2})]")
        hello.click()

    # driver.quit()

if __name__ == '__main__':
    scrape()
```