RAHUL KAYITHI, ALPHAN YANG, HOANGCAMT NGUYEN

Redistricting and Gerrymandering NJ Congressional map

Fair Map:

In creating our redistricting algorithm for creating a fair map for New Jersey's congressional districts, we tried to incorporate multiple techniques to address the complex requirements of redistricting. After trying many different algorithm types, we settled on a flip-step approach .The flip-step algorithm is great for redistricting because it makes small changes, one neighborhood at a time. This means we can carefully adjust the map to make sure it's fair, without making big mistakes. One downside of the flip-step algorithm is that it can be time-consuming, as it adjusts districts bit by bit, which might not be efficient for large-scale redistricting tasks. Additionally, because it makes changes incrementally, there's a risk of getting stuck in a local optimum, where the algorithm makes a district slightly better but misses out on a much better overall configuration that could be achieved with more significant changes. We found this out the hard way, as many hours were eaten up by running the algorithms, trying to tweak it, running it again, etc. Alphan had an algorithm running, went to sleep, and then woke up to the code still executing!

We tried employing geometric calculations, such as the Polsby-Popper measure, to assess the compactness of proposed districts, ensuring that they are as regular in shape as possible. This metric is particularly important as it reflects the intuitive notion of what constitutes a "fair" shape for a district. However, our algorithm revealed challenges in balancing compactness with the other critical factors of redistricting, such as political balance and population equality. Despite our efforts, the compactness scores

indicated room for improvement, suggesting that our methodology for aggregating precincts into districts may need to be refined to create more compact districts without compromising other redistricting principles. Ultimately, we encountered too many challenges with incorporating Polsby-Popper, so we had to move on to the other aspects of our algorithm.

The main two fields we had some success with (very liberal use of the word "some") was in population equality and political leaning. We had functions dedicated to calculating district populations and ensuring they fall within an acceptable deviation range.

```python
def good_pop_change(district_populations, target_district, data_file, current_precinct):
    total_population = sum(district_populations.values())
    num_districts = len(district_populations)
    average_population = total_population / num_districts

    deviation_percentage = 10
    lower_bound = average_population * (1 - deviation_percentage / 100)
    upper_bound = average_population * (1 + deviation_percentage / 100)

    precinct_total_population = data_file[data_file['GEOID20'] == current_precinct]['Total_2020_Total'].values[0]

    # Update population for target district with the precinct's population
    new_district_population = district_populations[target_district] + precinct_total_population

    # Check if new district population is within the deviation range
    if not (lower_bound <= new_district_population <= upper_bound):
        return False

    # Check if the overall population distribution across all districts is improved or at least not worsened
    for district, population in district_populations.items():
        if district == target_district:
            # Check with the updated population for the target district
            current_deviation = abs(new_district_population - average_population)
        else:
            # Check with the existing population for other districts
            current_deviation = abs(population - average_population)

        # If any district falls outside the deviation range and is not improved, return False
        if current_deviation > abs(population - average_population) and not (lower_bound <= population <= upper_bound):
            return False

    # If all checks pass, the population change is considered good
    return True
```

```python
def calculate_district_populations(assignments, precinct_data):
    populations = {}
    for district_id in assignments['District'].unique():
        district_precincts = assignments[assignments['District'] == district_id]['GEOID20']
        populations[district_id] = precinct_data[precinct_data['GEOID20'].isin(district_precincts)]['Total_2020_Total'].sum()
    return populations
```

In our algorithm, we have a function named 'calculate_district_populations' that we used for tallying

up the number of people in each district. This function goes through each district and adds up the population of all its precincts. We needed to make sure every district has about the same number of people, which is a fundamental requirement for fair representation.

Another important function for population control was ‘ good_pop_change’. This one checks to see if moving a precinct from one district to another would keep the population within an acceptable range. We have a target population for each district, and we allow for a 10% wiggle room above or below this target. The good_pop_change function makes sure that when we consider moving a precinct, it doesn't cause the district to go over or fall short of this range. This helps us maintain a balance, so no district is too big or too small compared to the others.

We integrate this with a political balance calculation that considers the distribution of party votes within districts. Our intention here is to create districts that are not only equal in population but also fair in their political representation. Nonetheless, the functionality that assesses whether a population change is beneficial could be further improved to better capture the nuances of population shifts and their implications for both balance and compactness.

```python
def calculate_political_balance(district, assignment, precinct_data):
    # Filter precinct data for the given district
    district_data = precinct_data[precinct_data['GEOID20'].isin(assignment[assignment['District'] == district]['GEOID20'])]

    # Sum up votes for Democrats, Republicans, and other parties in the district
    total_dem_votes = district_data['Dem_2020_Pres'].sum()
    total_rep_votes = district_data['Rep_2020_Pres'].sum()
    total_votes = district_data['Total_2020_Pres'].sum()

    # Calculate the balance
    if total_votes == 0:
        return 0  # To handle division by zero if there are no votes

    # Calculate the percentage for Democrats and Republicans
    dem_percentage = total_dem_votes / total_votes
    rep_percentage = total_rep_votes / total_votes

    # Calculate balance as an integer
    # Closer to 0 indicates a better balance, higher values indicate more skew
    balance = abs(dem_percentage - rep_percentage) * 100  # Scale to make it an integer

    return int(balance)
```

The function in the algorithm that addresses political balance by considering the distribution of party votes within districts is calculate_political_balance. This function is specifically designed to analyze the voting patterns within a district, summing up the votes for each political party based on recent election data. It calculates the share of votes for the major parties, allowing us to gauge how politically balanced a district is. A district with a near-equal share of votes for different parties is considered well-balanced, reflecting a competitive political environment. On the other hand, a district where one party overwhelmingly dominates the vote share is seen as less balanced.

The redistricting process also hinges on maintaining the contiguity of districts. Our algorithm includes a rigorous check for contiguity when precincts are reassigned, ensuring that districts do not become fragmented. However, the complexities of ensuring contiguity while also optimizing for political balance and compactness have proven to be a significant challenge. The iterative nature of the algorithm, which attempts to find the best precinct swaps, sometimes encounters limitations in identifying the most optimal moves, especially when multiple redistricting criteria must be satisfied

simultaneously. Enhancing the algorithm's efficiency in exploring the vast solution space and improving its heuristics for precinct selection will be essential steps in refining our redistricting approach.

Results:

While redistricting, we tried to construct a fair representation of New Jersey's congressional districts, aiming to balance the components of demographic representation, political competitiveness, and compliance with legal redistricting principles. Our "fair map" demonstrates our effort towards achieving proportional representation, displayed by a marginally improved disproportionality score of -15.53% compared to the actual 2022 map's -16.24%, suggesting a closer alignment with the state's partisan divide. Despite this, our map indicates a bias against Republicans, a point we must scrutinize further. Additionally, we successfully enhanced electoral competitiveness, with our score of 28.95% outperforming the actual map's 21.24%, reflecting our commitment to fostering districts where elections are genuinely contestable, and thus, promoting a more vibrant democratic process.

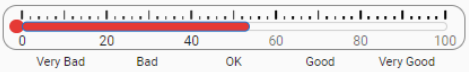
Our Map:

Proportionality

All else equal, prefer maps that are more proportional.

Metric		Description
• Disproportionality	-15.53%	The deviation from the number of whole seats closest to proportional. Smaller is better. By convention, positive values of bias metrics favor Republicans & negative val

Rating



Notes

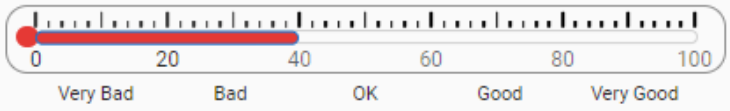
- The average map-wide Democratic two-party vote share is 56.04%, the Republican 43.96%.
- The number of Democratic seats closest to proportional is seven. The likely number of Democratic seats is 8.86. The likely number of unexpected Democratic seats (won) lost is -1.86.

Competitiveness

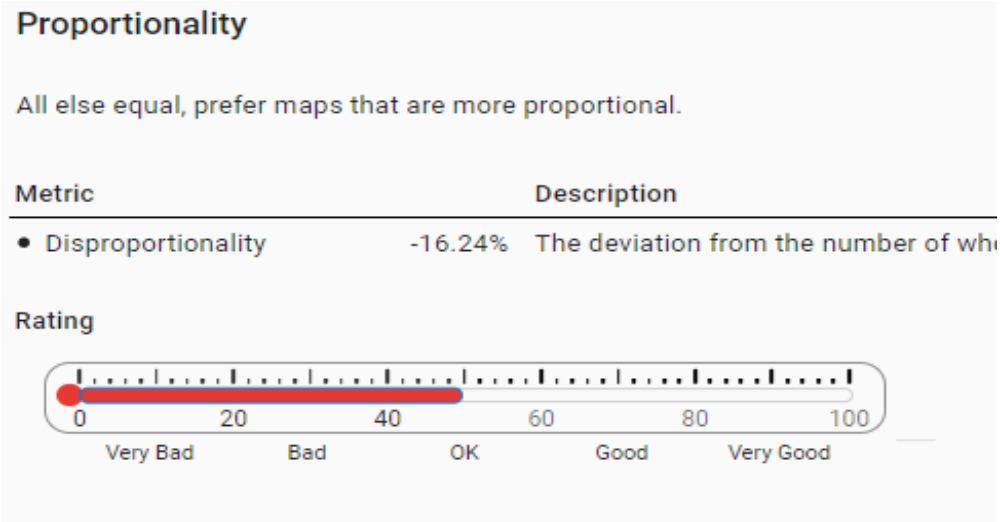
All else equal, prefer maps that are more competitive.

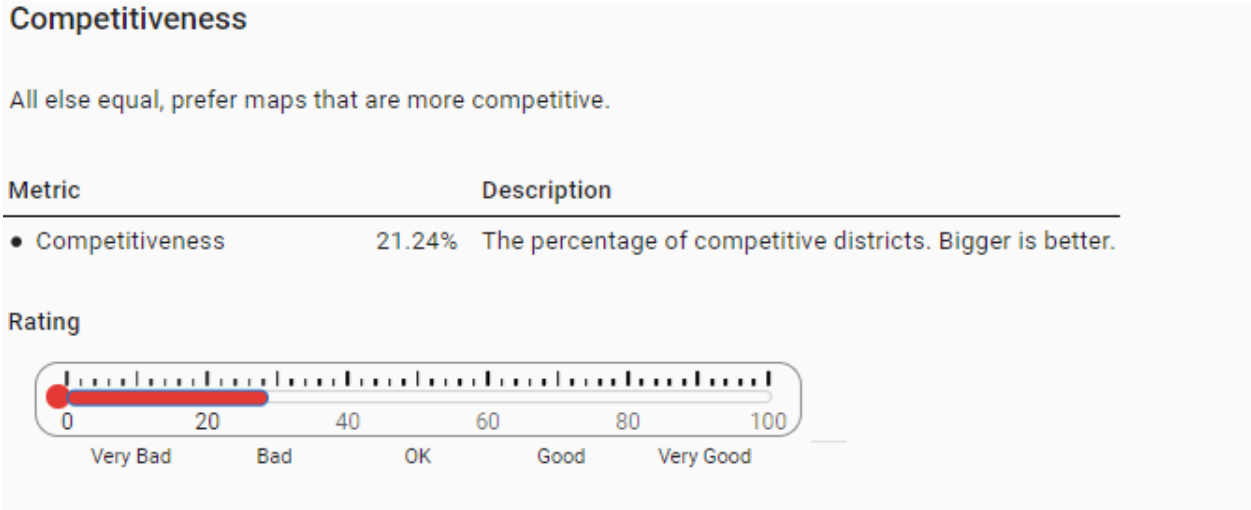
Metric		Description
• Competitiveness	28.95%	The percentage of competitive districts. Bigger is better.

Rating



Actual Assignment:

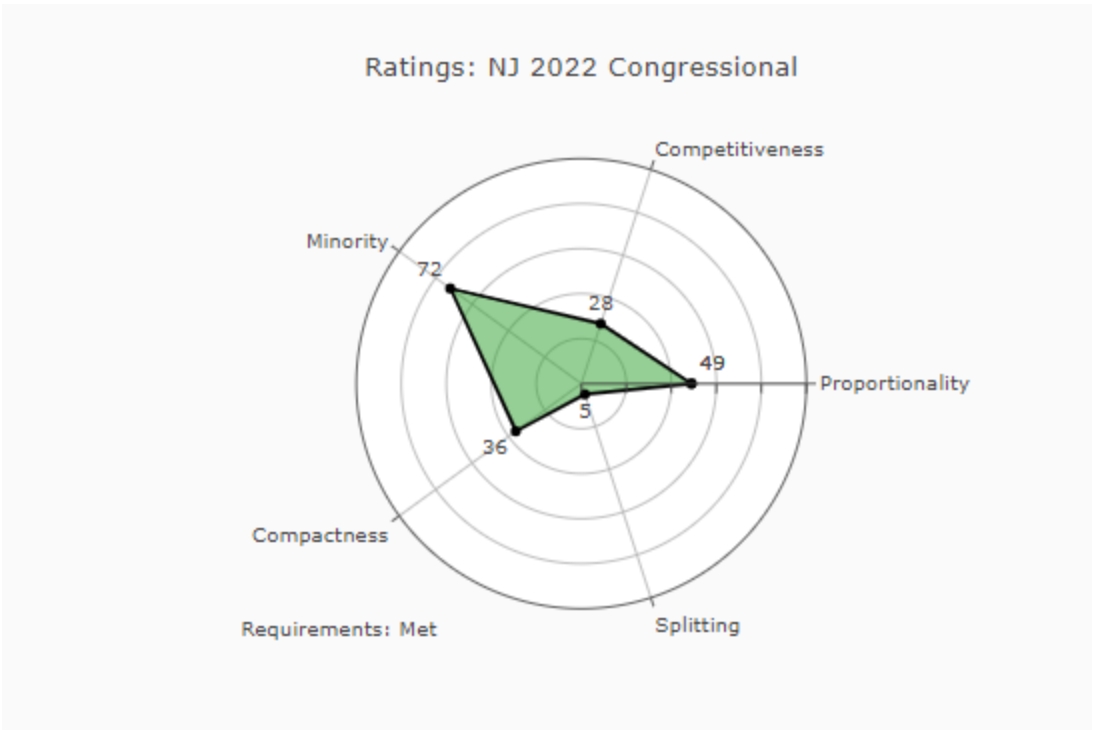


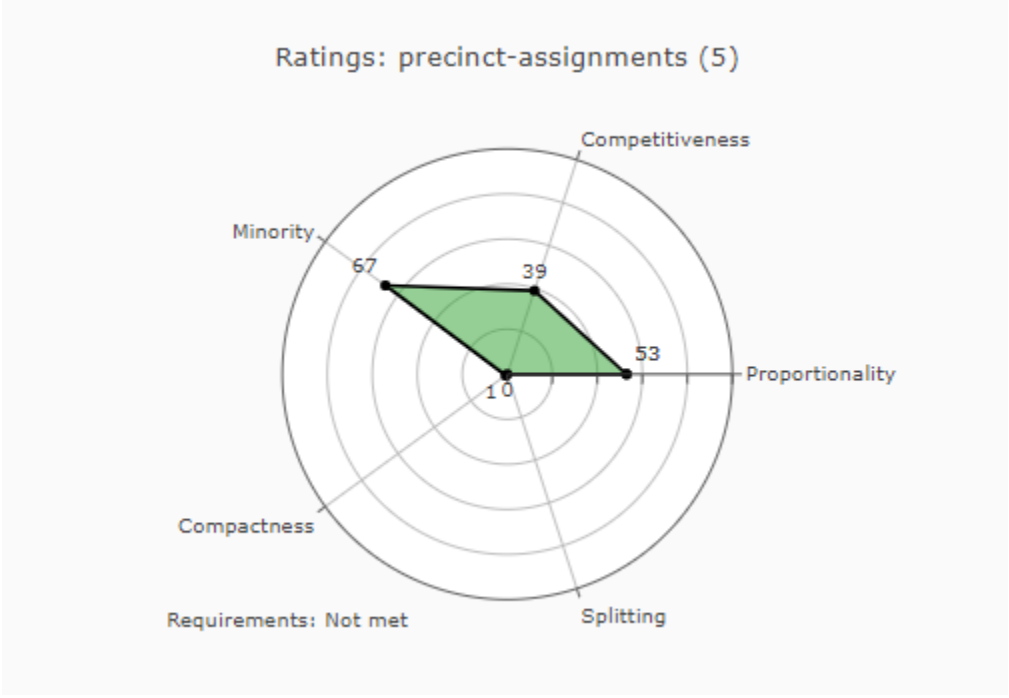


However, our approach to creating fair districts unveiled certain trade-offs, especially in the realms of community integrity and geographic compactness. The radar chart's low compactness score and a splitting rating of 10 highlight these shortcomings, implying that our map may have inadvertently divided communities and constructed less geographically sensible districts. While our intentions were to generate a balanced and equitable map, these metrics suggest that in our pursuit to

address partisan fairness and competitiveness, we might have overlooked the importance of preserving the unity of communities of interest and the traditional redistricting principle of creating compact districts.

Reflecting on minority representation, our fair map scored slightly lower (67) compared to the actual map (72), signaling room for improvement in creating opportunity districts for minority populations. Although we strived to ensure that minorities have fair opportunities to elect representatives of their choice, it is clear that we need to enhance our approach in this area. Our focus on maintaining a delicate balance between various redistricting principles may have led to these less-than-optimal outcomes for minority representation. Moving forward, we recognize the need to fine-tune our redistricting algorithm to better uphold the values of community coherence, compactness, and equal representation for minority groups, without compromising on the overall fairness of the electoral map.


Ratings: NJ 2022 Congressional
Competitiveness
Minority
72
28
49
Proportionality
5
36
Compactness
Requirements: Met
Splitting


Ratings: precinct-assignments (5)
Competitiveness
Minority
67
39
53
Proportionality
1 0
Compactness
Requirements: Not met
Splitting

Unfair Map:

Our 'unfair' map's goal was to gerrymander the NJ congressional districts with a bias towards minority representation and a decreased efficiency gap for political leanings which reduced the impact of Republican voters. Upon analysis, it's evident that our map only subtly shifts the political balance without crossing the thresholds that typically signal gerrymandering (basically, the map is still a little too "fair" for our preference).

We utilized another variation of the flip-step to design our algorithm for the unfair map. Like the previous algorithm, one of the main challenges was the time required to run the algorithm, with some iterations taking over an hour, but the code was largely trimmed down and eventually condensed into a more manageable program that took around 10 minutes of runtime.

In the creation of our 'unfair' map, our algorithm took a strategic approach aimed at reshaping district lines to benefit the Democratic party, which already has a majority presence in the state with 55.07% of voters, while Republicans have 43.19%. Through the calculate_political_leaning function, we meticulously analyzed the political composition of each district, ensuring that the precincts we considered flipping would augment the voting power of our target demographic. This function became the backbone of our algorithm, providing the insights necessary to make data-driven decisions that could subtly but effectively alter the political landscape.

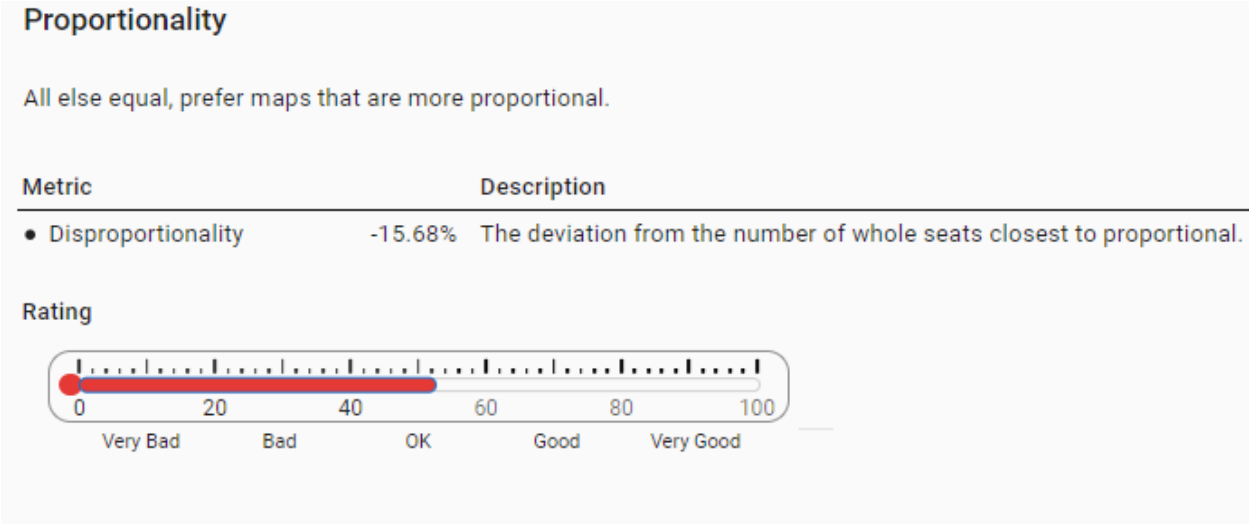
```python
# Function to calculate the political leaning of a district
def calculate_political_leaning(district_num, assignment, data_file):
    district_voters = data_file[data_file['GEOID20'].isin(assignment[assignment['District'] == district_num]['GEOID20'])]
    dem_voters = district_voters['Dem_2020_Pres'].sum()
    rep_voters = district_voters['Rep_2020_Pres'].sum()
    return dem_voters, rep_voters
```

The flip_border_precincts function is where our algorithm's strategic intent comes into play. By identifying and flipping precincts on the border of two districts, we aimed to augment the voter base of our preferred party in strategic districts. We implemented demographic constraints, such as the maximum white population percentage, to maintain the veneer of fairness. However, this demographic consideration, while serving as a tool for shaping political outcomes, introduced complexity to our algorithm. Ensuring these flips did not violate contiguity or demographic thresholds proved challenging, as each flip had to be assessed for multiple criteria, sometimes resulting in the reversal of the flip if it did not meet our strategic goals.

```
# Function to flip border precincts with political and demographic considerations
def flip_border_precincts(district_from, district_to, max_white_percentage):
    border_precincts = []
    district_list = nj_flipstep_assignment[nj_flipstep_assignment['District'] == district_from]['GEOID20']

    for precinct in district_list:
        neighbors = ast.literal_eval(nj_contiguity[nj_contiguity['Precinct'] == precinct]['Neighbors'].values.tolist()[0])
        for neighbor in neighbors:
            if nj_flipstep_assignment.loc[nj_flipstep_assignment['GEOID20'] == neighbor, 'District'].values[0] == district_to:
                border_precincts.append(precinct)
                break

    # Randomly sample precincts and flip them unless it breaks contiguity or white population constraint
    flipped_precincts = np.random.choice(border_precincts, min(10, len(border_precincts)), replace=False)
    for flip in flipped_precincts:
        # Temporarily flip the precinct
        nj_flipstep_assignment.loc[nj_flipstep_assignment['GEOID20'] == flip, 'District'] = district_to

        # Check contiguity
        if not isDistrictContiguous(district_from, nj_flipstep_assignment, nj_contiguity) or \
           not isDistrictContiguous(district_to, nj_flipstep_assignment, nj_contiguity):
            nj_flipstep_assignment.loc[nj_flipstep_assignment['GEOID20'] == flip, 'District'] = district_from
            print("Contiguity broken by flipping precinct " + flip)
            continue

        # Check white population percentage
        district_population = getDistrictPopulations(nj_flipstep_assignment, nj_precinct_data, 12)[district_to]
        district_white_population = nj_precinct_data[nj_precinct_data['GEOID20'].isin(nj_flipstep_assignment[nj_flipstep_assignment['District'] == district_to]['GEOID20'])]['White_2020_Total'].sum()
        if (district_white_population / district_population) > max_white_percentage:
            nj_flipstep_assignment.loc[nj_flipstep_assignment['GEOID20'] == flip, 'District'] = district_from
            print("White population percentage exceeded by flipping precinct " + flip)

        # Additional check to favor Democrats
        dem_voters, rep_voters = calculate_political_leaning(district_to, nj_flipstep_assignment, nj_precinct_data)
        if dem_voters < rep_voters:
            nj_flipstep_assignment.loc[nj_flipstep_assignment['GEOID20'] == flip, 'District'] = district_from
            print("Flipping precinct " + flip + " does not favor Democrats. Reverting.")
```

Our algorithm's design also included a nuanced focus on minority representation, guided by the flip_border_precincts function, which was informed by demographic data to ensure our redistricting efforts did not disproportionately affect minority communities. By setting parameters such as the maximum white population percentage, we intentionally shaped districts to either increase or maintain minority representation. This approach served a dual purpose: it reinforced the image of fair play by

aligning with the principles of the Voting Rights Act, while simultaneously enabling us to consolidate voting blocks that favored our preferred political party. However, this focus on demographics required a careful balancing act to ensure compliance with legal standards and avoid the impression of racial gerrymandering. Ultimately, its impact was relatively limited, however. We noticed that the demographic shift was not substantial (even though there were some minute changes). Our iterative process across district pairs, flipping border precincts, demonstrates our algorithm's methodical approach to redistricting, however, this same process launched many difficulties at us. Ensuring that each flip improved our map's political leaning without disrupting contiguity or demographic balance required a delicate balance and often resulted in a trade-off between political advantage and the risk of detectable gerrymandering.



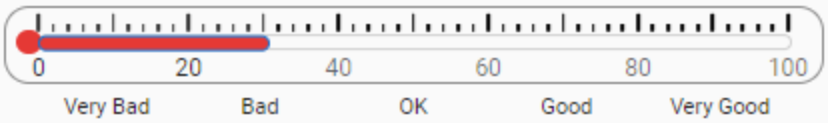
The deviation from proportionality is slightly greater than the official Congressional map, with a -15.68% score. This suggests a calculated approach to grant an advantage to one party, yet the score is close enough to the official map's -16.24% that it could be defended as reflecting the natural political geography of New Jersey rather than an overt manipulation.

Competitiveness

All else equal, prefer maps that are more competitive.

Metric		Description
• Competitiveness	22.36%	The percentage of competitive districts. Bigger is better.

Rating



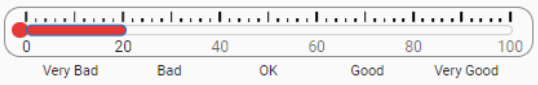
In comparison to the actual congressional map, our version shows a marked decrease in competitiveness, denoted by a 22.36% score. This lower level of competitiveness may initially appear to be a disadvantage, as it implies a greater number of safe seats and could diminish the vitality of electoral contests. However, this could also be presented as a result of our efforts to respect existing political communities and maintain stability in representation, which can be a legitimate redistricting objective. Moreover, the relatively high score for minority representation, standing at 71, could be leveraged as a counterargument to any claims of unfairness, suggesting that our map effectively ensures minorities have a voice in the political process, which is a critical aspect of fair representation under the law.

Compactness

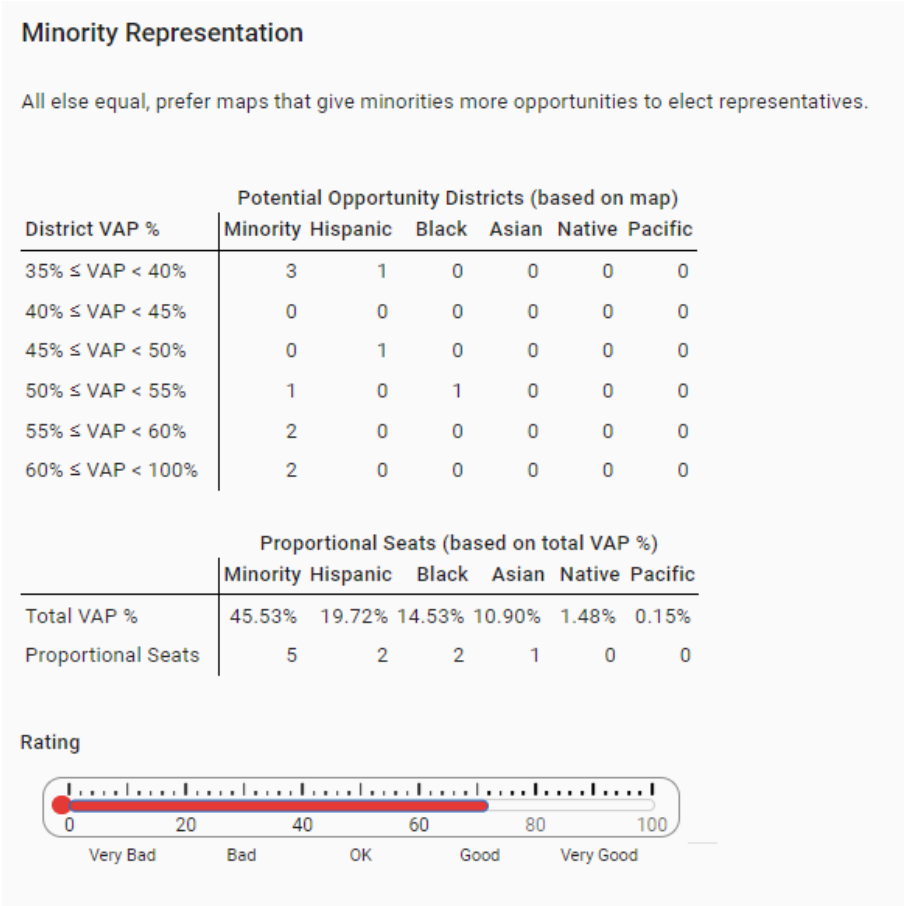
All else equal, prefer maps with districts that are more compact.

Metric		Description
• Reock	0.3451	Measures how dispersed district shapes are. Bigger is better.
• Polsby-Popper	0.1086	Measures how indented district shapes are. Bigger is better.

Rating



The compactness of our districts, while not ideal, is not overly low, indicating that we have made an effort to maintain reasonable district shapes. Our map could be critiqued for potentially prioritizing political factors over geographic coherence, but given that the compactness score does not drastically diverge from the official map, this aspect of the design might withstand legal scrutiny. It's important to note, however, that the lower compactness could weaken our position if challenged in court, as opponents might argue it reflects an intent to dilute or concentrate certain voting blocs.



Minority Representation

All else equal, prefer maps that give minorities more opportunities to elect representatives.

Potential Opportunity Districts (based on map)

District VAP %	Minority	Hispanic	Black	Asian	Native	Pacific
35% ≤ VAP < 40%	3	1	0	0	0	0
40% ≤ VAP < 45%	0	0	0	0	0	0
45% ≤ VAP < 50%	0	1	0	0	0	0
50% ≤ VAP < 55%	1	0	1	0	0	0
55% ≤ VAP < 60%	2	0	0	0	0	0
60% ≤ VAP < 100%	2	0	0	0	0	0

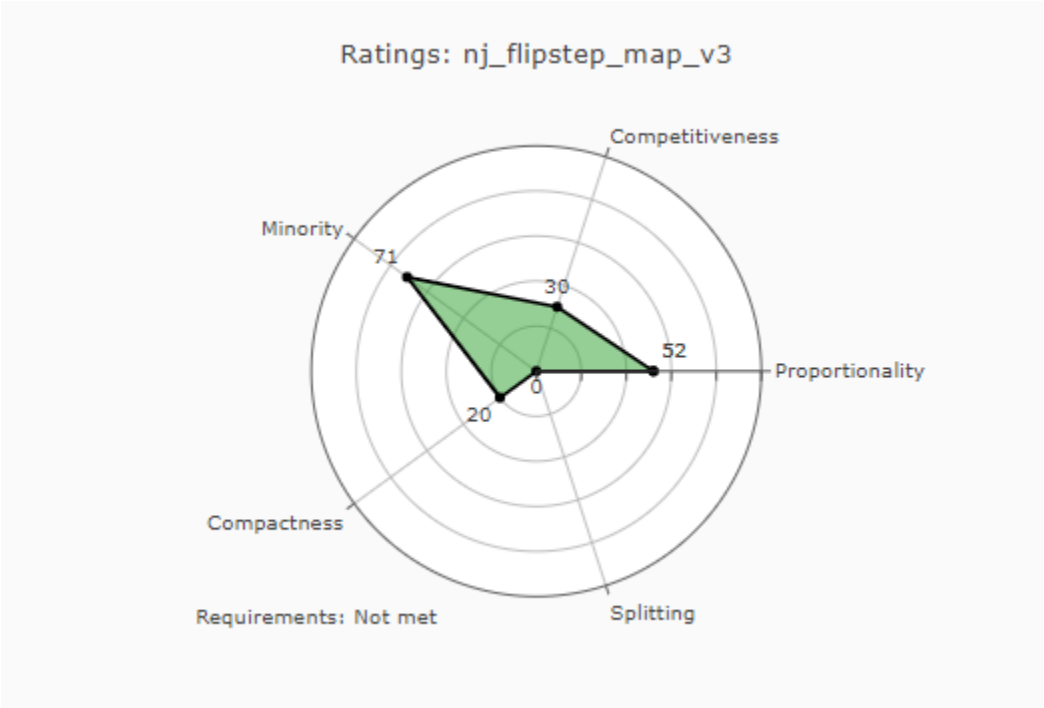
Proportional Seats (based on total VAP %)

	Minority	Hispanic	Black	Asian	Native	Pacific
Total VAP %	45.53%	19.72%	14.53%	10.90%	1.48%	0.15%
Proportional Seats	5	2	2	1	0	0

Rating

Our map's most significant strength lies in its attention to minority representation, which not only aims to meet the requirements of the Voting Rights Act but also provides a strong defense against

accusations of unfair gerrymandering. By showing that our map potentially increases the number of districts where minority groups could influence the outcome, we add a layer of complexity to the argument over fairness. In essence, our redistricting efforts present a map that, while crafted to be 'unfair' in its political outcomes, is nuanced enough in its execution to present a facade of fairness, possibly withstanding legal challenges based on its adherence to certain redistricting principles.



When coding our 'unfair' map, we set out to tilt the balance towards a certain party using our flip-step algorithm, while still trying to look fair. We carefully picked which neighborhoods to move into different districts, hoping to boost our chosen side without making it too obvious. Despite our careful planning, we didn't manage to execute on our goals as well as we would have preferred. We did have some wins, but keeping the changes under the radar while also following the rules turned out to be trickier than we thought.