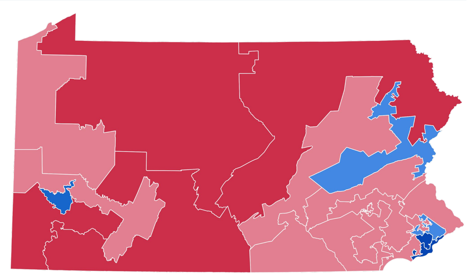
A Novel Approach to Algorithmic Redistricting: Combating Gerrymandering with Artificial Intelligence Tools

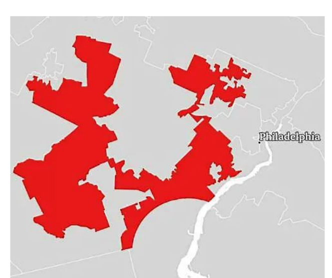
Deven Hagen

Background

- Each state is apportioned a number of House delegates proportional to its population
- Every ten years, after the Census, each state is divided up into districts that each elect a single representative to Congress every two years until the next Census
- The state legislature is typically tasked with creating the district maps, although some states have started using independent redistricting commissions (IRCs)
- Subject to partisan and racial gerrymandering
  - Gerrymandering: the manipulation of boundaries to benefit a certain group
    - Partisan gerrymandering: maps that give certain political parties more power
    - Racial gerrymandering: maps that dilute the influence of certain demographic groups
- Use of "packing" and "cracking" to rearrange populations in certain ways

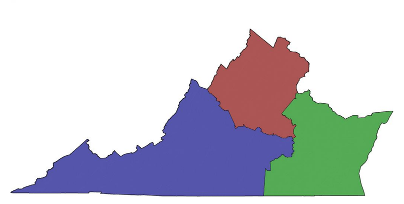


Party	Republican	Democratic
Last election	**12**	7
Seats won	**13**	5
Seat change	▲ 1	▼ 2
Popular vote	2,710,070	**2,793,538**
Percentage	48.77%	**50.28%**



Existing Solutions

- Many attempts to use computers/algorithms to improve upon human mapmaking and allow for more fair solutions
  - Brian Olsen's compactness-focused approach
  - Auto-Redistrict (genetic algorithm)
- While these are excellent and effective, the redistricting process is complex and subjective, so creating more unique algorithms will strengthen the process
- Independent redistricting commissions can use the suggestions of algorithms when creating maps
  - Provides impartial guidance that is not binding
- Goal is to come up with a novel algorithm that is demonstrated to meaningfully improve upon current maps



MICHIGAN INDEPENDENT CITIZENS REDISTRICTING COMMISSION

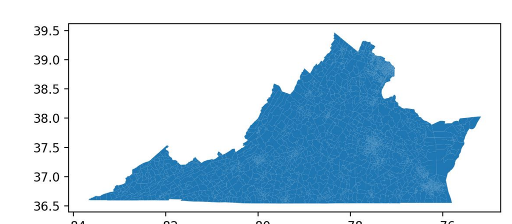
Research Question

Can I create a new algorithm that will improve upon existing maps in certain states, as measured using selected metrics that encompass partisan bias, racial bias, competitiveness, and compactness?

Data Preprocessing

- I used demographic/geospatial precinct data from the Census Bureau and pared down the columns for the essentials
- I used election data from state-specific sources
  - Ex: Virginia Dept. of Elections
- I experimented with two sets of election results
  - Composite of state and national election results 2016-20
  - 2020 presidential election results
- I used the Python library GeoPandas, an extension of Pandas, to handle the geospatial data, combining all of my data into a single "GeoDataFrame"
- In order to ensure that my perimeter and area calculations were correct, I used a Conus Albers projection
  - EPSG:5070
- At this point, I was ready to try implementing some algorithms

GEOID20	DISTRICT	NAME	ADJPOP	CVAP	WHITE	HISPANIC	BLACK	ASIAN
5.1E+10	100	Chincoteag	3357	2800	2660	75	10	0
5.1E+10	100	Atlantic	1519	1271	1147	47	181	3
5.1E+10	100	Greenbackv	2890	1746	1669	53	46	13
5.1E+10	100	New Churcl	3590	2738	1167	76	1347	92
5.1E+10	100	Bloxom	1384	949	831	4	117	1
5.1E+10	100	Parksley	1742	1041	836	39	184	4
5.1E+10	100	Saxis	411	302	268	4	18	0
5.1E+10	100	Mappsville	1806	1375	627	13	578	3
5.1E+10	100	Rue	1922	1142	327	157	611	0
5.1E+10	100	Accomac	2898	2097	1345	106	649	14
5.1E+10	100	Tangier	440	315	315	0	0	0
5.1E+10	100	Nandua	3720	2987	2073	212	667	4
5.1E+10	100	Bobtown	1156	963	824	18	98	28
5.1E+10	100	Melfa	1921	1297	732	28	429	22
5.1E+10	100	Wachapreag	1009	878	663	30	140	0
5.1E+10	100	Painter	3773	2994	1296	29	1743	4
5.1E+10	100	Woodbrook	4043	3235	2440	35	569	174
5.1E+10	100	Branchland	2395	1965	1580	35	325	0
5.1E+10	100	Agnor-Hurt	4153	2616	1607	156	805	53
5.1E+10	100	Dunlora	4419	3555	3155	70	195	130
5.1E+10	100	Northside	5516	4593	3642	242	536	164



GerryChain

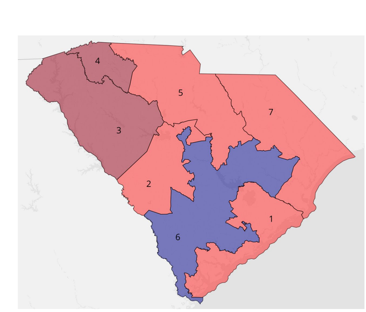
- GerryChain - Python library for using Markov Chain Monte Carlo methods in the redistricting sphere
  - Relatively recent - last updated in 2024
- Stores state precincts as a graph, allowing for much faster runtime
- Allowed me to easily create initial partitions of a state with population-balanced districts
- Has built-in ways to calculate statistics of interest through its *metrics* sublibrary
- I decided to use the SingleMetricOptimizer, which allows the user to optimize a certain value
  - Can accomplish this through either Short Bursts, Simulated Annealing, or Tilted Runs
    - I chose Short Bursts, because this method seemed to work best for me
- To move forward, however, I would need to come up with my own unique heuristic that would provide the novelty in my project

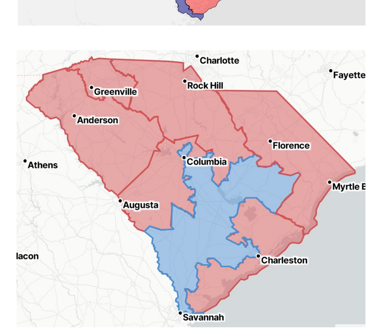
```
from gerrychain import Graph, Electi
from gerrychain.updaters import cut_
from gerrychain.metrics.compactness
from gerrychain.proposals import rec
from gerrychain.tree import recursiv
from gerrychain.accept import always
from gerrychain.constraints import s
from gerrychain.optimization import
from gerrychain.metrics.partisan imp
```

```
total_population = sum([graph.nodes[n][pop_col] for n in graph.nodes])
assignment = recursive_tree_part(
    graph,
    range(11),
    total_population/11,
    pop_col,
    0.0075
)

initial_partition = GeographicPartition(graph, assignment, myupdaters)
```

South Carolina: Current Map



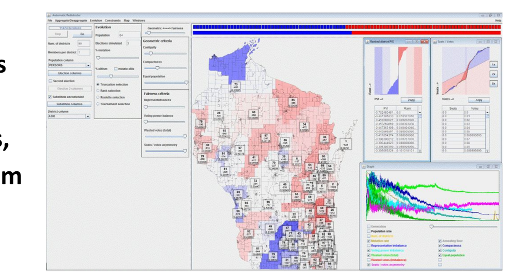


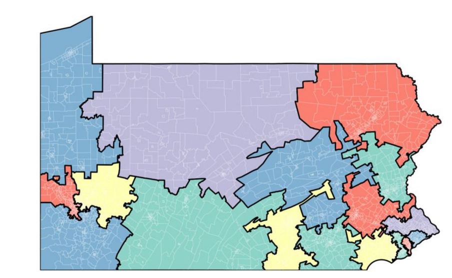
Metric	
• Reock	0.3611
• Polsby-Popper	0.2147

Metric	Value	Favors Democrats in this % of Scenarios*	More Skewed than this % of Historical Plans‡	More Pro-Democratic than this % of Historical Plans‡
Efficiency Gap	19.3% Pro-Republican	<1%	99%	1%
Declination	0.51 Pro-Republican	<1%	91%	5%

Unsuccessful Attempts

- At first, I attempted to implement a genetic algorithm
  - Had seen examples of this before, such as with the Auto-Redistrict website
- Struggled with creating a way to "merge" maps, which is an important part of a genetic algorithm
  - Breeding stage
- Decided to shift toward an implementation of Haas et al.'s Seed-Fill-Shift-Repair
- Because my dataset was quite large, and because manipulating a GeoDataFrame is not very efficient, it took forever to run, despite my time-saving strategies
  - Ex: pickling, creating a graph of neighboring precincts, combining steps in the algorithm
- The districts in maps created by SFSR are also quite jagged, which isn't ideal
- Eventually found GerryChain



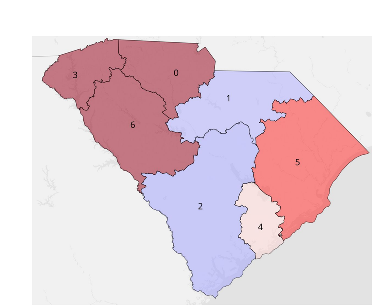


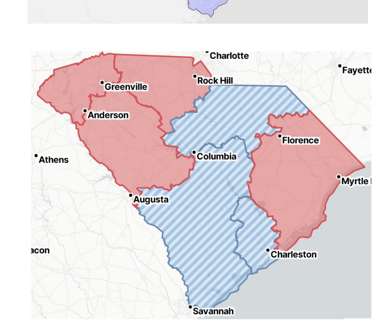
Heuristic Development and Implementation

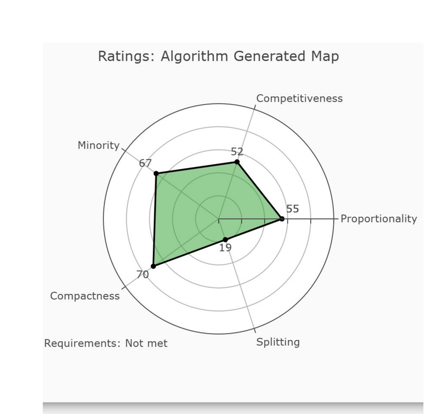
- Common considerations in the redistricting process:
  - Population balance: The districts should have roughly equal populations
  - Contiguity: Each district should be one piece, with no holes or tears
  - Compactness: The districts should have a large area to perimeter ratio
  - Partisan bias: The map should not benefit one party over the other
  - Racial bias: The map should not disenfranchise any racial group
- I set a population balance of 0.75% and a contiguous set of districts as my map constraints
- To measure compactness, I used the average Polsby-Popper score for a district, equal to 4*pi*area over perimeter squared; a circle has a value of 1, the maximal value
- To measure partisan bias, I used a compound statistic formed from the average of three metrics: partisan bias, efficiency gap, and mean-median difference
- Because African-Americans tend to vote strongly for Democrats, partisan and racial bias go hand-in-hand, so I expected my heuristic to combat racial bias as well
- In my heuristic, I actually used 1 minus the average district Polsby-Popper score, as well as the absolute value of the average of the three partisan bias metrics, so that I could minimize the overall heuristic as opposed to maximizing it
- I used a scaling factor on the compound partisan bias metric so that both terms would have influence on the algorithm
- Heuristic:

20*(abs(efficiency_gap)+abs(mean_median)+abs(partisan_bias))/3+(num_districts-polsby_popper)

South Carolina: My Algorithm's Map





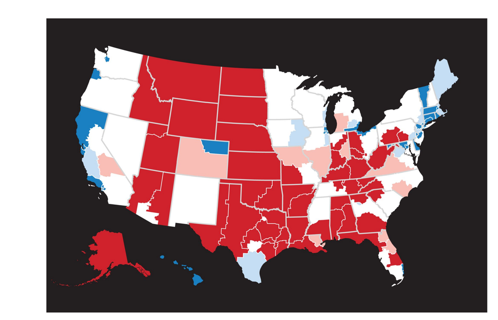


Metric	
• Reock	0.4816
• Polsby-Popper	0.2891

Metric	Value	Favors Democrats in this % of Scenarios*	More Skewed than this % of Historical Plans‡	More Pro-Democratic than this % of Historical Plans‡
Efficiency Gap	5.0% Pro-Republican	34%	51%	22%
Declination	0.02 Pro-Democratic	68%	6%	54%

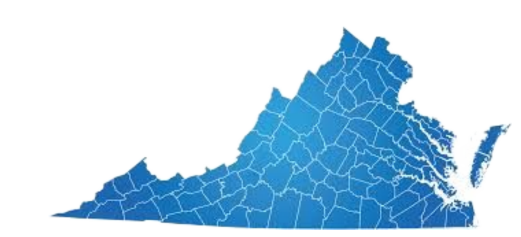
Conclusion

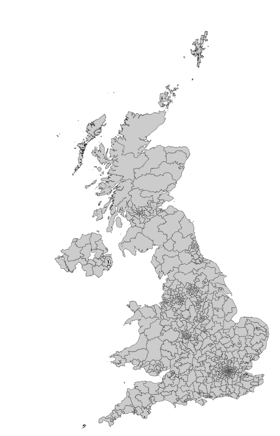
- I was able to successfully create a novel approach for automated redistricting that combats gerrymandering
- My algorithm can be used in concert with others to increase fairness in the redistricting process, hopefully in conjunction with IRCs
- Eventually, it is possible that algorithms could take over the process entirely, although we are not at that stage yet
- Limitations
  - I used precincts as the building blocks for districts in my algorithm
    - Cannot make districts of ideal size
  - My heuristic is certainly not comprehensive yet
    - Will work to make it more complex
  - It may be impossible to find a "perfect heuristic"



Future Work and Next Steps

- Make heuristic more complex
  - Take into account other considerations
  - County splitting, competitiveness
- The next redistricting cycle isn't until 2030, but I hope to contact existing independent redistricting commissions and provide them with my algorithm
- Potentially put up a website that would allow users to create maps that are optimized for certain metrics
- Examine the merits of this approach with redistricting for state legislatures
  - Gerrymandering is just as prevalent at the state level and oftentimes worse
- Research whether it can be scaled to other countries, potentially with multiparty systems
  - Most have independent commissions, but my algorithm can still help them





References

[1] Altman, M., & McDonald, M. P. (2013). Paradoxes of political reform: Congressional redistricting in florida. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.2421908

[2] Chen, J., & Rodden, J. (2015). Cutting through the thicket: Redistricting simulations and the detection of partisan gerrymanders. *Election Law Journal*, *14*(4).

[3] Cirincione, C., Darling, T. A., & O'Rourke, T. G. (2000). Assessing south carolina's 1990s congressional districting. *Political Geography*, *19*(2), 189-211. https://doi.org/10.1016/s0962-6298(99)00047-5

[4] Duignan, B. (2024, September 21). Gerrymandering. Encyclopedia Britannica. https://www.britannica.com/topic/gerrymandering

[5] Fifield, B., Higgins, Michael, Imai, K., & Tarr, A. (2020). Automated redistricting simulation using markov chain monte carlo. *Journal of Computational and Graphical Statistics*, *29*(4), 715-728. https://doi.org/10.1080/10618600.2020.1739532

[6] Haas, C., Hachadoorian, L., Kimbrough, S. O., Miller, P., & Murphy, F. (2020). Seed-Fill-Shift-Repair: A redistricting heuristic for civic deliberation. *PloS one*, *15*(9), e0237935. https://doi.org/10.1371/journal.pone.0237935

[7] Kevin Baas. (2018, May 11). *About*. AutoRedistrict. http://autoredistrict.org/

[8] Liu, Y. Y., Cho, W. K. T., & Wang, S. (2016). PEAR: A massively parallel evolutionary computation approach for political redistricting optimization and analysis. *Swarm and Evolutionary Computation*, *30*, 78-92. https://doi.org/10.1016/j.swevo.2016.04.004

[9] Olson, B. (2020, October 17). *2010 redistricting results*. BDistricting. https://bdistricting.com/2010/

[10] *Virginia data*. (2024). Redistricting Data Hub. https://redistrictingdatahub.org/state/virginia

[11] Volić, I. (2024). *Making democracy count: How mathematics improves voting, electoral maps, and representation*. Princeton University Press.