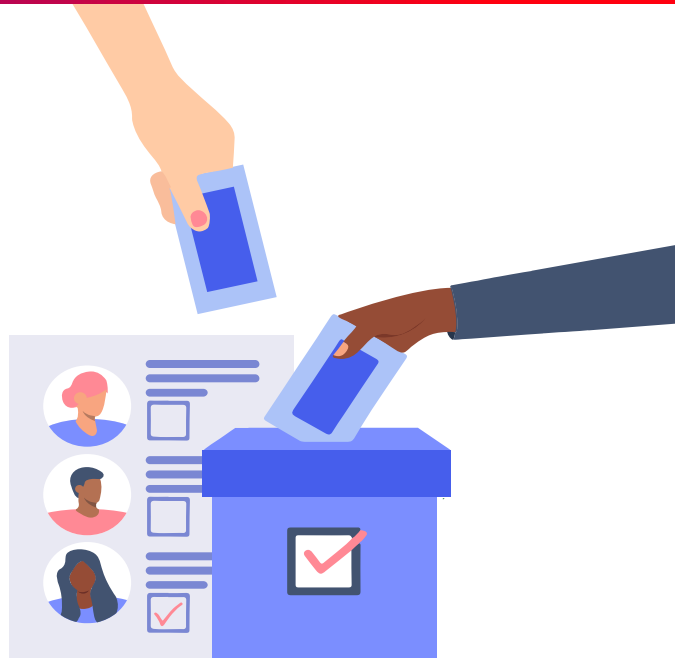
Have I Been Gerrymandered?

Mason duBoef, Henry Jack, Josh Marion,
Keerti Sundaram, Sam West



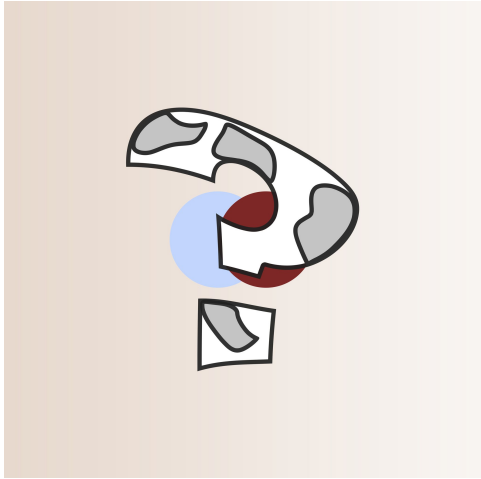
What is Have I Been Gerrymandered?

Have I Been Gerrymandered? is an interactive web tool

We created Have I Been Gerrymandered? to provide voters with an easy to use tool, indicating whether their own districts are gerrymandered.

Gerrymandering scores are determined by an algorithm we developed based on academic research.

Users can access our site at https://have-i-been-gerrymandered.github.io/react/

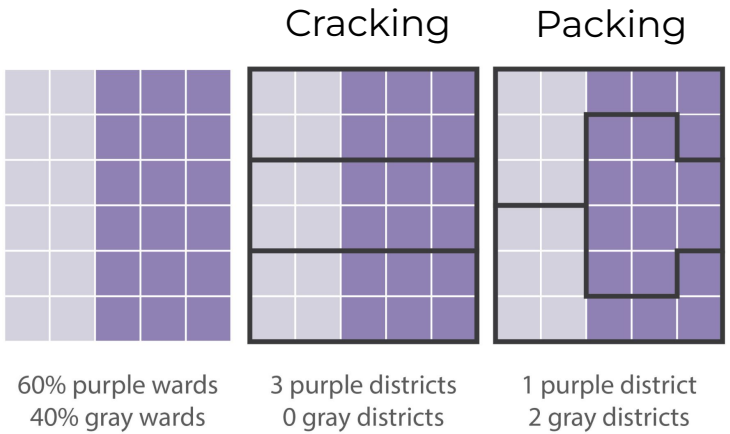


“Informed voters, fair borders”

What is Gerrymandering?

- Manipulating the boundaries of a set of electoral districts to favor a certain party
- Candidate with most votes wins district

- Two main methods
  - “Packing”
    - Placing many opposition voters in a single non-competitive district
    - Gives an advantage in neighboring districts
  - “Cracking”
    - Cutting up a block of opposition voters
    - Making sure to maintain a majority in each district



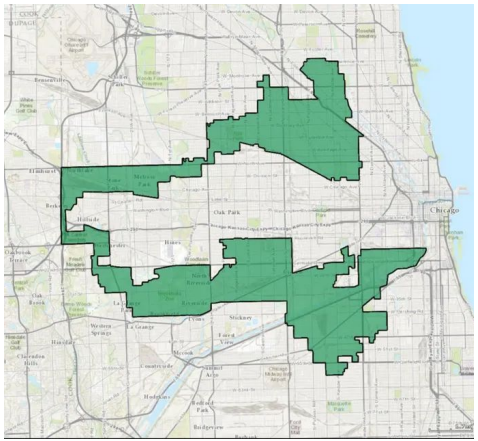
Real World Gerrymandering

- Representatives draw district maps for their own state
  - Long time swing states usually fair
    - Nevada, Indiana etc.
  - Worst gerrymandering in states historically controlled by one party with growing opposition

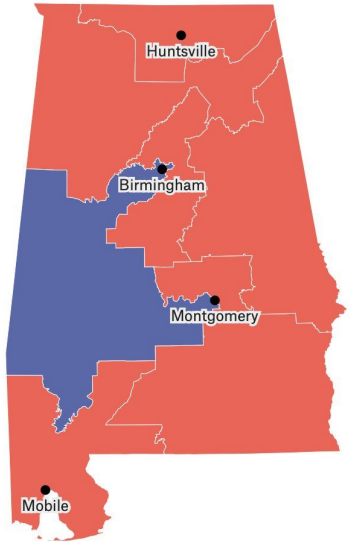
- State supreme courts can turn down a new map if they determined to be unconstitutionally gerrymandered

- Demographics, vote counts and district shape can all indicate gerrymandering
- Fair redistricting usually aims to
  - Keep demographically similar communities together
  - Keep district populations similar
  - Not produce electoral advantage for either party

Chicago's "Ear Muffs"



Packing In Alabama



Features

Gerrymandering Scoring Algorithm

Score Breakdown

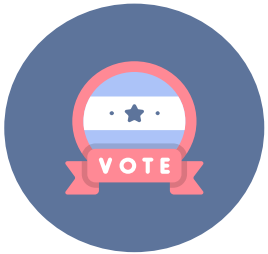
Interactive Map

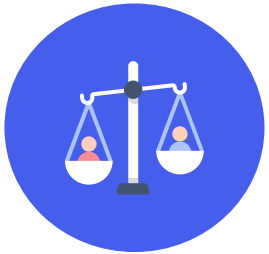
Navigation Bar, Algorithm & Information Page

State District Maps

Hosting on GitHub Pages

“What District Am I?” Feature





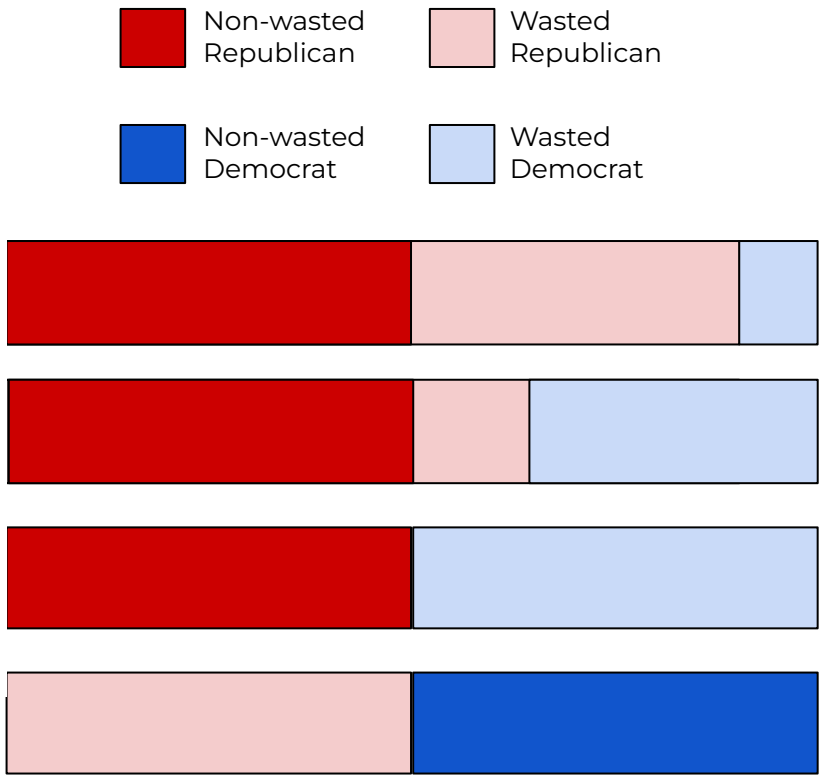
Demo

Our Algorithm: Efficiency Gap

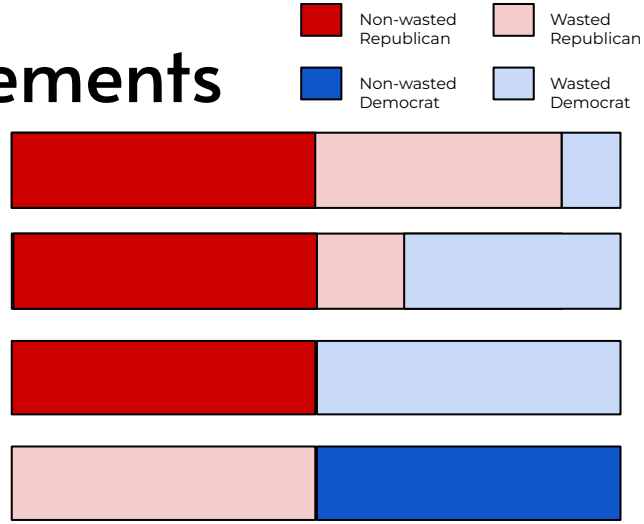
w = votes for winner
l = votes for loser

$$w_{wasted} = \frac{w+l}{2} - 1$$
$$l_{wasted} = l$$

$$\text{Efficency Gap} = \frac{|w_{wasted} - l_{wasted}|}{w+l}$$



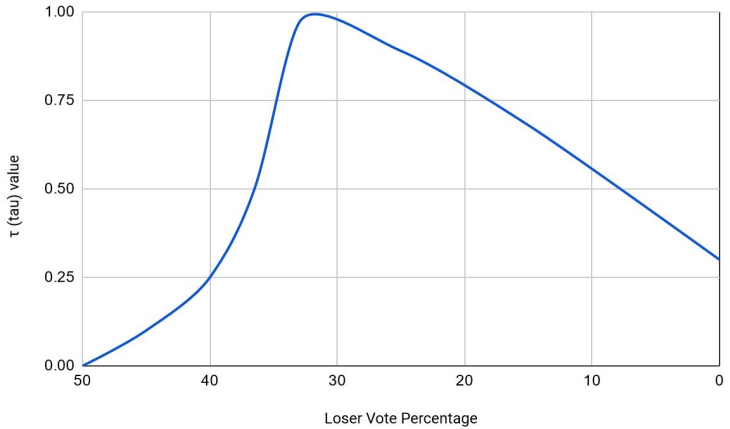
Our Algorithm: Improvements



- Votes assigned variable amount of “wastedness” (τ) between 0 and 1

- Surplus votes for winner
  - First surplus votes -> lower τ
  - Later surplus votes -> higher τ

- Votes for loser
  - Thresholds somewhat arbitrary
  - Narrow loss ⟶ lower τ
  - Blow out loss ⟶ lower τ
    - Indicates loser packing
  - Comfortable loss (~35%) ⟶ high τ
    - Indicates opposition cracking



Development & Technical Strategy

We used the **Scrum** framework with 1 week sprints
We used Python and React.js throughout development

Tools we used:

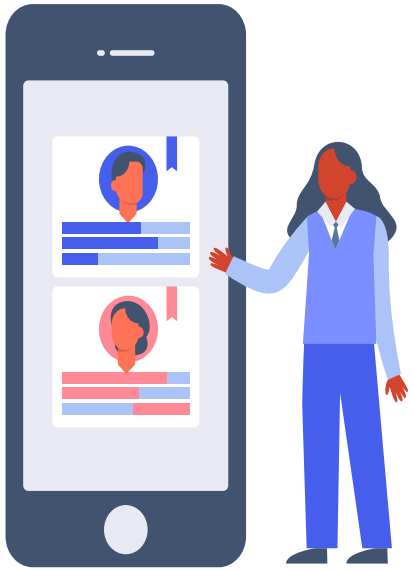
Each task throughout our development process was categorized into one of four categories:

1. Algorithm development
2. Front end development
3. Integration & Testing
4. Documentation

Design Patterns

We used the **Conditional Rendering** React Design Pattern, as when the user navigates to a different page, the page rerenders.

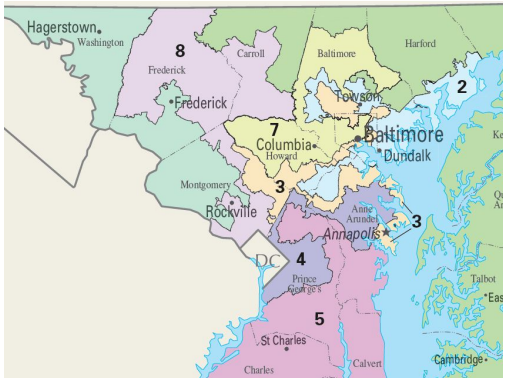
We also used the **Template Method**, as it was helpful for rendering in React.



Future of our Application

Congressional districts are constantly altered, therefore maintaining our tool is important so it continues to be useful for users. Most of us have participated in RCOS and this would make a great open source software, there is an opportunity to contribute to both frontend and backend features.

v.2.0 [and beyond] Features



Interactive map with zooming functionality

Zip code input that takes the user straight to their district

Resource links for information on unfair redistricting

Updated gerrymandering algorithm that factors demographic and geographic data into it

Dark and colorblind mode to improve accessibility

Future of our Application

v.2.0 [and beyond] Features

Better formatting for mobile devices

Scripts to pull data on district lines and voting data

Contact form

Allow users to create accounts to save districts

Push notifications during election season or after redistricting

Uniform district maps

Overt indication of which party the gerrymandering favors

Score influenced by score of neighboring districts

Group Dynamics

Product Owner: Mason

Scrum Master: Keerti

Developers: Josh, Henry, Sam, Mason, and Keerti

Communicated via group chat

Held weekly updates, as well as sprint review and retrospective meetings.

Held documentation meetings during weeks that deliverables were due.

What Did and Didn't Work

What Worked

Scrum Framework

Sprint Planning, Sprint Review, Sprint Retrospective

Jira Taskboard

GitHub Issues

GitHub Repositories

Creating and Updating a Project Schedule

Small Meeting with Mav :)

Quantifying Gerrymandering

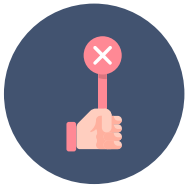
What Didn't Work

Daily Standup

Timeline

Ambitious Feature List

Lack of Experience with Frontend Design



What We Learned

Henry: I learned a lot about React and how to do more frontend coding. I also learned about how important it is to meet and be in consistent communication with your teammates. I also never had experience working with UML so working with diagrams was new to me.

Josh: I learned how valuable using a development framework is. Scrum is very useful, and it definitely helped propel us towards our goal. Also, I learned how important documentation is, as it greatly helped us keep track of our project.

Sam: Gained knowledge on how to work in a group with strict deadlines. Strengthened my knowledge on OOP concepts and learned helpful skills like the SCRUM methodology, design patterns, testing a project and using React.

What We Learned

Mason: Creating a detailed release schedule with measurable milestones, interim and beta releases stood out to me as useful practice that I anticipate I'll use in future projects. In the past, I've struggled to meet deadlines, not breaking personal projects down into intermediary steps/releases. Also, the process of researching algorithms & exploring niche technical solutions within a structured schedule was a valuable experience for me.

Keerti: I learned about the importance of documentation in a software project and how diagrams such as class and sequence aid the development process. Additionally, I learned more about developing a new project in a group and how the design, development, and test processes work.

References

https://developer.wordpress.org/coding-standards/wordpress-coding-standards/javascript/

https://aglowiditsolutions.com/blog/react-design-patterns/#:~:text=React%20Design%20Patterns%20are%20core,js%20project.

https://www.changeil.org/2020/03/why-chicagos-4th-congressional-district-those-earmuffs-are-about-fairness-and-not-gerrymandering/

Warrington, Gregory S. “Quantifying Gerrymandering Using the Vote Distribution.” Election Law Journal: Rules, Politics, and Policy, vol. 17, no. 1, 2018, pp. 39–57., https://doi.org/10.1089/elj.2017.0447.

QUESTIONS