NORTHERN ARIZONA UNIVERSITY

School of Informatics, Computing, and Cyber Systems

User Manual

Date: December 1st, 2024
Team Name: FairyMander
Sponsor: Bridget Bero
Professor: Isaac Shaffer
Mentor: Vahid Nikoonejad Fard

Team Members:
Izaac Molina (Team Lead)
Dylan Franco
Jeysen Angous
Sophia Ingram
Ceanna Jarrett

1 Introduction

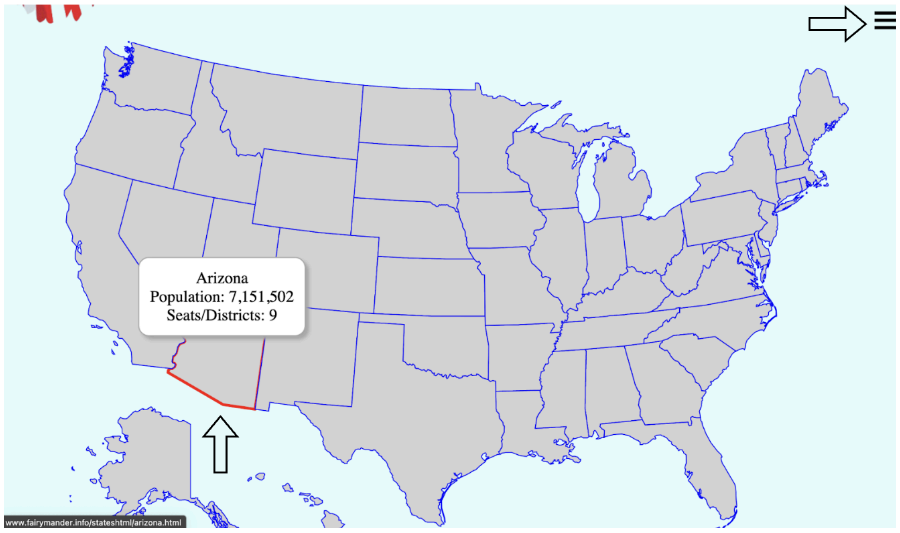
We are pleased to have been a part of this impactful project against gerrymandering. Team FairyMander was able to create a product that meets all your needs and specifications. Our website is fully operational with features to display our generated districts, educate users on redistricting, and fairness metrics. Along with our website, we have an easy-to-use algorithm where users can generate their own districts. It has been our pleasure to create a system that leaves a positive impact on the world.

2 Website

To visit the website, go to https://fairymander.info/

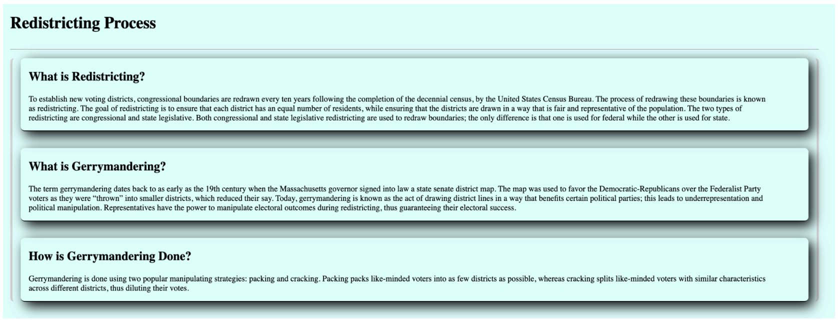
2.1 Landing Page

The landing page consists of the interactable United States map and hamburger dropdown menu.



2.2 Redistricting Process

One of the educational portions of the landing page educates the user on redistricting



Redistricting Process

What is Redistricting?

To establish new voting districts, congressional boundaries are redrawn every ten years following the completion of the decennial census, by the United States Census Bureau. The process of redrawing these boundaries is known as redistricting. The goal of redistricting is to ensure that each district has an equal number of residents, while ensuring that the districts are drawn in a way that is fair and representative of the population. The two types of redistricting are congressional and state legislative. Both congressional and state legislative redistricting are used to redraw boundaries; the only difference is that one is used for federal while the other is used for state.

What is Gerrymandering?

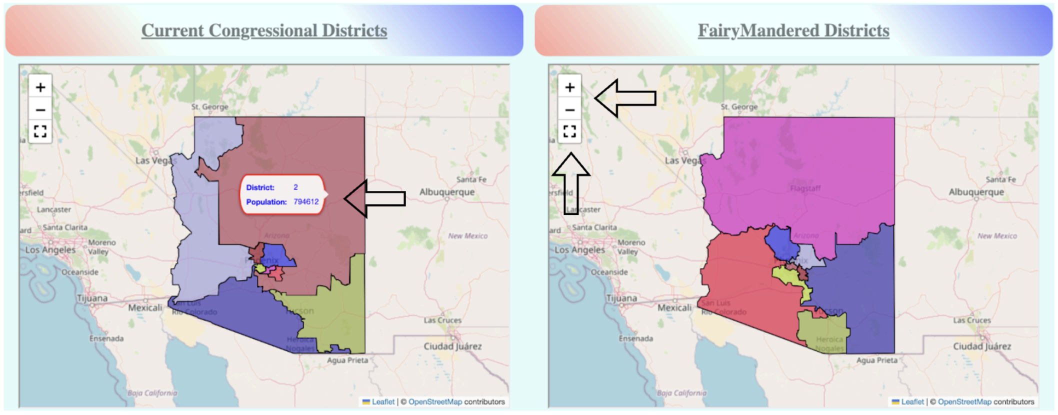
The term gerrymandering dates back to as early as the 19th century when the Massachusetts governor signed into law a state senate district map. The map was used to favor the Democratic-Republicans over the Federalist Party voters as they were "thrown" into smaller districts, which reduced their say. Today, gerrymandering is known as the act of drawing district lines in a way that benefits certain political parties; this leads to underrepresentation and political manipulation. Representatives have the power to manipulate electoral outcomes during redistricting, thus guaranteeing their electoral success.

How is Gerrymandering Done?

Gerrymandering is done using two popular manipulating strategies: packing and cracking. Packing packs like-minded voters into as few districts as possible, whereas cracking splits like-minded voters with similar characteristics across different districts, thus diluting their votes.

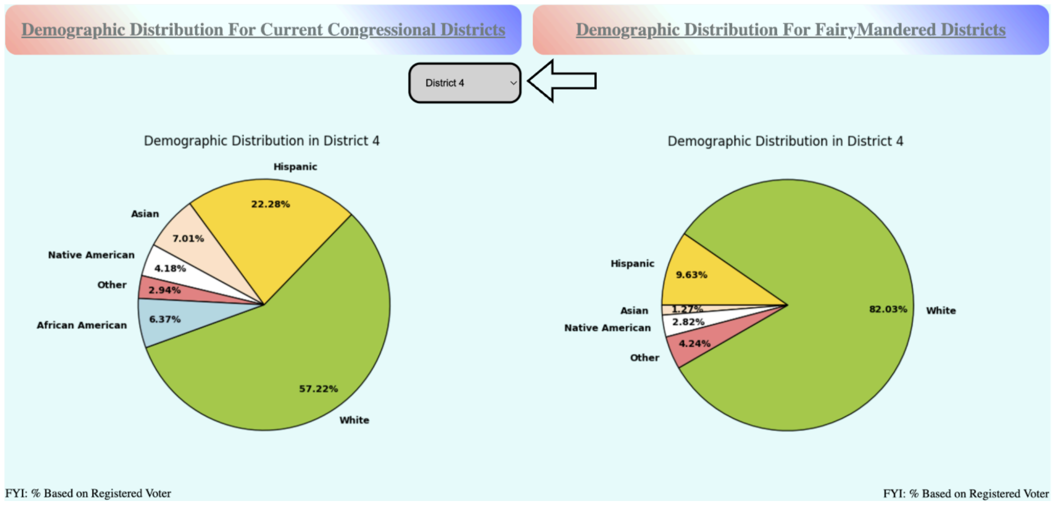
2.3 Districts and Demographic Distribution

A side-by-side comparison is shown for both current congressional districts and "FairyMandered" Districts with the ability to zoom in and out and fullscreen for readability. Once a user hovers over districts, its number and population are shown.



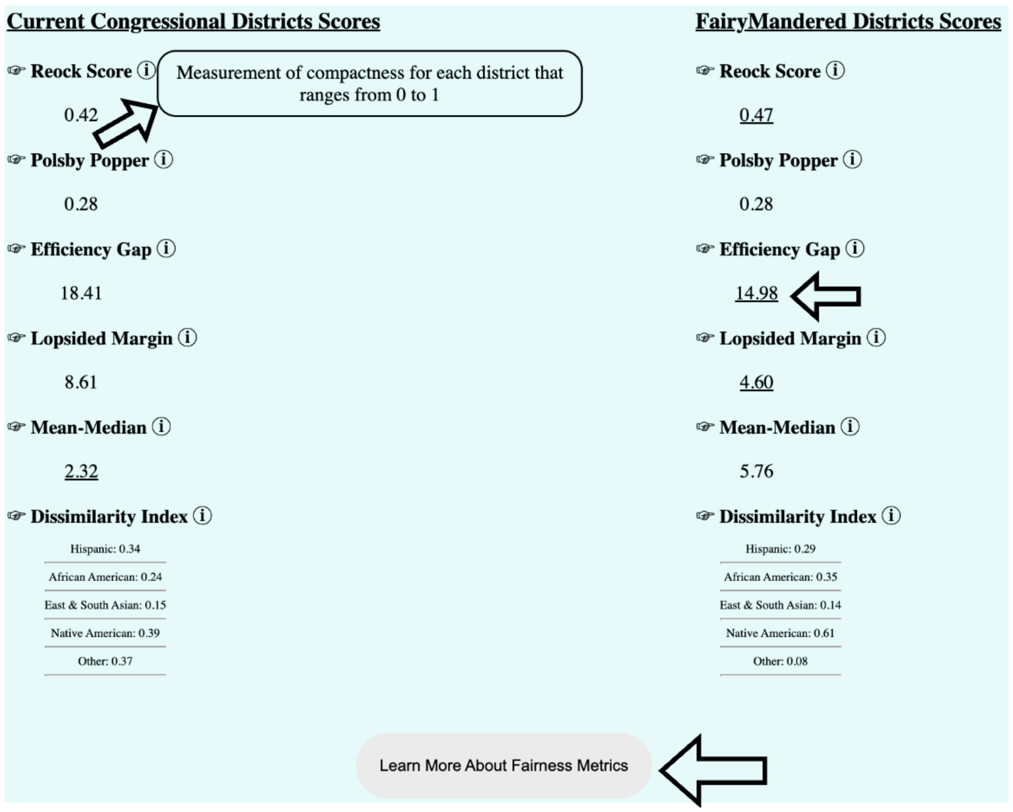
2.4 Demographic Distribution

Demographic distributions are shown for each district in a dropdown menu format in the form of a pie chart for the current congressional and "FairyMandered" districts.


Demographic Distribution For Current Congressional Districts
Demographic Distribution For FairyMandered Districts
District 4
Demographic Distribution in District 4
Hispanic 22.28%
Asian 7.01%
Native American 4.18%
Other 2.94%
African American 6.37%
White 57.22%
Demographic Distribution in District 4
Hispanic 9.63%
Asian 1.27%
Native American 2.82%
Other 4.24%
White 82.03%
FYI: % Based on Registered Voter
FYI: % Based on Registered Voter

2.5 Fairness Metrics

Fairness metrics scores are shown in a side-by-side comparison; an underlined score indicates which of the two metrics is better. An "i" icon will show what the score measures once hovered. Additionally, upon clicking "Learn More About Fairness Metrics", the user will be taken to a glossary page that will have a more in-depth description of a fairness metric score or the ability to view the formula.



Current Congressional Districts Scores

Reock Score ⓘ

Measurement of compactness for each district that ranges from 0 to 1

0.42

Polsby Popper ⓘ

0.28

Efficiency Gap ⓘ

18.41

Lopsided Margin ⓘ

8.61

Mean-Median ⓘ

2.32

Dissimilarity Index ⓘ

Hispanic: 0.34
African American: 0.24
East & South Asian: 0.15
Native American: 0.39
Other: 0.37

FairyMandered Districts Scores

Reock Score ⓘ

0.47

Polsby Popper ⓘ

0.28

Efficiency Gap ⓘ

14.98

Lopsided Margin ⓘ

4.60

Mean-Median ⓘ

5.76

Dissimilarity Index ⓘ

Hispanic: 0.29
African American: 0.35
East & South Asian: 0.14
Native American: 0.61
Other: 0.08

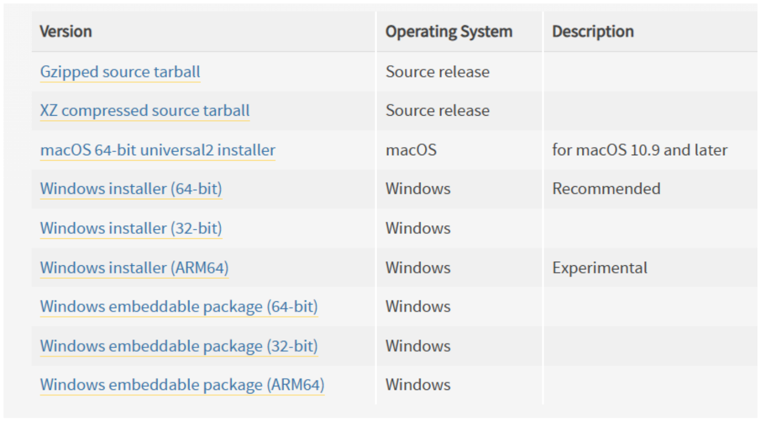
3 Installation

Step 1: Install VS Code

Visit this website and install VS Code for your system.

Step 2: Install Python 3.11.0

1. Visit the following website to download python 3.11 Scroll down to the `Files` section and download the version for your system: '**MacOS**' for Mac and '**Windows 64-bit**' for Windows



Version	Operating System	Description
Gzipped source tarball	Source release	
XZ compressed source tarball	Source release	
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later
Windows installer (64-bit)	Windows	Recommended
Windows installer (32-bit)	Windows	
Windows installer (ARM64)	Windows	Experimental
Windows embeddable package (64-bit)	Windows	
Windows embeddable package (32-bit)	Windows	
Windows embeddable package (ARM64)	Windows	

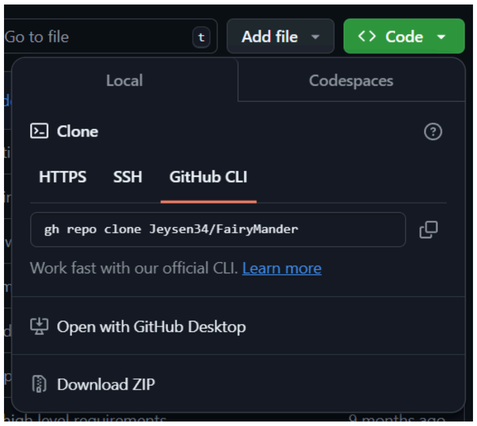
2. Run the installer
3. Ensure you select `Add python.exe to PATH` before clicking `Install Now`
4. Check if your installation was successful by opening a terminal and entering:
   a. **Windows:** `python –version`
   b. **MacOS/Linux:** `python3 –version`

   it should say Python 3.11.0 or a newer version

Additional Help: For Windows installation, watch this video until 2:22**.** For Mac installation, watch this video until 3:47

Step 3: Download the Codebase

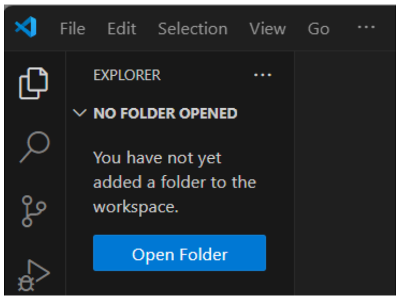
1. Visit the following website to install our codebase.
2. Click the green `Code` button, then select `Download ZIP`.



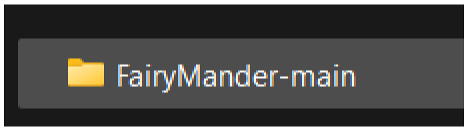
```
gh repo clone Jeysen34/FairyMander
```

Work fast with our official CLI. Learn more

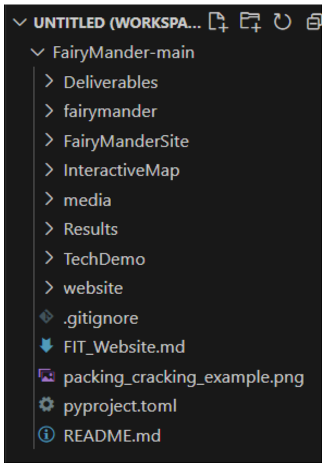
3. Move the downloaded ZIP file to your desired location and extract its contents.
4. Open VS Code, click on `Open Folder`, and navigate to the extracted Fairymander codebase folder.



You have not yet added a folder to the workspace.



Your workspace should look like this:

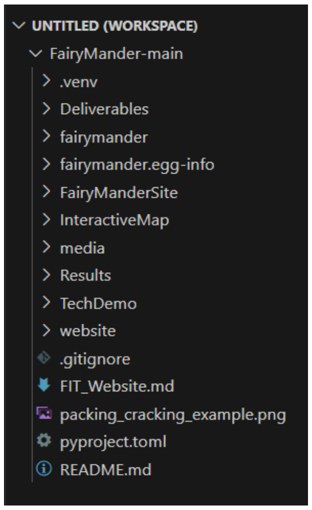


```
UNTITLED (WORKSPA...
  FairyMander-main
    > Deliverables
    > fairymander
    > FairyManderSite
    > InteractiveMap
    > media
    > Results
    > TechDemo
    > website
    .gitignore
    FIT_Website.md
    packing_cracking_example.png
    pyproject.toml
    README.md
```

Step 4: Create a Python Virtual Environment

Video Tutorial for Step 4:

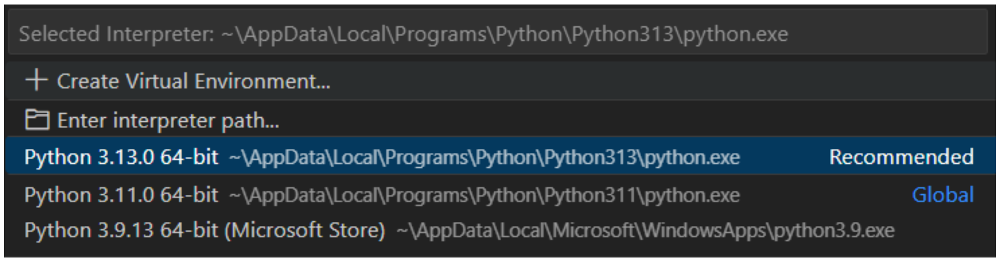
1. In VS Code, open the terminal by clicking `View > Terminal` from the menu.
2. Navigate to the main Fairymander codebase folder via the terminal.
   NOTE: It is important that you make the venv in the main folder containing all the other folders FairyMander, Results, Deliverables, etc. Like in the below image:



```
UNTITLED (WORKSPACE)
  FairyMander-main
    > .venv
    > Deliverables
    > fairymander
    > fairymander.egg-info
    > FairyManderSite
    > InteractiveMap
    > media
    > Results
    > TechDemo
    > website
    .gitignore
    FIT_Website.md
    packing_cracking_example.png
    pyproject.toml
    README.md
```

3. Create a venv:
   a. **Windows:** Press `Ctrl + Shift + P`
   b. **MacOS/Linux:** Press `Cmd + Shift + P`

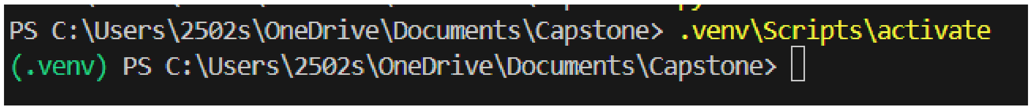
   Then, select `Python: Create Environment`, choose `Venv`, and finally, under `Python: Select Interpreter`, pick `Python 3.11.0`.



```
Selected Interpreter: ~\AppData\Local\Programs\Python\Python313\python.exe
+ Create Virtual Environment...
Enter interpreter path...
Python 3.13.0 64-bit ~\AppData\Local\Programs\Python\Python313\python.exe              Recommended
Python 3.11.0 64-bit ~\AppData\Local\Programs\Python\Python311\python.exe              Global
Python 3.9.13 64-bit (Microsoft Store) ~\AppData\Local\Microsoft\WindowsApps\python3.9.exe
```

4. Activate the virtual environment by entering the appropriate command in the terminal:
   a. **Windows:** `.venv\Scripts\activate`
   b. **MacOS/Linux:** `source .venv/bin/activate`

   You will know the environment is activated when you see the green `(.venv)`



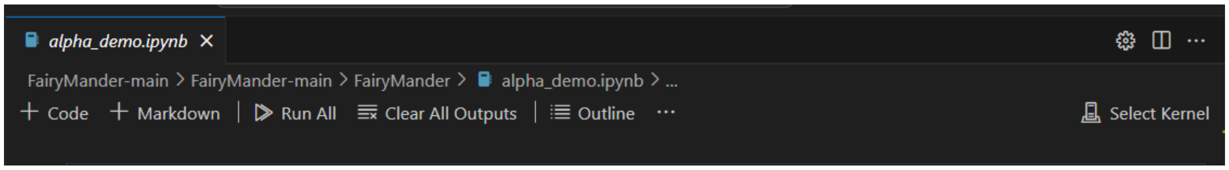
```
PS C:\Users\2502s\OneDrive\Documents\Capstone> .venv\Scripts\activate
(.venv) PS C:\Users\2502s\OneDrive\Documents\Capstone>
```

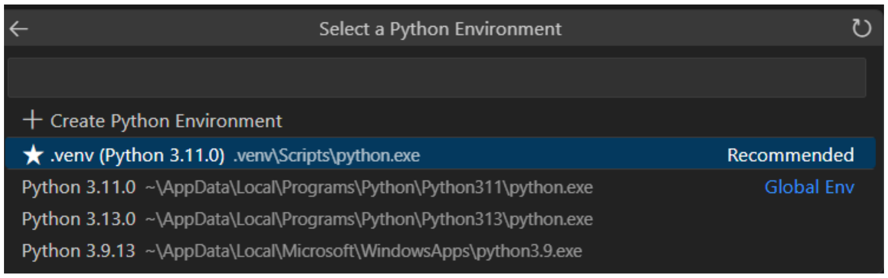
5. Move into the main FairyMander folder and build the FairyMander package by entering the command `pip install -e .` in the terminal

Additional Help: VS Code [document tutorial]() and a [video tutorial]()

Step 5: Generate Districts

Open `alpha_demo.ipynb` (located in the `Fairymander` folder) and click `Select Kernel` at the top-right corner.
Note: If you do not see your virtual environment listed in the options click `Select Another Kernel` then `Python Environments` then select your .venv

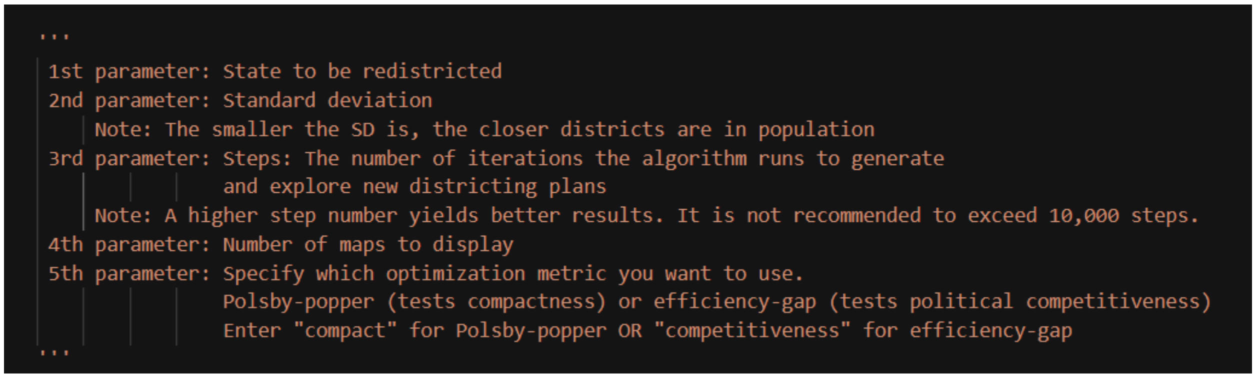




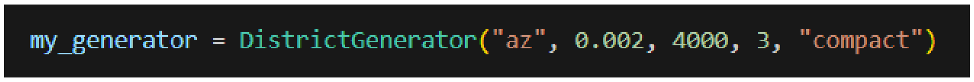
```
Select a Python Environment
+ Create Python Environment
★ .venv (Python 3.11.0) .venv\Scripts\python.exe                                Recommended
Python 3.11.0 ~\AppData\Local\Programs\Python\Python311\python.exe              Global Env
Python 3.13.0 ~\AppData\Local\Programs\Python\Python313\python.exe
Python 3.9.13 ~\AppData\Local\Microsoft\WindowsApps\python3.9.exe
```

Note: If prompted to install additional dependencies, such as `ipykernel`, click `Install` to proceed.

The district generator takes 5 parameters:



```
'''
1st parameter: State to be redistricted
2nd parameter: Standard deviation
    Note: The smaller the SD is, the closer districts are in population
3rd parameter: Steps: The number of iterations the algorithm runs to generate
               and explore new districting plans
    Note: A higher step number yields better results. It is not recommended to exceed 10,000 steps.
4th parameter: Number of maps to display
5th parameter: Specify which optimization metric you want to use.
               Polsby-popper (tests compactness) or efficiency-gap (tests political competitiveness)
               Enter "compact" for Polsby-popper OR "competitiveness" for efficiency-gap
'''
```

Ex) 

```
my_generator = DistrictGenerator("az", 0.002, 4000, 3, "compact")
```

The above line of code will perform redistricting on Arizona with a standard deviation of 0.002 and 4,000 steps. It will display 3 maps and will use Polsby-Popper as its optimization metric (meaning the generator will create districts that are compact)

Notes on Parameters:

- The state must be entered in all lowercase (e.g., 'ca,' 'tx,' 'nv').

- The higher the standard deviation the longer the algorithm takes. The lower the standard deviation the closer the districts will be in population

- The higher the steps the longer the algorithm takes, higher steps will give the algorithm a higher chance of producing better districts

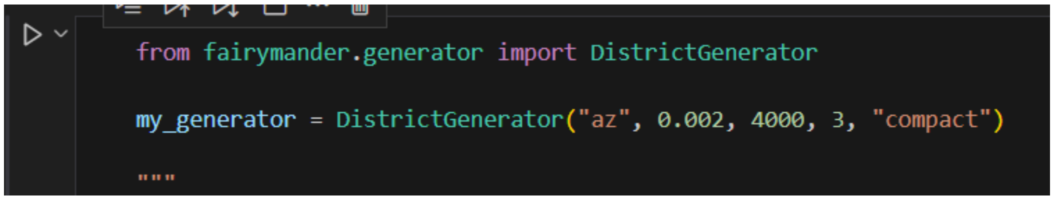
- For districts with a small population (Utah < population <= Ohio) a standard deviation of 0.002 to 0.004 is a good starting point and 4,000 to 7,000 steps should be good. For districts with a large population (California, Texas, etc.) 0.004 to 0.007 for the SD and 6,000 to 10,000 steps should be efficient

- Some low population states, like Idaho, can produce good maps with as little as 100 steps

- If none of the maps being produced are good, increase the number of steps, **OR** keep generating districts with a small step size (less than 1,000) until you get one that is satisfactory. Do not exceed 10,000 steps

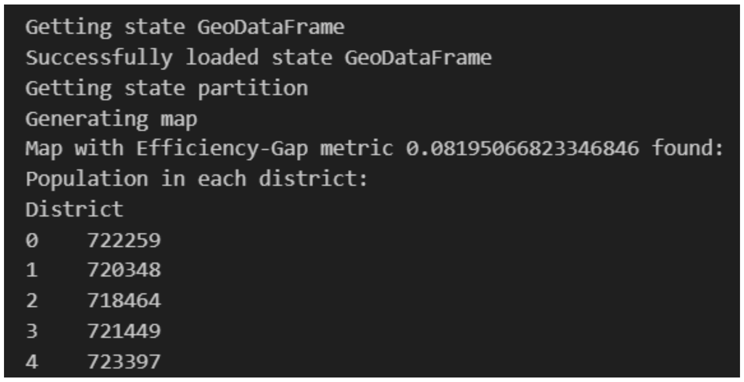
- If population differences between districts are too large—meaning the district with the smallest population and the district with the largest population differ by an amount that exceeds the user's desired threshold (e.g., 800, 1,000, 2,000, etc.)—reduce the standard deviation (SD) by 0.001

To generate districts hit the play button at the very top left:

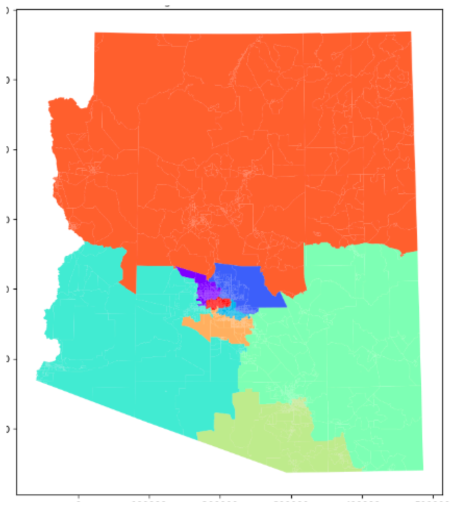


```
from fairymander.generator import DistrictGenerator

my_generator = DistrictGenerator("az", 0.002, 4000, 3, "compact")

"""
```

The districts will begin generating, the user will be given some status messages and the current step. Once the generator is done the optimization metric score will be shown along with each district's population and district maps, see below.

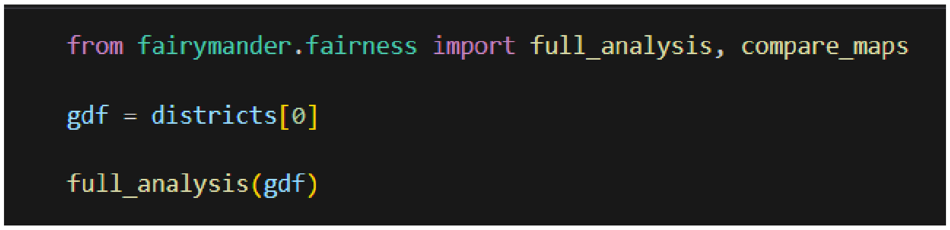


```
Getting state GeoDataFrame
Successfully loaded state GeoDataFrame
Getting state partition
Generating map
Map with Efficiency-Gap metric 0.08195066823346846 found:
Population in each district:
District
0    722259
1    720348
2    718464
3    721449
4    723397
```



Districts can then be evaluated for fairness using the next block of code, also shown below. Enter in the number for which map you would like to evaluate, 0 for the 1st map, 1 for the 2nd, etc.

Running this code will show the user fairness metric scores for the first generated district map:



```
from fairymander.fairness import full_analysis, compare_maps

gdf = districts[0]

full_analysis(gdf)
```

The user can compare generated districts using the next block of code:

```
compare_maps(districts[1], districts[2])
```

And by running the next block of code, the user can compare the generated districts to the current districts:

```
from fairymander.data import get_curr_district_file

gdf = get_curr_district_file('az')
print(gdf)
compare_maps(districts[0], gdf)
```

Using the next block of code, users can convert the generated maps to folium. This will create an interactive map, allowing the user to zoom in and out and see which cities are located in which district.

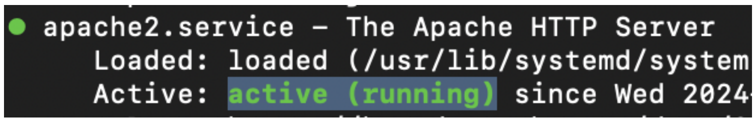
```
from fairymander.folium_converter import map_to_folium

res = map_to_folium('az', districts[0])

res
```

Video Tutorial of [running the generator]()

4 Configuration and Daily Operation

The website will be live using the Apache HTTP server alongside Amazon Web Services (AWS). Should the sponsor request server access, the PEM key and password will be provided. Should any issues arise with viewing the web application, the following are some common actions that can be taken to solve any problems that may occur.

`sudo systemctl status apache2` - use to check the status of the server
`sudo systemctl restart apache2` - use to restart the server, should the status be inactive.



```
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system
     Active: active (running) since Wed 2024
```

5 Maintenance

The main bit of maintenance that will need to be done is to download new files from census.org when the new data releases every 10 years. In order to do this, we have included **census_data_scraper.py** with the rest of the files. Here is how to run it:

Prerequisites

Assuming you already have Python installed from the instructions above, you will need to run the following commands in your terminal.
WINDOWS: `pip install requests` and then `pip install beautifulsoup4`
MacOS/Linux: `pip3 install requests` and then `pip3 install beautifulsoup4`

Running the Program

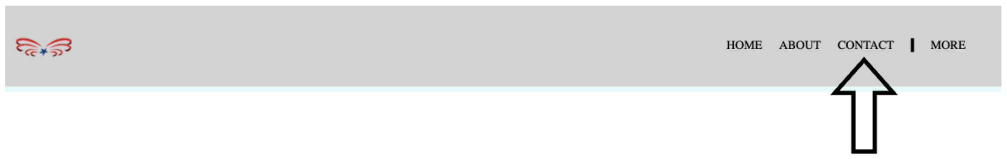
2. *Download the Script* - Save the provided Python script (**census_data_scraper.py**) to a folder where you want the data to be downloaded to.
3. *Update the Base URL (optional)* - The script is set to download census data for the year 2020 by default. If you want to scrape data for a different year, open the script in a text editor and modify the `base_url` variable near the top of the file to point to the desired year's URL.
4. *Access the Script* - Open a terminal window and navigate to the directory where you saved the script. For example:
   a. **Windows:** `cd path\to\your\script`
   b. **MacOS/Linux:** `cd path/to/your/script`
5. *Run the Script* - Run the script by typing:
   a. **Windows:** `python census_data_scraper.py`
   b. **MacOS/Linux:** `python3 census_data_scraper.py`

If you would like more detailed information on how the script works, you can view the detailed comments available in the file.

6 Troubleshooting

Website

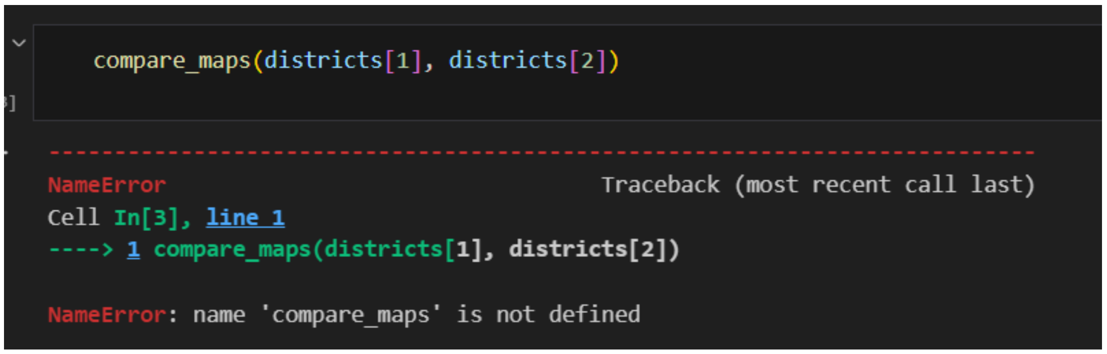
Should any issues arise with the usage of the FairyMander web application, the user may submit a feedback form by clicking on the "CONTACT" tab, located at the navigation panel on top of the page.



Algorithm

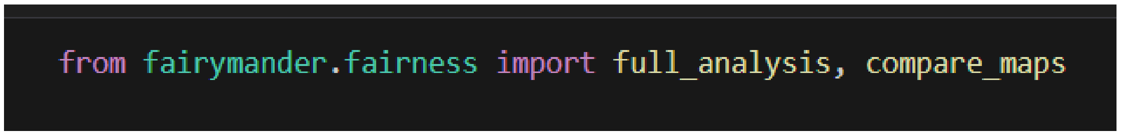
1. If the algorithm tells you something along the lines of: `Failed to find a balanced cut` try re-running it ~2 times, if you keep getting the same error raise the SD by 0.001

2. If you get an error like `NameError: name 'compare_maps' is not defined`:



```
compare_maps(districts[1], districts[2])

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[3], line 1
----> 1 compare_maps(districts[1], districts[2])

NameError: name 'compare_maps' is not defined
```

Fix it by running the block of code that contains the function's import:



```
from fairymander.fairness import full_analysis, compare_maps
```

7 Conclusion

Thank you for trusting us in the production of your product, we hope it meets all your expectations and proves to be useful. It has been our pleasure to produce a product with the potential to change lives. Best wishes from your FairyMander developers, Izaac Molina, Dylan Franco, Jeysen Angous, Sophia Ingram, and Ceanna Jarrett.