# Pro-Vision

*Setu Loomba (CNetID: setu)*
*Diego Mendoza-Martinez (CNetID: diegomendozamz)*
*Angel Rodriguez-Gonzalez (CNetID: angelrodriguezg)*

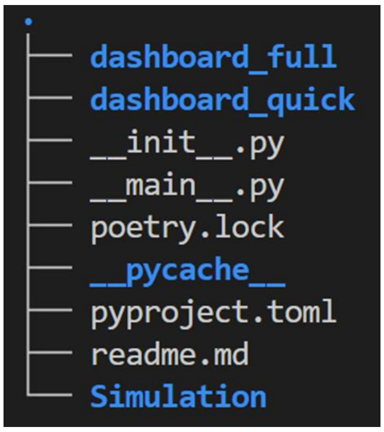
## Abstract

Pro-Vision is an effort to highlight visual correlations/disparities between socioeconomic indicators and access to time-sensitive public services/facilities (i.e., provisions) for citizens of the City of Chicago, as well as to test the resilience of the system. Using data from the City of Chicago, the US Census, TravelTravel API, and other secondary sources, we created a layered map of the City of Chicago with provision locations and time-distance coverage (i.e., isochrones) at the Census Tact level (2010). We then overlap this coverage with a colored map depending on vulnerability degree, based on selected socioeconomic variables. Finally, we stress the network by applying shocks to it, in order to measure the sensitivity of the current coverage to stochastic changes.

The insights derived from this interactive map and dashboard intend to better inform policymakers about the under/over coverage of provisions on vulnerable geographical locations.

## Software Structure

The main software structure is as follows:



```
.
├── dashboard_full
├── dashboard_quick
├── __init__.py
├── __main__.py
├── poetry.lock
├── __pycache__
├── pyproject.toml
├── readme.md
└── Simulation
```

Figure 1. Directory Tree

- *dashboard_full*: Directory with files that pull data from the Travel API and processes information to construct the overlayed map.
- *dashboard*_quick: Directory with files that quickly pull pre-processed data to construct the overlayed map.
- *__init__.py*: File to mark the directory as a Python package.
- *__main__.py*: File to execute the Python program.
- *poetry.lock*: Virtual environment installer.
- *__pycache__*: Cache files.
- pyproject.toml: Settings file for the unified Python program.
- *Readme.md*: User guide.
- *Simulation*: Directory with files that simulate shocks in the City of Chicago as a network.

To see the extended software structure and all dependencies see Appendix A.

## Data and Code Responsibilities

The data collection process was completed by all project members as Table 1 shows. The collected socioeconomic indicators were used to construct the heat map layer to visualize Quartiles (Census Tact level) and to build the nodes of the network system (Community Area level). Information about provisions were collected from the City of Chicago’s Data Portal to locate these facilities in a second map layer. Time-distance data was collected from Travel Time API to (1) trace in another map layer the coverage boundaries of the provisions’ locations, and to (2) measure minimum distance to police stations from Community Areas.

| **Data Point** | **Data Source** | **Person Responsible** |
|---|---|---|
| Socio-economic Indicators | Census, Chicago Atlas (CSVs) | Diego Mendoza-Martinez |
| Location of provisions | City of Chicago (CSVs) | Angel Rodriguez-Gonzalez |
| Boundaries of census tracts and community areas | City of Chicago (CSVs) | Setu Loomba |
| Isochrones | Travel Time API | Setu Loomba |
| Times matrix | Travel Time API | Angel Rodriguez-Gonzalez |

Table 1. Data source responsibilities

The code responsibilities were divided as follows:

| **Responsibility** | **Person Responsible** | **Main code File/s** |
|---|---|---|
| API wrapper, data pipelining and data cleaning | Setu Loomba | All other files in *dashboard_quick* and *dashboard_full* |
| Visualizations and Dashboard Interactivity | Diego Mendoza-Martinez | In dashboard_quick and dashboard_full: app.py, utility.py |
| Shock Simulation and Network Interactivity, and API wrapper | Angel Rodriguez-Gonzalez | All files in Simulation Directory |
| Project Integration | Team | |

Table 2. Code responsibilities

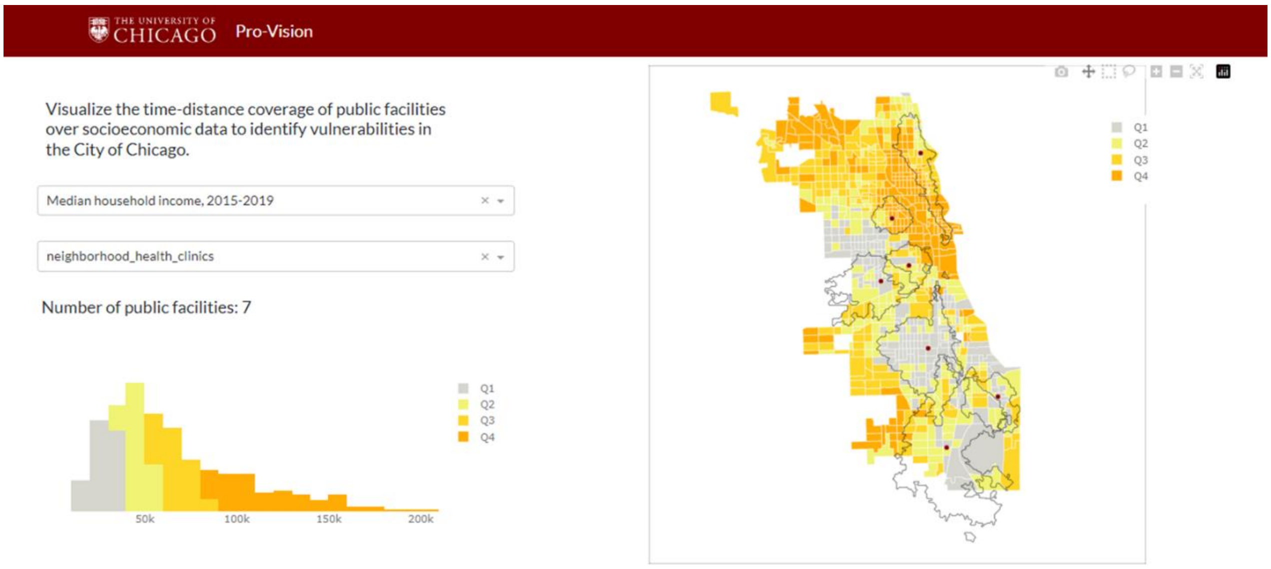
## User guide

### Coverage Map Dashboard

Pro-vision's Dashboard and visual features are designed to allow the policymaker/analyst to derive key insights in an easy-to-understand and -interact manner. The display is divided in two columns. The left column contains a brief description of the Map's objective, two dropdown menus and a histogram. The first dropdown menu contains several socioeconomic variables that can be displayed in the map and the histogram. The second dropdown menu contains a list of Provisions (e.g., Neighborhood Health Clinics, Police Stations, etc.) that the user can select as well. At the end, there is a histogram that is only included if the user selected a socioeconomic variable. The histogram can be toggled between Quartiles (i.e., Q1 lowest quartile to Q4 highest quartile) to select/deselect socioeconomic groups; the user can also hover the histogram to see the ranges for each Quartile.

The right column of the dashboard shows the map, which is traced by Census Tacts (2010). If no socioeconomic variable and no provision is selected, the map is displayed in gray and the user can hover the map to see the GEOIDs. If a socioeconomic variable is selected without any provision, the map is displayed as a choropleth map where the user can toggle as well by Quartiles. If a provision is selected without any socioeconomic variable, the map is displayed as empty (i.e., no color) but with points indicating the exact longitude and latitude of all the facilities of this provision over the map. Most importantly, the 10-minute distance coverage of these facilities are traced as boundaries in this selection. If a socioeconomic variable and provision are selected, the map displays the choropleth map and the points with its coverage boundaries. This map is fully interactive, allows to toggle between Quartiles and hover over the points to get the address of the facility and hover over the colored census tacts to get the value of the socioeconomic variable.

The full dashboard displays as follows:


THE UNIVERSITY OF CHICAGO Pro-Vision

Visualize the time-distance coverage of public facilities over socioeconomic data to identify vulnerabilities in the City of Chicago.

Median household income, 2015-2019

neighborhood_health_clinics

Number of public facilities: 7

Figure 2. Full Dashboard

The Provisions that are included in the dropdown menu are:

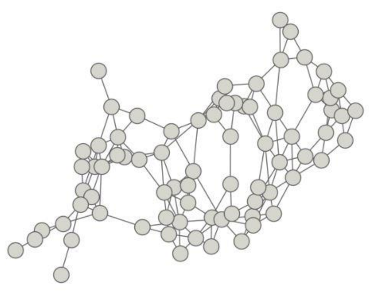
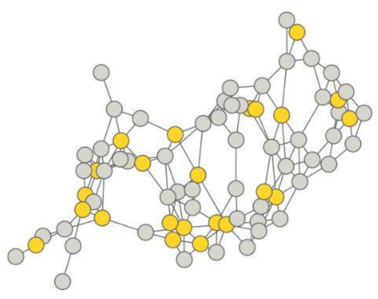
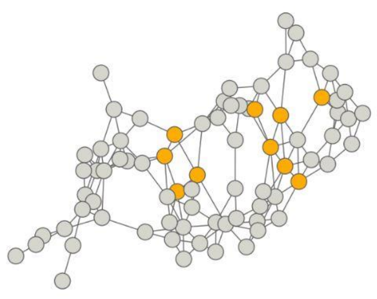
- Neighborhood Health Clinics
- Warming Centers
- Police Stations
- Fire Stations

The Socioeconomic Variables that are included in this map are:

- Uninsured rate (% of residents, 2015-2019)
- Homicide (crimes), 2017-2021
- Major crime (crimes), 2016-2020
- Violent crime (crimes), 2016-2020
- Eviction rate (% of renter-occupied households, 2018)
- Severely rent-burdened (% of renter-occupied housing units), 2015-2019
- Traffic crashes (number of crashes), 2021
- High School Graduation rate (% of residents), 2015-2019
- Unemployment rate (%), 2015-2019
- Median Household Income, 2015-2019
- Per Capita Income, 2015-2019
- Poverty rate (% of residents), 2015-2019
- Demographics, Minorities (% of residents), 2016-2020
- Population (residents), 2015-2019

<u>Shock Simulation</u>

As a final step, we move from a geographical representation of Chicago to an abstract one by turning the city into a network, in order to make use of Graph Theory to simulate random shocks. We build an adjacency matrix where each node is a community area, and these are connected by edges if they are bordering. All nodes are labelled with the following binary attributes: {'Tensioned area': 0-1, 'Provision center within the area': 0-1}.

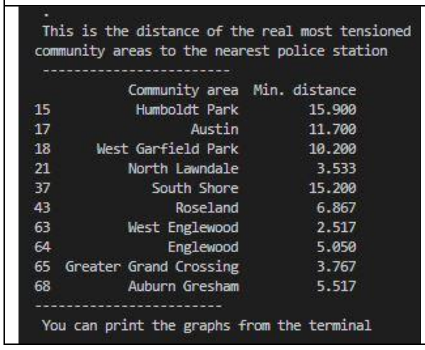
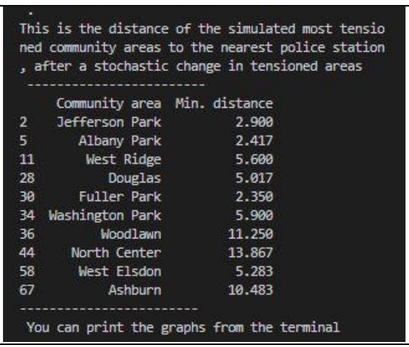
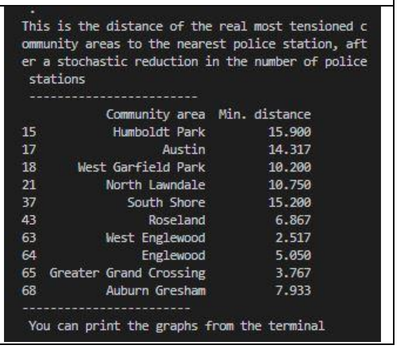
| Chicago as a network of nodes | Community areas with police station | Tensioned community areas |
|---|---|---|
|  |  |  |

We restrict our analysis to homicide rate as socioeconomic indicator and police stations as provision service, and apply two independent kinds of shocks:

- Stochastic allocation of tension degree to each community area

- Reduction of the number of police stations by a factor of 0.4

To measure the resilience of the system to these shocks (change in social tension patterns / abrupt budgetary reduction), we build a table with the most tensioned areas and the distance to the nearest police stations after the shock.

| Status quo | Shock in tensioned areas | Shock in number of provision centers |
|---|---|---|
|  |  |  |

```
 .
 This is the distance of the real most tensioned
community areas to the nearest police station
 -----------------------
             Community area  Min. distance
15            Humboldt Park         15.900
17                   Austin         11.700
18       West Garfield Park         10.200
21           North Lawndale          3.533
37              South Shore         15.200
43                 Roseland          6.867
63           West Englewood          2.517
64                Englewood          5.050
65  Greater Grand Crossing           3.767
68            Auburn Gresham         5.517
------------------------
 You can print the graphs from the terminal
```

```
 .
This is the distance of the simulated most tensio
ned community areas to the nearest police station
, after a stochastic change in tensioned areas
 -----------------------
    Community area  Min. distance
2    Jefferson Park          2.900
5       Albany Park          2.417
11      West Ridge           5.600
28         Douglas           5.017
30      Fuller Park          2.350
34  Washington Park          5.900
36         Woodlawn         11.250
44     North Center         13.867
58      West Elsdon          5.283
67         Ashburn          10.483
-----------------------
 You can print the graphs from the terminal
```

```
 .
This is the distance of the real most tensioned c
ommunity areas to the nearest police station, aft
er a stochastic reduction in the number of police
 stations
 -----------------------
             Community area  Min. distance
15            Humboldt Park         15.900
17                   Austin         14.317
18       West Garfield Park         10.200
21           North Lawndale         10.750
37              South Shore         15.200
43                 Roseland          6.867
63           West Englewood          2.517
64                Englewood          5.050
65  Greater Grand Crossing           3.767
68            Auburn Gresham         7.933
-----------------------
 You can print the graphs from the terminal
```

<u>Run Application and Simulation</u>

To run package in Linux:

1. Clone this repository to your Linux machine.
2. From inside the project directory (Pro-vision), run 'poetry install' to install dependencies.
3. Run 'poetry shell' to create a virtual environment.
4. Once the environment is running, you are ready to run the Pro-vision dashboard and simulation generator.
5. Run the Dashboard and Simulation:
    a. To run the quick version of the dashboard (uses pre-processed local files), go to 'Pro-vision/dashboard_quick' and run 'python3 app.py'.
    b. To run the full version of the dashboard (includes back-end data cleaning and api-access to fetch isochrones), go to 'Pro-vision/dashboard_full' and run 'python3 app.py'
    c. To run the simulation generator, go to the directory called 'Simulation' and run 'python3 simulation.py'

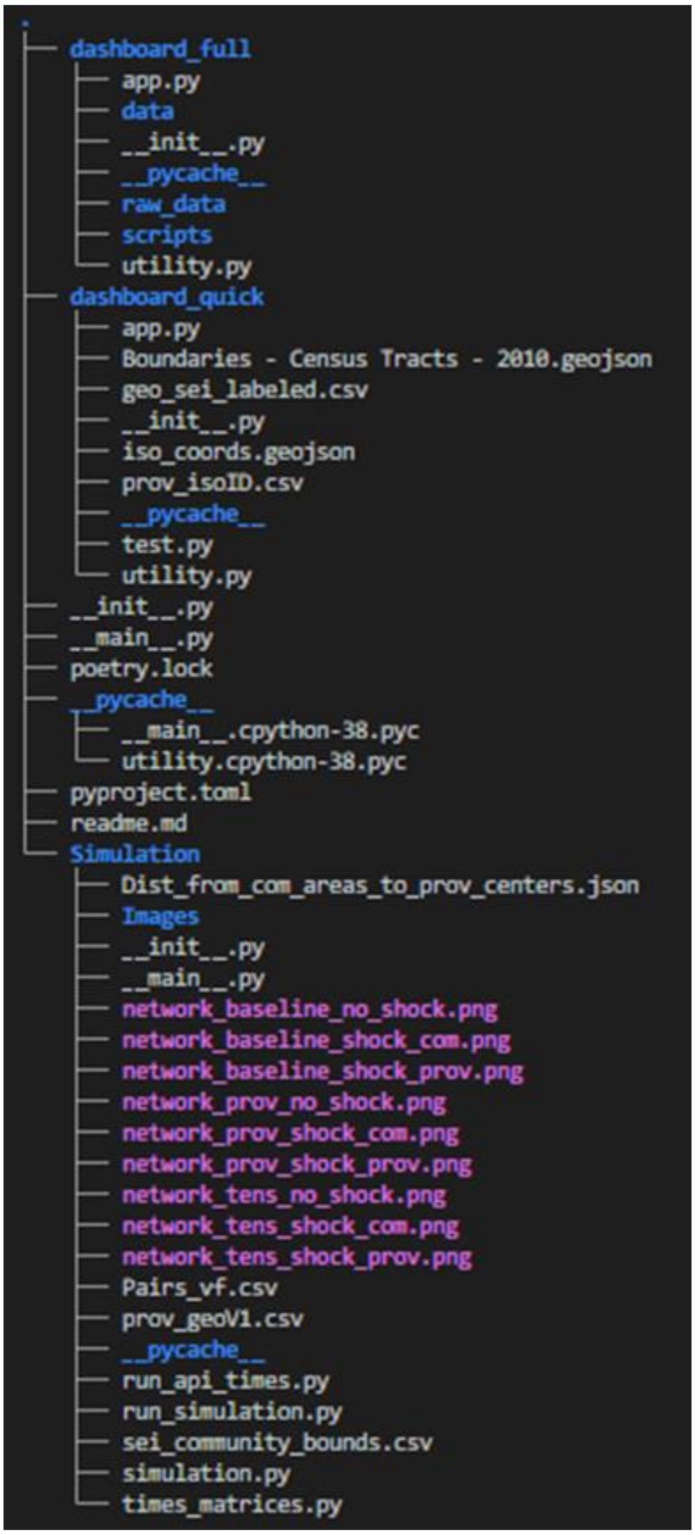
## Final Comments

The original objective of this project is to study the time-distance coverage of public provisions in Chicago, and to map it over socioeconomic indicators to identify possible patterns in vulnerabilities in the City of Chicago. Not only was this goal successfully accomplished, but we further improved our descriptive analysis with a probabilistic study of the resilience of the system, relying on Graph Theory: we introduced a Simulation module that allows the user to apply shocks in both the provision's coverage and in the degree of tension of the different areas. Although some ideas were excluded (e.g., using a Google API to locate facilities) because of the vast availability of data for the City of Chicago, we managed to target our efforts to deliver an innovative tool that can identify visual correlations that can better inform policymakers' decisions.

This project has the potential to develop more insightful tools. For instance, implementing econometric methods to translate the identified visual correlations, including input boxes to set time-distance and mode for the coverage and/or automate reports that locate and statistically describe the vulnerable areas of the City of Chicago. The analysis can be extended to other cities, using Chicago as a (very satisfactory!) coverage counterfactual.

## Appendix

Appendix A. Extended Software Structure



```
.
├── dashboard_full
│   ├── app.py
│   ├── data
│   ├── __init__.py
│   ├── __pycache__
│   ├── raw_data
│   ├── scripts
│   └── utility.py
├── dashboard_quick
│   ├── app.py
│   ├── Boundaries - Census Tracts - 2010.geojson
│   ├── geo_sei_labeled.csv
│   ├── __init__.py
│   ├── iso_coords.geojson
│   ├── prov_isoID.csv
│   ├── __pycache__
│   ├── test.py
│   └── utility.py
├── __init__.py
├── __main__.py
├── poetry.lock
├── __pycache__
│   ├── __main__.cpython-38.pyc
│   └── utility.cpython-38.pyc
├── pyproject.toml
├── readme.md
└── Simulation
    ├── Dist_from_com_areas_to_prov_centers.json
    ├── Images
    ├── __init__.py
    ├── __main__.py
    ├── network_baseline_no_shock.png
    ├── network_baseline_shock_com.png
    ├── network_baseline_shock_prov.png
    ├── network_prov_no_shock.png
    ├── network_prov_shock_com.png
    ├── network_prov_shock_prov.png
    ├── network_tens_no_shock.png
    ├── network_tens_shock_com.png
    ├── network_tens_shock_prov.png
    ├── Pairs_vf.csv
    ├── prov_geoV1.csv
    ├── __pycache__
    ├── run_api_times.py
    ├── run_simulation.py
    ├── sei_community_bounds.csv
    ├── simulation.py
    └── times_matrices.py
```