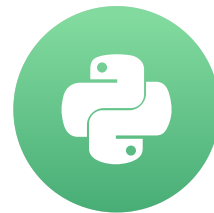


American Community Survey: Annual Change

ANALYZING US CENSUS DATA IN PYTHON

Lee Hachadoorian

Asst. Professor of Instruction, Temple
University



Census History: Counts and Samples

Full count of core demographic characteristics:

- Decennial Census 1790 - 2010+

Sample of extensive social and economic characteristics:

- Decennial Census "Long Form" (SF3) 1970 - 2000, ~15% of households
- Annual American Community Survey 2005+, ~1% of households

B25045 - Tenure by Vehicles Available by Age

Variable	Label
-----	-----
B25045001	Total
B25045002	Owner Occupied
B25045003	No Vehicle Available
B25045004	Householder 15 to 34 Years
B25045005	Householder 35 to 64 Years
B25045006	Householder 65 Years and Over
B25045007	1 or More Vehicles Available
B25045008	Householder 15 to 34 Years
B25045009	Householder 35 to 64 Years
B25045010	Householder 65 Years and Over
B25045011	Renter Occupied
B25045012	No Vehicle Available
B25045013	Householder 15 to 34 Years
B25045014	Householder 35 to 64 Years
B25045015	Householder 65 Years and Over
B25045016	1 or More Vehicles Available
B25045017	Householder 15 to 34 Years
B25045018	Householder 35 to 64 Years
B25045019	Householder 65 Years and Over

ACS Detailed Table Request - Setup

```
import requests
import pandas as pd

HOST, dataset = "https://api.census.gov/data", "acs/acs1"

get_vars = ["B25045_" + str(i + 1).zfill(3) + "E" for i in range(19)]
get_vars = ["NAME"] + get_vars

print(get_vars)
```

```
['NAME', 'B25045_001E', 'B25045_002E', 'B25045_003E', 'B25045_004E',
 'B25045_005E', 'B25045_006E', 'B25045_007E', 'B25045_008E', 'B25045_009E',
 'B25045_010E', 'B25045_011E', 'B25045_012E', 'B25045_013E', 'B25045_014E',
 'B25045_015E', 'B25045_016E', 'B25045_017E', 'B25045_018E', 'B25045_019E']
```

ACS Detailed Table Request - Setup

```
import requests
import pandas as pd

HOST, dataset = "https://api.census.gov/data", "acs/acs1"

get_vars = ["B25045_" + str(i + 1).zfill(3) + "E" for i in range(19)]
get_vars = ["NAME"] + get_vars

# print(get_vars)

predicates = {}
predicates["get"] = ",".join(get_vars)
predicates["for"] = "us:"
```

Requesting Same Variables from Multiple Years

```
# Initialize data frame collector
dfs = []

for year in range(2011, 2018):
    base_url = "/".join([HOST, str(year), dataset])
    r = requests.get(base_url, params=predicates)
    df = pd.DataFrame(columns=r.json()[0], data=r.json()[1:])
    # Add column to hold year value
    df["year"] = year
    dfs.append(df)

# Concatenate all data frames in collector
us = pd.concat(dfs)
```

Requesting Same Variables from Multiple Years

```
print(us.head())
```

```
      NAME B25045_001E B25045_002E ... B25045_019E us  year
0  United States  114991725   74264435 ...    3232812  1  2011
0  United States  115969540   74119256 ...    3447172  1  2012
0  United States  116291033   73843861 ...    3662322  1  2013
0  United States  117259427   73991995 ...    3847400  1  2014
0  United States  118208250   74506512 ...    4044430  1  2015
```

```
[5 rows x 22 columns]
```

Let's Get Some Data!

ANALYZING US CENSUS DATA IN PYTHON

Margins of Error

ANALYZING US CENSUS DATA IN PYTHON



Lee Hachadoorian

Asst. Professor of Instruction, Temple
University

Margins of Error

- Table B25045 - Tenure by Vehicles Available by Age of Householder
 - B25045_001E - **Estimate** of total occupied housing units
 - B25045_001M - **Margin of Error** of the estimate

name	B25045_001E	B25045_001M
Alabama	1,844,546	±11,416
Alaska	257,330	±3,380
Arizona	2,356,055	±12,130
Arkansas	1,127,621	±7,837

Margins of Error

```
B25045.head()
```

	NAME	B25045_001E	B25045_001M	state
0	Alabama	1844546	11416	01
1	Alaska	257330	3380	02
2	Arizona	2356055	12130	04
3	Arkansas	1127621	7837	05
4	California	12468743	22250	06

Margins of Error

```
B25045.columns = ["name", "total", "total_moe", "state"]  
B25045.head()
```

	name	total	total_moe	state
0	Alabama	1844546	11416	01
1	Alaska	257330	3380	02
2	Arizona	2356055	12130	04
3	Arkansas	1127621	7837	05
4	California	12468743	22250	06

Relative Margin of Error

Margin of Error as a Percent of the Estimate:

$$RMOE = 100 \times MOE / Estimate$$

	NAME	B25045_001E	B25045_001M	state	rmoe
0	California	13005097	17539	06	0.134863
1	Wyoming	225796	3968	56	1.757338

	NAME	B25045_001E	B25045_001M	state	county	rmoe
0	Los Angeles County	3311231	8549	06	037	0.258182
1	Sutter County, Cal	31945	907	06	101	2.839255

Margins of Error of Breakdown Columns

B25045_004E — Owner Occupied?No Vehicle Available?Householder 15 to 34 Years

	NAME	B25045_004E	B25045_004M	state	rmoe
0	California	10964	1519	06	13.854433
1	Wyoming	25	48	56	192.000000

	NAME	B25045_004E	B25045_004M	state	county	rmoe
0	Los Angeles Cou	1942	634	06	037	32.646756
1	Sutter County,	0	210	06	101	inf

Standard Errors

$$Z_{90} = 1.645$$

$$SE_x = \frac{MOE_x}{Z_{90}}$$

Statistically Significant Difference

$$Z = \frac{x_1 - x_2}{\sqrt{SE_{x_1}^2 + SE_{x_2}^2}}$$

	total	total_moe	year
4	12944178	15703	2016
4	13005097	17539	2017

```
Z_CRIT = 1.645
x1 = int(ca["total"][ca["year"] == 2017])
x2 = int(ca["total"][ca["year"] == 2016])

se_x1 = float(ca["total_moe"][ca["year"] == 2017] / Z_CRIT)
se_x2 = float(ca["total_moe"][ca["year"] == 2016] / Z_CRIT)
```


Statistically Significant Difference

$$Z = \frac{x_1 - x_2}{\sqrt{SE_{x_1}^2 + SE_{x_2}^2}}$$

	total	total_moe	year
4	12944178	15703	2016
4	13005097	17539	2017

```
Z = (x1 - x2) / _____(_____)
```

Statistically Significant Difference

$$Z = \frac{x_1 - x_2}{\sqrt{SE_{x_1}^2 + SE_{x_2}^2}}$$

	total	total_moe	year
4	12944178	15703	2016
4	13005097	17539	2017

```
Z = (x1 - x2) / numpy.sqrt(_____)
```

Statistically Significant Difference

$$Z = \frac{x_1 - x_2}{\sqrt{SE_{x_1}^2 + SE_{x_2}^2}}$$

	total	total_moe	year
4	12944178	15703	2016
4	13005097	17539	2017

```
Z = (x1 - x2) / numpy.sqrt(se_x1**2 + se_x2**2)
print(abs(Z) > Z_CRIT)
```

```
True
```

Approximating SE for Derived Estimates

$$SE_{a+b+\dots} = \sqrt{SE_a^2 + SE_b^2 + \dots}$$

$$MOE_{a+b+\dots} = Z_{90} SE_{a+b+\dots}$$

```
states["novehicle_65over"] = \
    states["owned_novehicle_65over"] + states["rented_novehicle_65over"]

states["novehicle_65over_moe"] = Z_CRIT * numpy.sqrt(\
    states["owned_novehicle_65over_moe"]**2 + \
    states["rented_novehicle_65over_moe"]**2\
)
```

Approximating SE for Derived Estimates

```
print(states[["name", "novehicle_65over", "novehicle_65over_moe"]].head())
```

	name	novehicle_65over	novehicle_65over_moe
0	Alabama	42267	4867.038791
1	Alaska	5575	1473.170747
2	Arizona	52331	6598.753623
3	Arkansas	22533	3155.583824
4	California	372772	15183.882878

Let's Practice!

ANALYZING US CENSUS DATA IN PYTHON

Basic Mapping with Geopandas

ANALYZING US CENSUS DATA IN PYTHON



Lee Hachadoorian

Asst. Professor of Instruction, Temple
University

Geospatial Data - Further Learning

- [Working with Geospatial Data in Python](#)
- [Visualizing Geospatial Data in Python](#)

Loading Geospatial Data

```
import geopandas as gpd
# Load a geospatial file
geo_state = gpd.read_file("state_computer_use.gpkg")
type(geo_state)
```

```
geopandas.geodataframe.GeoDataFrame
```

Geopandas Data Frames

```
print(geo_state.columns)
```

```
Index(['state', 'postal', 'name', 'geometry', 'total', 'has_computer',  
      'desktop_laptop', 'desktop_laptop_only', 'portable_device',  
      'portable_device_only', 'no_computer'],  
      dtype='object')
```

Geopandas Data Frames

```
print(geo_state.head())
```

```
state postal ... portable_device_only no_computer
0    06    CA ...          1052406          1263635
1    08    CO ...          148749          168639
2    11    DC ...           23554           32916
3    16    ID ...           46565           67454
4    17    IL ...          415840          640062
```

```
[5 rows x 11 columns]
```

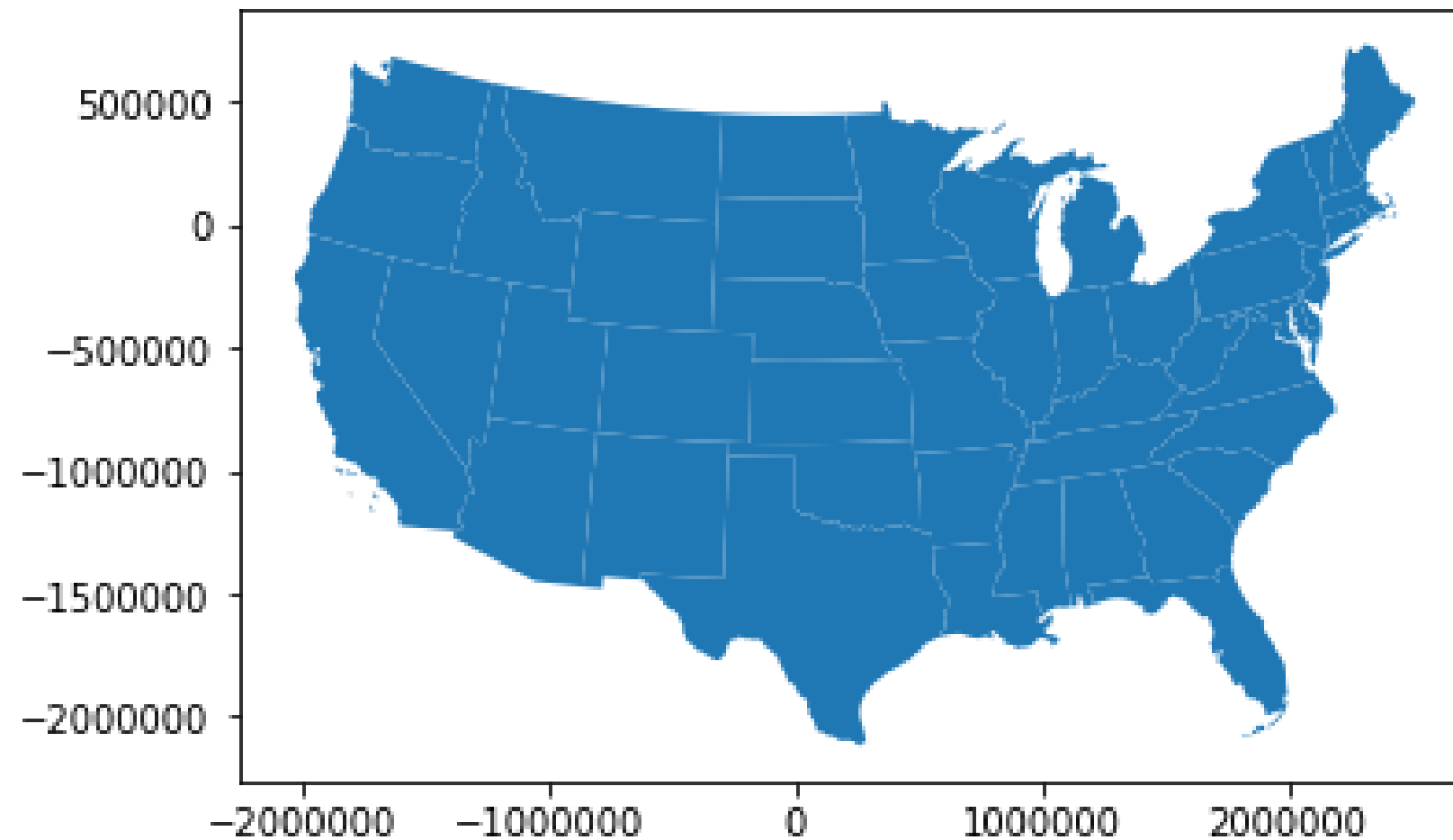
Geopandas geometry Column

```
print(geo_state["geometry"].head())
```

```
0    (POLYGON ((-1716661.572795197 -1091585.2669098...
1    (POLYGON ((-787764.8711845676 -679501.90042785...
2    (POLYGON ((1950824.825659711 -402441.066172522...
3    (POLYGON ((-1357254.591159301 77405.1771214068...
4    (POLYGON ((720437.1241238854 -496471.759394428...
Name: geometry, dtype: object
```

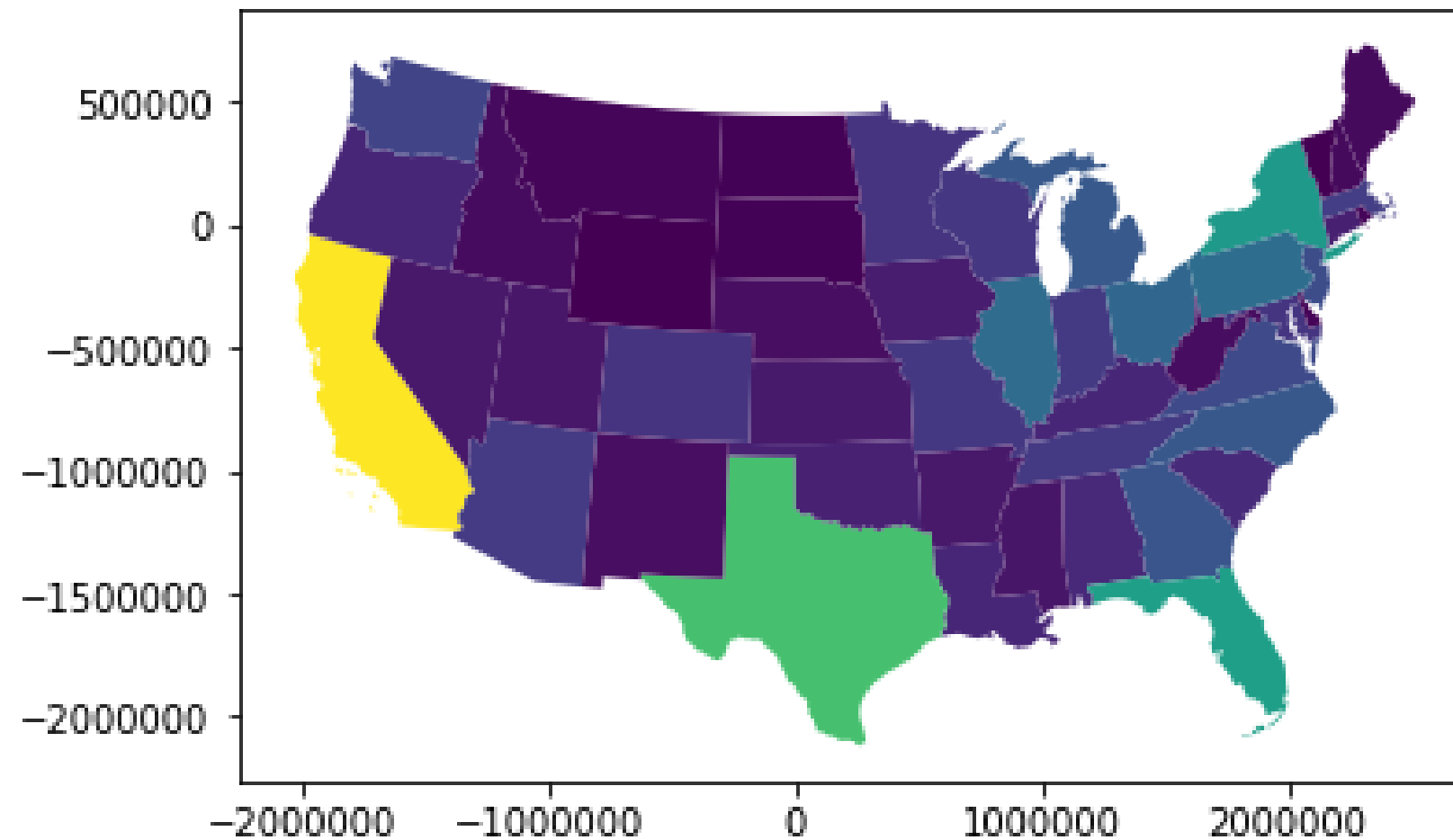
Geopandas Plotting

```
geo_state.plot()
```



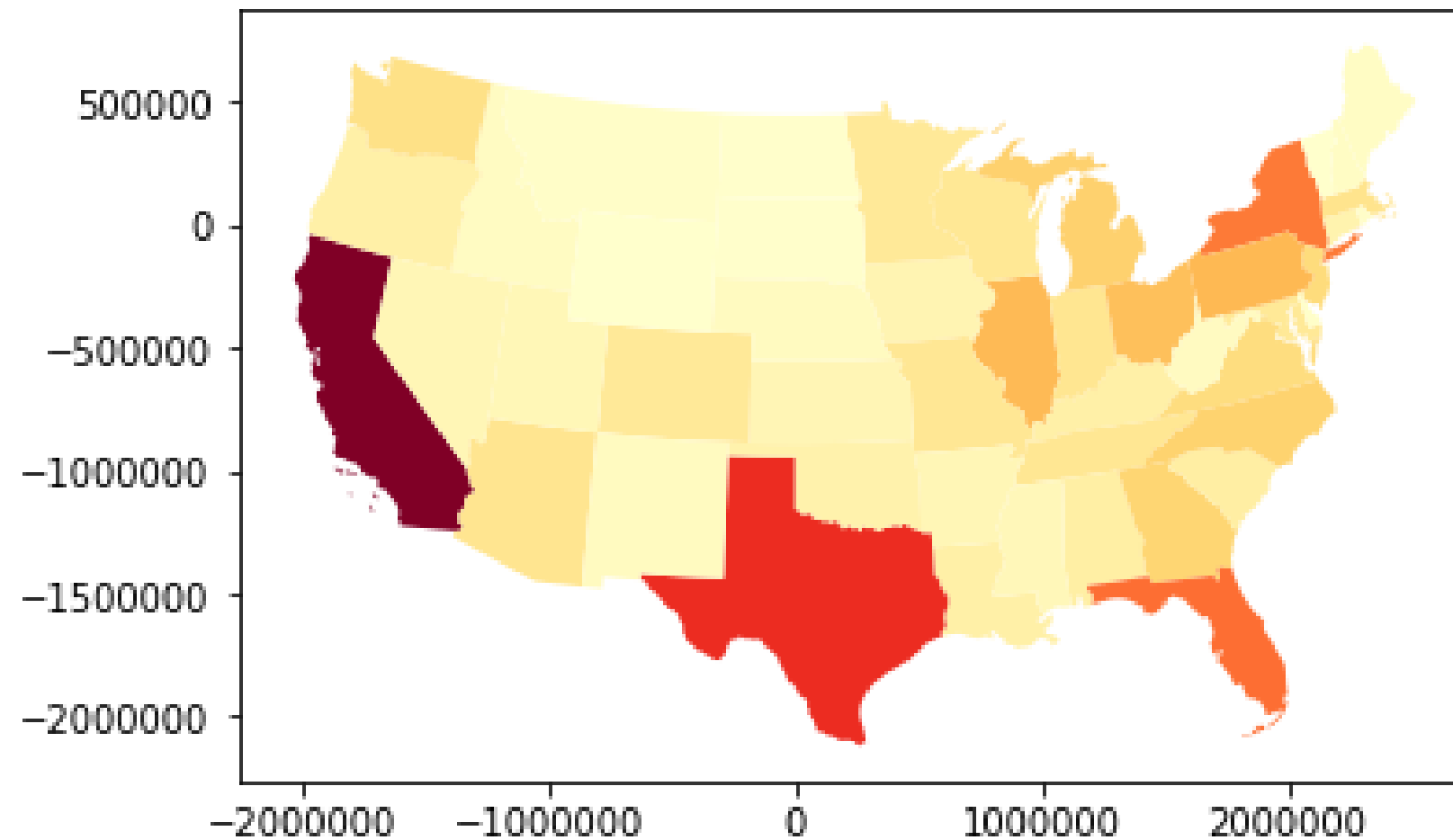
Choropleth Maps

```
geo_state.plot(column = "has_computer")
```



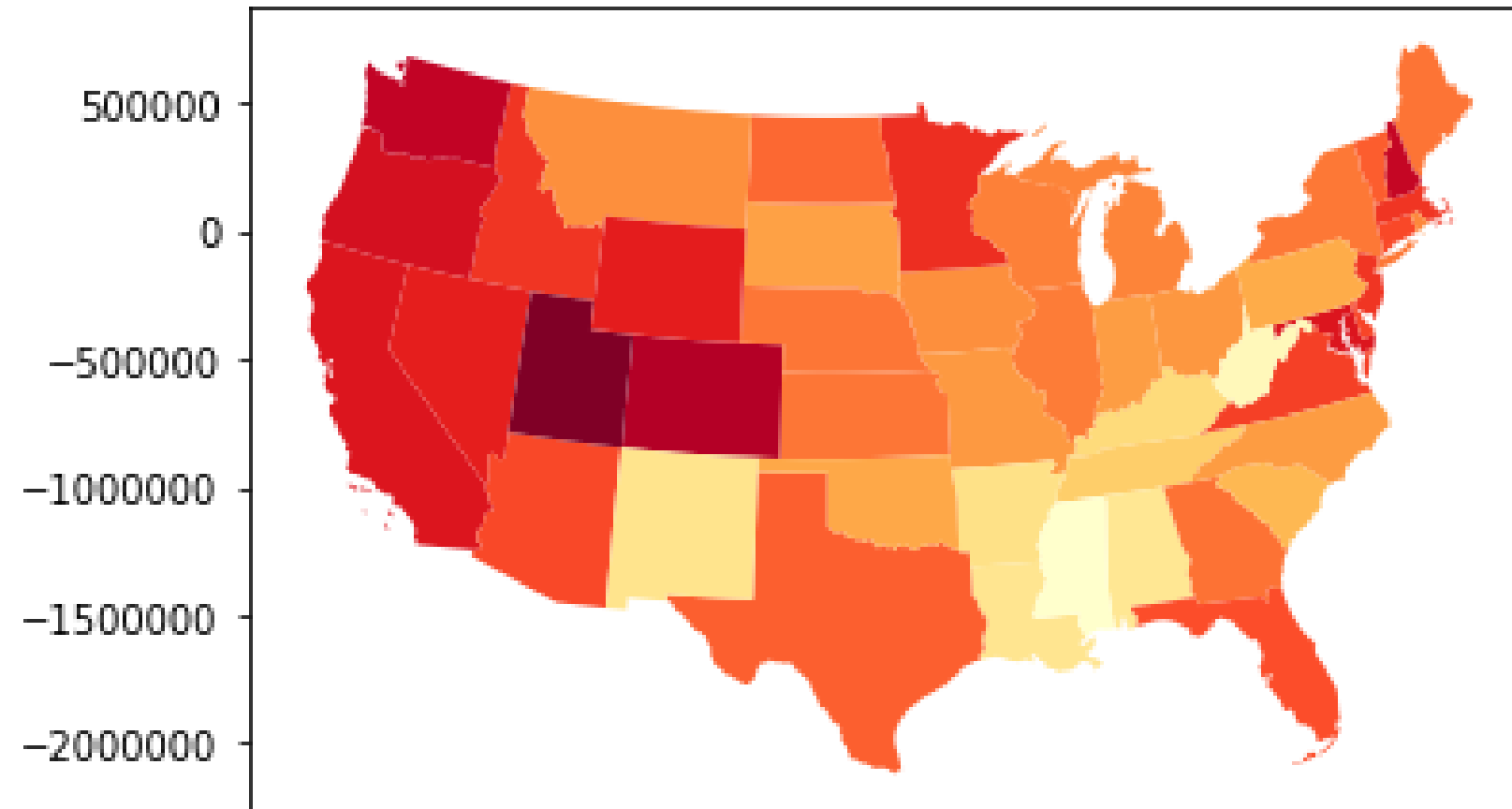
Choropleth Maps

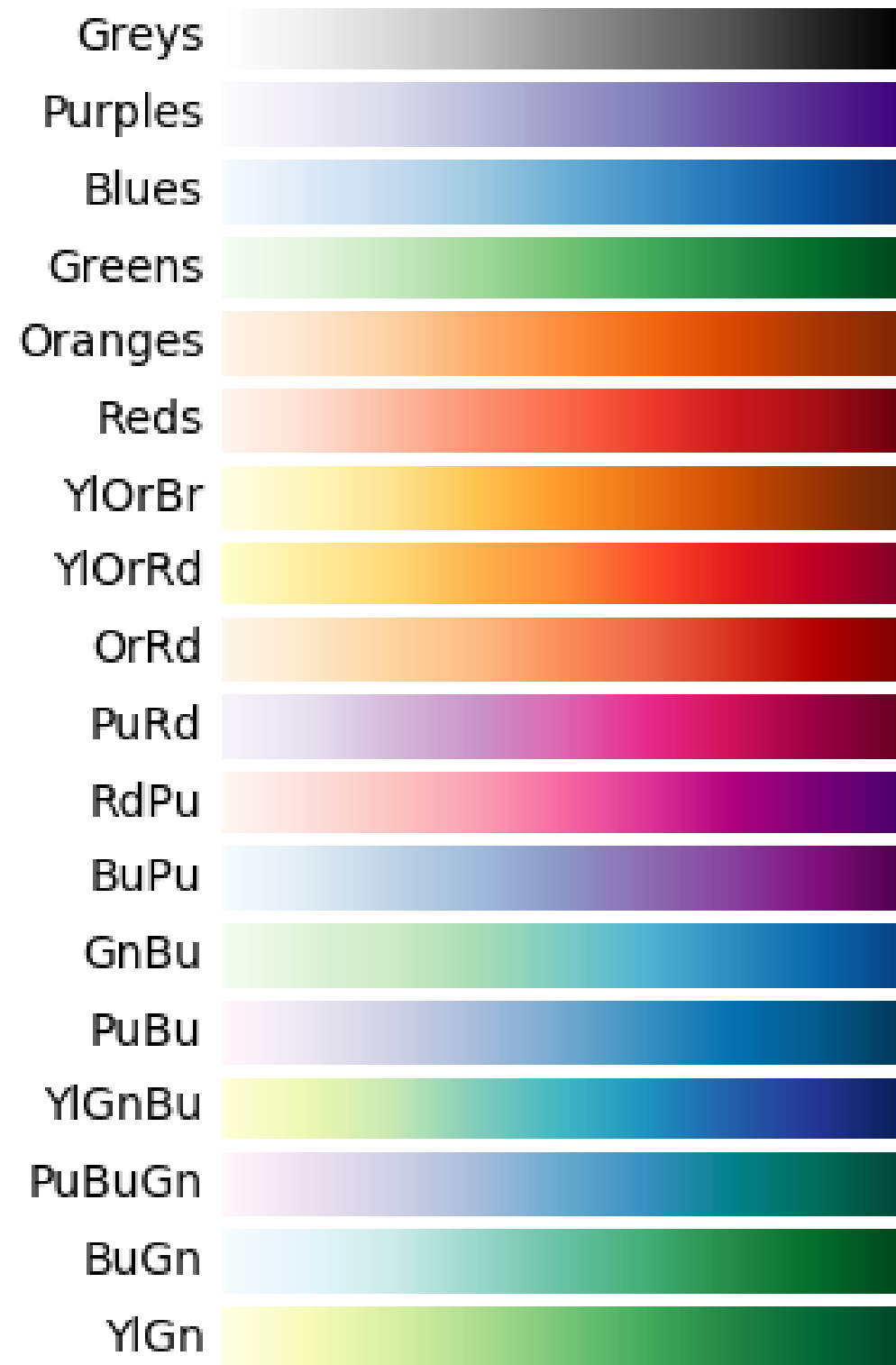
```
geo_state.plot(column = "has_computer", cmap = "YlOrRd")
```



Choropleth Maps

```
geo_state["pct_has_computer"] = 100 * geo_state["has_computer"] / geo_state["total"]  
geo_state.plot(column = "pct_has_computer", cmap = "YlOrRd")
```





Matplotlib Sequential Colormaps

<https://matplotlib.org/users/colormaps.html>

Let's practice!

ANALYZING US CENSUS DATA IN PYTHON

Neighborhood Change

ANALYZING US CENSUS DATA IN PYTHON



Lee Hachadoorian

Asst. Professor of Instruction, Temple
University

What Is Gentrification?

- Disinvestment in urban core
- Declining middle-class population and deteriorating housing stock
- Return of middle and upper-middle class households who renovate older housing stock
- Potential displacement of working class, Black, and immigrant households

Operationalizing Gentrification

- **Gentrifiable**
 - **Low median income:** Median household income (MHI) below metro area median
 - **Slow housing construction:** New build in previous two decades less than metro area
- **Gentrifying**
 - **Increasing educational attainment:** % with BA or higher is growing faster than metropolitan area
 - **Increasing house value:** Median house value greater than previous time period (adjusted for inflation)

¹ Freeman, Lance. 2005. "Displacement or Succession?: Residential Mobility in Gentrifying Neighborhoods." Urban Affairs Review 40 (4): 463–91.

Data Sources

- **2000 Census of Population and Housing - Summary File 3**
 - **P53:** Median Household Income in 1999 (Dollars)
 - **H34:** Year Structure Built
 - **P37:** Sex by Educational Attainment for the Population 25 Years and Over
 - **H85:** Median Value (Dollars) for All Owner-Occupied Housing Units
- **American Community Survey 5-Year Data (2008-2012)**
 - **B15003:** Educational Attainment for the Population 25 Years and Over
 - **B25077:** Median Value (Dollars) - Owner-occupied housing units

bk_2000: Brooklyn Census Tracts 2000

state	State FIPS
county	County FIPS
tract	Tract FIPS
geometry	Geometry column
mhi	Median Household Income (tract)
mhi_msa	Median Household Income (NY Metro Area)
median_value	Median House Value (tract)
median_value_msa	Median House Value(NY Metro Area)
pct_recent_build	Percent of housing built between 1980 and 1999 (tract)
pct_recent_build_msa	Percent of housing built between 1980 and 1999 (NY Metro Area)
pct_ba	Percentage of 25 year olds with BA or higher (tract)
pct_ba_msa	Percentage of 25 year olds with BA or higher(NY Metro Area)

Boolean Criteria

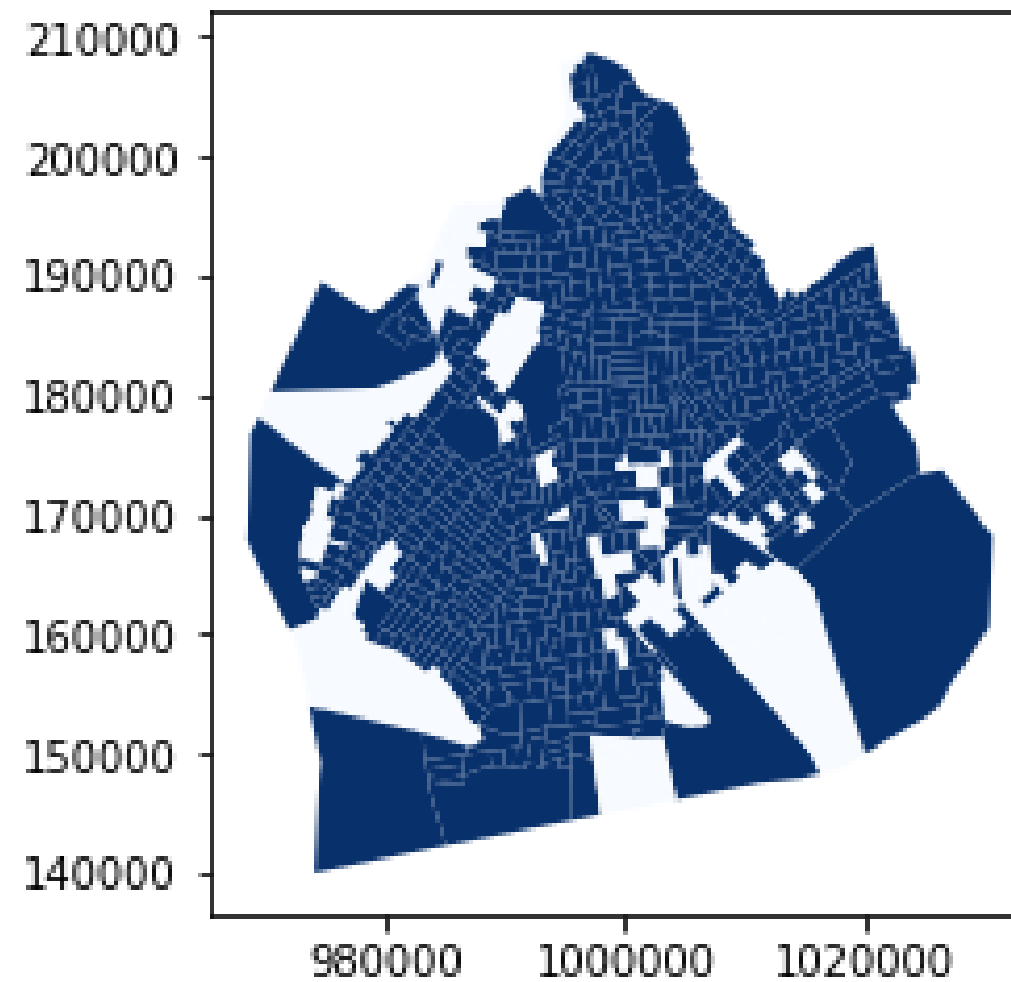
```
bk_2000[["tract", "mhi", "mhi_msa"]].head()
```

	tract	mhi	mhi_msa
0	051200	31393	50795
1	051300	30000	50795
2	051400	32103	50795
3	051500	36107	50795
4	051600	25148	50795

```
bk_2000["low_mhi"] = bk_2000["mhi"] < bk_2000["mhi_msa"]
```


Mapping Low Income Tracts

```
bk_2000.plot(column = "low_mhi", cmap = "Blues")
```



Let's practice!

ANALYZING US CENSUS DATA IN PYTHON