greg_nmbrlang

May 5, 2022

1 Appendix: GREG Database Wrangling

This notebook is included separately, because it contains the code used to transform the GREG dataset into values in a suitable format to swap in for the withheld WLMS data.

[1]:
```python
import pandas as pd
import numpy as np
import geopandas as gpd

import matplotlib.pyplot as plt
import seaborn as sns
```

```
/opt/homebrew/lib/python3.9/site-packages/geopandas/_compat.py:111: UserWarning:
The Shapely GEOS version (3.10.2-CAPI-1.16.0) is incompatible with the GEOS
version PyGEOS was compiled with (3.10.1-CAPI-1.16.0). Conversions between both
will be slow.
  warnings.warn(
```

1.1 Import Shapefiles

[2]:
```python
# Linux is a true UNIX (unlike some other OSs), therefore case-sensitive! :)
virtual = gpd.read_file('data_raw/Virtual_country')
virtual.head()
```

[2]:
```
   uniq_cnt25  point5_id   pop95  maize  pasture  suit_new  sorghum  allcrops  \
0          39   247867.0  0.0002    0.0      0.0       0.0      0.0       0.0
1          40   247868.0  0.0002    0.0      0.0       0.0      0.0       0.0
2          40   247869.0  0.0003    0.0      0.0       0.0      0.0       0.0
3          40   247870.0  0.0003    0.0      0.0       0.0      0.0       0.0
4          40   247871.0  0.0003    0.0      0.0       0.0      0.0       0.0

                                            geometry
0  POLYGON ((-86.00000 82.00000, -86.50000 82.000…
1  POLYGON ((-85.50000 82.00000, -86.00000 82.000…
2  POLYGON ((-85.00000 82.00000, -85.50000 82.000…
3  POLYGON ((-84.50000 82.00000, -85.00000 82.000…
4  POLYGON ((-84.00000 82.00000, -84.50000 82.000…
```

[3]:
```
greg = gpd.read_file('greg')
greg.head()
```

[3]:
```
  FIPS_CNTRY  GROUP1  GROUP2  GROUP3  G1ID  G2ID  G3ID  \
0         AA      12       0       0   312     0     0
1         AC      16       0       0   354     0     0
2         AF      33      53       0   117   202     0
3         AF      24      34       0   898    12     0
4         AF      34      41       0    12  1051     0

                                        G1SHORTNAM G2SHORTNAM G3SHORTNAM  \
0                                Curaçao Islanders       None       None
1  English-speaking population of the Lesser Anti…       None       None
2                                           Baloch     Brahui       None
3                                         Persians    Afghans       None
4                                          Afghans     Tajiks       None

                                         G1LONGNAM  \
0                                Curaçao Islanders
1  English-speaking population of the Lesser Anti…
2                                Baloch (Baluchis)
3                                         Persians
4                      Afghans (Pushtuns, Pathans)

                     G2LONGNAM G3LONGNAM  FeatureID          AREA  COW  \
0                         None      None          0  2.007795e+08    0
1                         None      None          1  5.398570e+08   58
2                       Brahui      None          2  1.189781e+10  700
3  Afghans (Pushtuns, Pathans)      None          3  1.653610e+09  700
4            Tajiks (Tadzhiks)      None          4  3.251011e+09  700

                                          geometry
0  POLYGON ((-69.88223 12.41111, -69.94695 12.436…
1  MULTIPOLYGON (((-61.73889 17.54055, -61.75195 …
2  POLYGON ((64.03937 30.02453, 64.03937 30.11267…
3  POLYGON ((61.75456 30.78628, 61.75833 30.79028…
4  POLYGON ((61.62285 31.39536, 61.64841 31.46713…
```

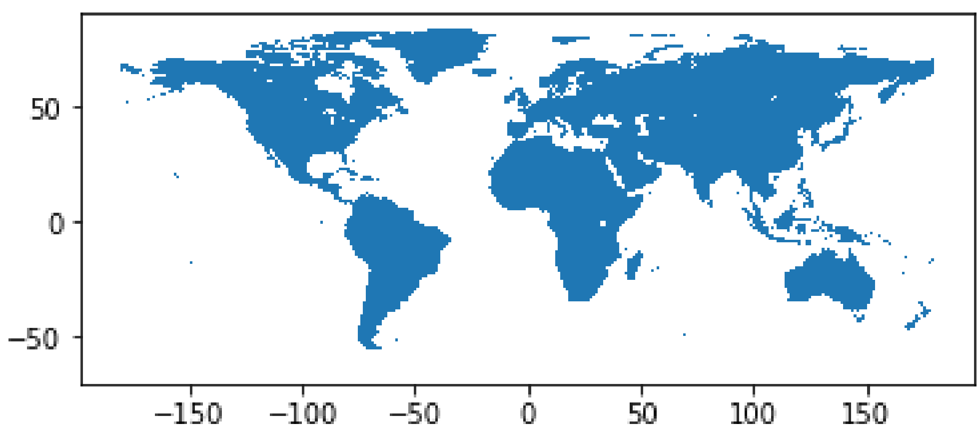
[4]:
```
virtual.crs, greg.crs
```

[4]:
```
(<Geographic 2D CRS: EPSG:4326>
 Name: WGS 84
 Axis Info [ellipsoidal]:
 - Lat[north]: Geodetic latitude (degree)
 - Lon[east]: Geodetic longitude (degree)
 Area of Use:
 - name: World.
```

```
- bounds: (-180.0, -90.0, 180.0, 90.0)
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich,
<Geographic 2D CRS: EPSG:4326>
Name: WGS 84
Axis Info [ellipsoidal]:
- Lat[north]: Geodetic latitude (degree)
- Lon[east]: Geodetic longitude (degree)
Area of Use:
- name: World.
- bounds: (-180.0, -90.0, 180.0, 90.0)
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich)
```

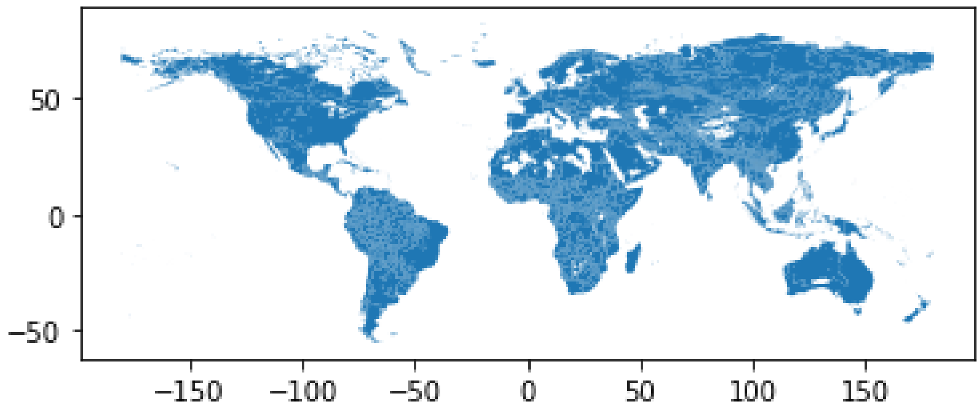
[5]:
```
virtual.plot()
```

[5]: <AxesSubplot:>



[6]:
```
greg.plot()
```

[6]: <AxesSubplot:>



1.2 Transform GREG

The original GREG format is a number of regions, each of which has up to three ethnic groups attached to it. Ethnic groups may also be attached to different regions. This code chunk melts, and then dissolves, the original `greg` dataset, such that we have one entry per ethnic group.

[7]:
```
melted = pd.melt(greg, id_vars = ['geometry'], value_vars = ['G1SHORTNAM',␣
  ↪'G2SHORTNAM', 'G3SHORTNAM'], value_name = 'SHORTNAM')
ethnicGroups = melted[melted['SHORTNAM'].notna()].drop('variable', axis = 1).
  ↪dissolve(by = 'SHORTNAM', aggfunc = 'first', as_index = False)
ethnicGroups
```

[7]:
```
        SHORTNAM                                           geometry
0     Abazinians  MULTIPOLYGON (((41.83519 44.08370, 41.86445 44…
1         Abkhaz  MULTIPOLYGON (((41.73878 42.62086, 41.71329 42…
2       Achaguas  MULTIPOLYGON (((-74.02123 2.16973, -73.98634 2…
3         Achang  POLYGON ((97.84312 24.33767, 97.84467 24.36087…
4       Achinese  MULTIPOLYGON (((97.81446 2.77691, 97.86672 2.7…
..           …                                                  …
923       Zagawa  MULTIPOLYGON (((25.88538 14.53904, 25.83321 14…
924    Zakhchins  POLYGON ((91.54557 47.36713, 91.54557 47.43751…
925     Zapotecs  POLYGON ((-94.96082 16.37316, -95.03084 16.322…
926        Zoque  MULTIPOLYGON (((-93.18895 16.87464, -93.13737 …
927        Zulus  POLYGON ((31.11778 -29.67945, 31.00972 -29.872…

[928 rows x 2 columns]
```

1.3 Perform Intersection

This cell intersects the imported dataset of cells with the dataset of ethnic groups, derived from GREG.

```
[8]: joined = gpd.overlay(virtual, ethnicGroups, how = 'intersection')
joined.head()
```

```
[8]:    uniq_cnt25  point5_id   pop95  maize  pasture  suit_new  sorghum  allcrops  \
0         211   247281.0  0.0301    0.0      0.0    0.0000      0.0       0.0
1         211   247282.0  0.0300    0.0      0.0    0.0000      0.0       0.0
2         335   241416.0  0.0271    0.0      0.0    0.0001      0.0       0.0
3         335   242134.0  0.0195    0.0      0.0    0.0001      0.0       0.0
4         335   242135.0  0.0330    0.0      0.0    0.0001      0.0       0.0

  SHORTNAM                                           geometry
0  Eskimos  MULTIPOLYGON (((-19.00000 81.71801, -19.14417 …
1  Eskimos  MULTIPOLYGON (((-19.00000 81.80707, -18.99083 …
2  Eskimos  POLYGON ((-72.00000 78.00000, -71.87679 78.000…
3  Eskimos  MULTIPOLYGON (((-73.00000 78.17449, -72.99834 …
4  Eskimos  POLYGON ((-72.34038 78.00000, -72.34695 78.003…
```

1.4 Coverage

These cells reduce each virtual country to *only contain cells in which the cell is completely covered by an ethnic group from GREG*, similar to our interpretation of the procedure described in Michalopolous.

First, we calculate the "area" of each small cell after it has been intersected with the transformed GREG dataset. Then we compare this area to the area of the full cell, and equivalent areas indicate that the cell is completely covered.

```
[9]: dissolved = joined[['point5_id', 'geometry']].dissolve('point5_id')
areasCell = dissolved.area.to_frame().rename(columns = {0: 'overlay'})
areasCell['full'] = virtual.set_index('point5_id').area
areasCell['complete'] = np.isclose(areasCell['overlay'], areasCell['full'])
areasCell
```

```
/opt/homebrew/lib/python3.9/site-packages/pygeos/set_operations.py:388:
RuntimeWarning: divide by zero encountered in unary_union
  result = lib.unary_union(collections, **kwargs)
/var/folders/l7/_yl1rg512jv095gql7v0v5r00000gn/T/ipykernel_1224/310672282.py:2:
UserWarning: Geometry is in a geographic CRS. Results from 'area' are likely
incorrect. Use 'GeoSeries.to_crs()' to re-project geometries to a projected CRS
before this operation.

  areasCell = dissolved.area.to_frame().rename(columns = {0: 'overlay'})
/var/folders/l7/_yl1rg512jv095gql7v0v5r00000gn/T/ipykernel_1224/310672282.py:3:
UserWarning: Geometry is in a geographic CRS. Results from 'area' are likely
incorrect. Use 'GeoSeries.to_crs()' to re-project geometries to a projected CRS
before this operation.

  areasCell['full'] = virtual.set_index('point5_id').area
```

[9]:
```
             overlay  full  complete
point5_id
49903.0     0.040962  0.25     False
49904.0     0.111268  0.25     False
49905.0     0.112302  0.25     False
49906.0     0.006557  0.25     False
50621.0     0.006873  0.25     False
…                …     …         …
242136.0    0.185729  0.25     False
242137.0    0.013326  0.25     False
242857.0    0.033278  0.25     False
247281.0    0.013842  0.25     False
247282.0    0.060065  0.25     False

[58073 rows x 3 columns]
```

This cell merges the overlay dataset calculated earlier with the coverage dataset, to determine whether each cell-ethnic group combination is of a cell with complete coverage.

[10]:
```
joinedCoverage = joined.merge(areasCell, left_on = 'point5_id', right_index =␣
  ↪True)
joinedCoverage.head()
```

[10]:
```
   uniq_cnt25  point5_id   pop95  maize  pasture  suit_new  sorghum  allcrops  \
0         211   247281.0  0.0301    0.0      0.0    0.0000      0.0       0.0
1         211   247282.0  0.0300    0.0      0.0    0.0000      0.0       0.0
2         335   241416.0  0.0271    0.0      0.0    0.0001      0.0       0.0
3         335   242134.0  0.0195    0.0      0.0    0.0001      0.0       0.0
4         335   242135.0  0.0330    0.0      0.0    0.0001      0.0       0.0

  SHORTNAM                                           geometry   overlay  full  \
0  Eskimos  MULTIPOLYGON (((-19.00000 81.71801, -19.14417 …  0.013842  0.25
1  Eskimos  MULTIPOLYGON (((-19.00000 81.80707, -18.99083 …  0.060065  0.25
2  Eskimos  POLYGON ((-72.00000 78.00000, -71.87679 78.000…  0.006730  0.25
3  Eskimos  MULTIPOLYGON (((-73.00000 78.17449, -72.99834 …  0.096789  0.25
4  Eskimos  POLYGON ((-72.34038 78.00000, -72.34695 78.003…  0.240012  0.25

   complete
0     False
1     False
2     False
3     False
4     False
```

Finally, we group by the virtual country ID, and count the number of unique ethnic groups (`SHORTNAM`), along with the number of complete cells (`point5_id`).

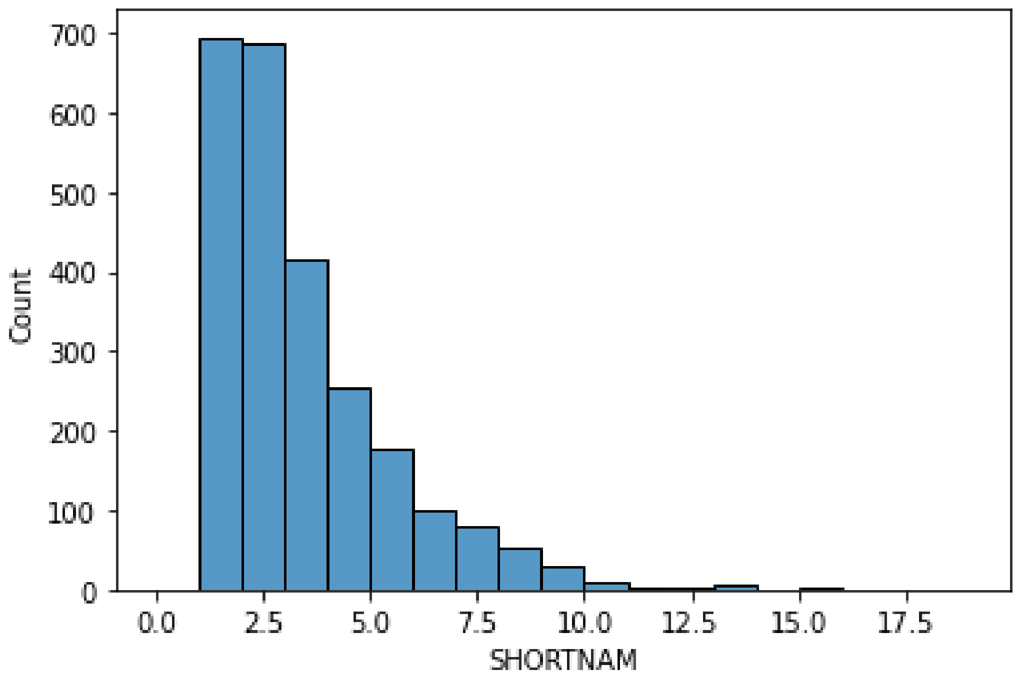
[11]:
```
countries = joinedCoverage[joinedCoverage['complete'] == True].
 ↪groupby('uniq_cnt25')[['point5_id', 'SHORTNAM']].nunique()
countries.to_csv('greg.csv')
```

[12]:
```
len(countries)
```

[12]: 2521

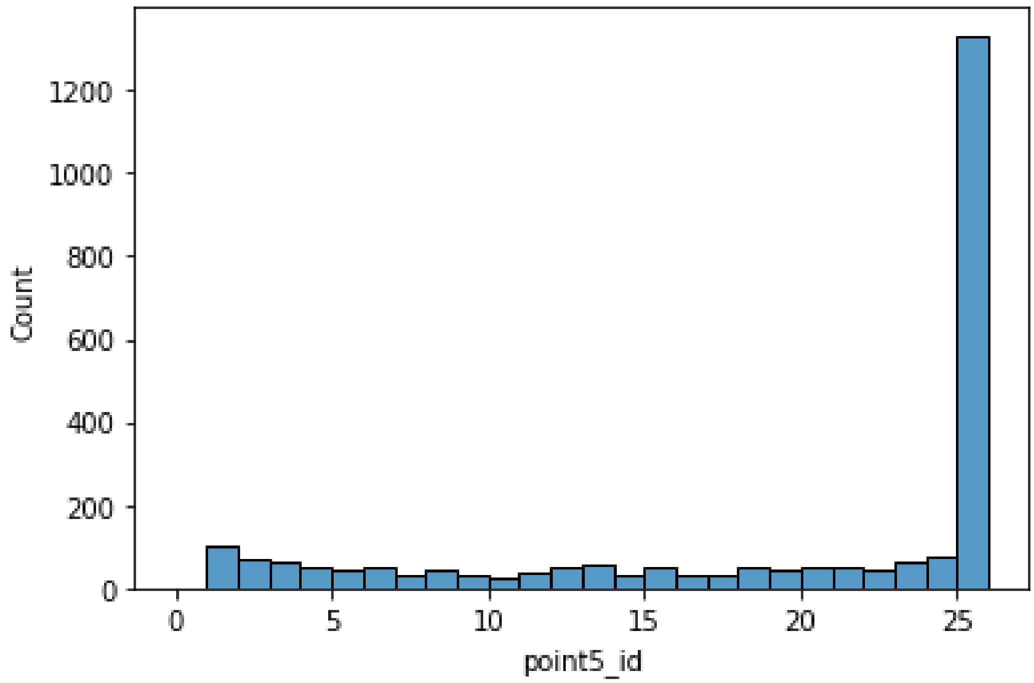
[13]:
```
sns.histplot(x = countries['SHORTNAM'], bins = np.arange(0, 20))
```

[13]: <AxesSubplot:xlabel='SHORTNAM', ylabel='Count'>



[14]:
```
sns.histplot(x = countries['point5_id'], bins = np.arange(27))
```

[14]: <AxesSubplot:xlabel='point5_id', ylabel='Count'>



1.5 Comparison to WLMS

When calculating number of ethnic groups per virtual country, we obtained 2521 countries with full coverage in at least one of its 25 cells. 1857 of these countries are included in the dataset derived from WLMS provided in the data download. 31 of the countries included in the data downloaded are *not* included in the 2521 countries we obtained.

[15]:
```
df = pd.read_stata('data_raw/Tables4-7b.dta')
df['uniq_cnt25'] = df['uniq_cnt25'].astype(int)
df.head()
```

[15]:
```
   uniq_cnt25 wbcode_centroid  clim_suit  soil_suit    sdclim    sdsoil  \
0         687             RUS   0.001554   0.508703  0.000107  0.017684
1         690             RUS   0.002942   0.537747  0.000533  0.097307
2         696             RUS   0.005517   0.567190  0.000540  0.126735
3         697             RUS   0.005504   0.513444  0.000640  0.097434
4         698             RUS   0.005641   0.473895  0.000487  0.058877

   sea_dist     emean    precav     tempav  …  tropics  erange_gecon  \
0  1.245290  0.134700  0.033053 -12.296000  …      0.0         0.064
1  1.535823  0.069462  0.029442 -12.593077  …      0.0         0.041
2  2.094293  0.059067  0.019826 -13.622666  …      0.0         0.065
3  2.156625  0.078250  0.018673 -13.852500  …      0.0         0.088
4  2.144950  0.031600  0.017096 -13.879000  …      0.0         0.066
```

```
   lnareakm2  lnmean_pop95  indigenous  diffemean  diffprecav  difftempav  \
0   0.615041    -10.844584        0.96   0.002367    0.000185   -0.025999
1   1.679177     -9.123105        0.96  -0.029205    0.001050    0.418035
2   1.922477    -10.733601        0.96  -0.019933    0.000729    0.396222
3   1.240280    -10.733601        0.96  -0.018306    0.000229    0.249722
4   1.554200    -10.233104        0.96  -0.021400   -0.000535   -0.084714

    diffavg  overlap
0 -0.000025      0.0
1  0.000062      0.0
2  0.000227      0.0
3  0.000077      0.0
4 -0.000057      0.0

[5 rows x 35 columns]
```

[16]:
```
len(df)
```

[16]: 1888

[17]:
```
countries.index.isin(df['uniq_cnt25']).sum()
```

[17]: 1857

[18]:
```
df['uniq_cnt25'].isin(countries.index).sum()
```

[18]: 1857

[ ]:
```

```