Replication of Spielman et al.'s 2020 Evaluation of the Social Vulnerability Index: Analysis Plan

Authors

- Liam Smith, lwsmith@middlebury.edu, @Liam-W-Smith, Middlebury College
- Joseph Holler*, josephh@middlebury.edu , @josephholler, ORCID link, Middlebury College

* Corresponding author

Abstract

This study is a *replication* of:

> Spielman, S. E., Tuccillo, J., Folch, D. C., Schweikert, A., Davies, R., Wood, N., & Tate, E. (2020). Evaluating Social Vulnerability Indicators: Criteria and their Application to the Social Vulnerability Index. Natural Hazards, 100(1), 417–436. https://doi.org/10.1007/s11069-019-03820-z

The Spielman et al. (2020) paper is in turn a replication of:

> Cutter, S. L., Boruff, B. J., & Shirley, W. L. (2003). Social vulnerability to environmental hazards. Social Science Quarterly, 84(2), 242–261. https://doi.org/10.1111/1540-6237.8402002

Spielman and others (2020) evaluate the internal consistency and construct validity of the Cutter, Boruff and Shirley (2003) Social Vulnerability Index (SoVI) through replications with varying geographic extents. First, they reproduce a national SoVI model and validate it against an SPSS procedure provided by the original research group (Hazards Vulnerability Research Institute at University of South Carolina). The original SoVI uses 42 independent z-score normalized variables from the U.S. Census, reduces the data to factors using Principal Components Analysis, selects the first eleven factors, inverts factors with inverse relationships to social vulnerability, and sums the factors together to produce a SoVI score. The reproduced SoVI model was slightly different than the original model due to changes in U.S. Census data, using only 28 variables.

Spielman et al. first reproduce SoVI with a nationwide extent. Then they replicate the SoVI by modifying the geographic extent and recalculating SoVI for each of ten Federal Emergency Management Agency (FEMA) regions and for a single state or cluster of states within each of the ten regions, resulting in 21 total indices. Internal consistency is assessed by calculating the Spearman's rank correlation coefficient of the SoVI score for counties in the state model compared to the FEMA region model and national model. Construct validity is assessed by summing the loadings for each input variable across the PCA factors in each model and calculating the variable's sign (positive/negative) and the rank of the variable's total loading compared to the other variables. The variables are summarized across all 21 versions of the SoVI model with regard to the number of times the sign is different from the national model and with regard to the minimum, maximum, and mean of their ranks.

Planned Deviation for Replication:

In this replication study, we extend Spielman et al.'s work by addressing the robustness of SoVI in the temporal dimension. Specifically, we construct a SoVI model using 5-year ACS data published each year between 2012 and 2021, resulting in 10 SoVI models altogether. We assess internal consistency by modelling each county's trend in SoVI scores over time. We construct a linear regression model for every county in our spatial extent, using time as our independent variable and z-score standardized SoVI scores as our dependent variable. Predicting SoVI scores using the year allows us to control for linear changes in social vulnerability over time. We calculate summary statistics and create a map of standard errors in order to interrogate SoVI's robustness over time.

We investigate theoretical consistency by summing the loadings for each input variable across the PCA factors in each model and calculating the variables sign (positive/negative) and the rank of the variable's total loading compared to the other variables. Since we vary time rather than space, we use a rank chart to display our results rather than a table of summary statistics like Spielman et al.

Parts of the code in this Jupyter notebook report are adapted from Spielman et al.'s GitHub repository. The original study states the intended open source permissions in the acknowledgements: "To facilitate advances to current practice and to allow replication of our results, all of the code and data used in this analysis is open source and available at (https://github.com/geoss/sovi-validity). Funding was provided by the US National Science Foundation (Award No. 1333271) and the U.S. Geological Survey Land Change Science Program."

Study Metadata

- `Key words`: Social vulnerability, social indicators, Principal Component Analysis, reproducibility
- `Subject`: Social and Behavioral Sciences: Geography: Human Geography
- `Date created`: June 19, 2023
- `Date modified`: August 22, 2023
- `Spatial Coverage`: United States, excluding Puerto Rico
- `Spatial Resolution`: Counties and county equivalents
- `Spatial Reference System`: EPSG:4269
- `Temporal Coverage`: 2008-2021 (data published in years 2012-2021)
- `Temporal Resolution`: 5-year estimates, compiled annually
- `Funding Name`: NSF Division of Behavioral and Cognitive Sciences
- `Funding Title`: Transforming Theory and STEM Education Through Reproductions and Replications in the Geographical Sciences
- `Award info URI`: https://www.nsf.gov/awardsearch/showAward?AWD_ID=2049837
- `Award number`: 2049837

Original study spatio-temporal metadata

- `Spatial Coverage`: United States, excluding Puerto Rico
- `Spatial Resolution`: Counties and county equivalents
- `Spatial Reference System`: EPSG:4269
- `Temporal Coverage`: 2008 - 2012 (data is the 2012 5-year ACS)
- `Temporal Resolution`: Estimated values are averaged from measurements over five years. The data in this study does not address change over time.

Study design

In our previous work, we computationally reproduced Spielman et al.'s original paper using the code provided in their Github repository (https://github.com/geoss/sovi-validity). A report on our reproduction is available online (https://osf.io/4s62b), and the code is included in our GitHub repository (https://github.com/HEGSRR/RPl-Spielman-2020).

The original paper was a replication study testing the sensitivity of SoVI to changes in geographic extent. The SoVI is a descriptive empirical model based on social vulnerability and place theory. Spielman et al. addressed the following null hypotheses in their work:

> OR-H1: SoVI is internally inconsistent.

To address this hypothesis, Spielman et al. illustrated that SoVI is not robust to changes in geographic extent by calculating SoVI scores for ten selected states or groups of states on three geographic extents: national, FEMA region, and state(s). The counties within the state(s) of interest were then selected and ranked according to their SoVI score. OR-H1 was tested by calculating Spearman's rank correlation between the state and FEMA region models and between the state and national models.

> OR-H2: SoVI is theoretically inconsistent.

To address this hypothesis, Spielman et al. used the same SoVI models as described under OR-H1. For each model, they summed all of the PCA factors together to determine the net influence of each variable in each model. Then they recorded the signs of each variable and calculated the number of deviations of the ten state and FEMA region models from the national model. They also ranked the variables by absolute value for each model and calculated summary statistics regarding the distribution of ranks for each variable amongst all models. Spielman et al. did not use a particular statistical method to test OR-H2, but illustrated substantial disagreements between variable rankings and signs amongst the 21 SoVI models.

In our replication, we begin with the same null hypotheses as Spielman et al., but we will test those hypotheses by varying the temporal extent rather than spatial extent.

RPl-H1: SoVI is internally inconsistent.

To address this hypothesis, we will calculate SoVI scores for the entire nation (excluding Puerto Rico) using 5-year ACS data published each year between 2012 and 2021. In Spielman et al.'s analysis, we expect rankings in a given subregion to be identical regardless of the spatial extent, because the attributes in the area and the factors causing vulnerability remain constant. This is no longer true when we vary time, because we expect some areas to become more or less vulnerable over time. Since we do not expect rankings to remain constant, Spearman's rank correlation coefficient is no longer an appropriate statistical test. Instead, we seek to analyze consistency of SoVI scores while controlling for change over time. To do this, we construct a linear regression model for every county in our spatial extent using time as our independent variable and z-score standardized SoVI scores as our dependent variable. We calculate summary statistics and create a map of standard errors of our regression coefficient in order to interrogate SoVI's robustness over time.

RPl-H2: SoVI is theoretically inconsistent.

To address this hypothesis, we will use the same SoVI models as described under RPl-H1. Like Spielman et al., we sum all of the PCA components together to determine the net influence of each variable in each model. We conduct an analogous analysis to Spielman et al.'s, calculating each variable's sign (positive/negative) and rank compared to the other variables. Since we address change over time, we present our results in the form of a rank chart rather than a table of summary statistics. Furthermore, rather than ranking the magnitude of coefficients, we split each model into positive and negative coefficients, and rank them separately. This allows us to present information about sign and magnitude of coefficients in the same graph.

Materials and procedure

Computational environment

From the reproduction study, our environment consisted of Python 3.9.16 and the software packages listed in requirements.txt. We are working in the same software environment for this replication and will add additional software packages as needed for our analysis, including statsmodels for linear regression.

Among the most important packages for this analysis are pygris for pulling census data directly into Python, pandas and geopandas for working with data tables and geospatial data, NumPy for linear algebra functions, and statsmodels for linear regression.

```
# Import modules, define directories
import pygris
import pandas as pd
import geopandas as gpd
from pygris.data import get_census
from pygris import counties
from pyhere import here
import numpy as np
import libpysal as lps
import lxml
import tabulate
from scipy.stats import spearmanr
from scipy.stats.mstats import zscore as ZSCORE
from scipy.stats import rankdata
import mdp as MDP
from operator import itemgetter
```

```python
import copy
from matplotlib.colors import ListedColormap
from matplotlib import patheffects as pe
import matplotlib.pyplot as plt
from IPython import display
from IPython.display import Markdown, Latex
import statsmodels.api as sm

pd.set_option("chained_assignment", None)

path = {
    "dscr": here("data", "scratch"),
    "drpub": here("data", "raw", "public", "spielman", "input"),
    "drpub2": here("data", "raw", "public"),
    "drpriv": here("data", "raw", "private"),
    "ddpub": here("data", "derived", "public", "version1"),
    "ddpriv": here("data", "derived", "private"),
    "rfig": here("results", "figures"),
    "roth": here("results", "other"),
    "rtab": here("results", "tables"),
    "og_out": here("data", "raw", "public", "spielman", "output"),
    "dmet": here("data", "metadata")
}
```

Data and variables

For Spielman et al.'s original study, the data sources were the 2008-2012 5-year American Community Survey and the 2010 decennial census, downloaded from Social Explorer. In our replication, we pull our data directly from the census into Python via a census API package known as pygris. These variables are based on the original work by Cutter et al. to create SoVI, and cover a wide range of social and demographic information, the particulars of which are described in the data dictionary below.

Since there are so many variables, we print their label, alias, and description just one time. To view information regarding the data type, domain, missing data values, and missing data frequency for each individual dataset, please see their individual metadata files.

(1)-(10) American Community Survey 5-year Estimates

```python
Markdown( here(path["dmet"], "RPl_ACS_geographic_metadata.md") )
```

```
<IPython.core.display.Markdown object>
```

```python
# Import data dictionary
rpl_vars = pd.read_csv( here(path["dmet"], "replication_vars.csv") )
rpl_vars.drop(columns=rpl_vars.columns[0], axis=1, inplace=True)

acs_variables = list(rpl_vars['Label'][1:])
rpl_vars
```

```
         Label                                            Alias  \
0        GEOID                       FIPS code unique identifier
1   B01002_001E                                      median age
2   B03002_001E  total population of respondents to race/ethnicity
3   B03002_004E                           total Black population
4   B03002_005E                 total Native American population
5   B03002_006E                           total Asian population
6   B03002_012E                          total Latinx population
```

7	B06001_002E	total population under 5 years of age
8	B09020_001E	total population over 65 years of age
9	B01003_001E	total population
10	B25008_001E	total population in occupied housing units
11	B25002_002E	total occupied housing units
12	B25003_003E	total renter occupied housing units
13	B25002_001E	total housing units for which occupancy status...
14	B09020_021E	total 65+ living in group quarters
15	B01001_026E	total female population
16	B11001_006E	total female-headed family households
17	B11001_001E	total households for which household type is k...
18	B25002_003E	total vacant housing units
19	B19025_001E	aggregate household income
20	B23022_025E	total males unemployed for last 12 months
21	B23022_049E	total females unemployed for last 12 months
22	B23022_001E	total population for which unemployment and se...
23	B17021_002E	total population below poverty level
24	B17021_001E	total population for which poverty information...
25	B25024_010E	number of mobile home housing units in structure
26	B25024_001E	total housing units in structure
27	C24010_038E	total female employed
28	C24010_001E	total population for which sex and occupation ...
29	B19055_002E	total households with social security income
30	B19055_001E	total households for which social security inc...
31	B09002_002E	total children in married couple families
32	B09002_001E	total children for which family type and age a...
33	B19001_017E	total households with over 200k income
34	B06007_005E	total Spanish-speakers who speak english less ...
35	B06007_008E	total people who speak another language and sp...
36	B06007_001E	total population with known language spoken at...
37	B16010_002E	total population with less than a high school ...
38	B16010_001E	total for which education, employment, languag...
39	C24050_002E	total population in extractive industries
40	C24050_001E	total population for which industry known
41	C24050_029E	total people in service occupations
42	B08201_002E	total households with no available vehicle
43	B08201_001E	total households for which vehicle status and ...
44	B25064_001E	median gross rent
45	B25077_001E	median home value

	Definition
0	Unique code for every county and county-equiva...
1	MEDIAN AGE BY SEX: Estimate!!Median age!!Total
2	HISPANIC OR LATINO ORIGIN BY RACE: Estimate!!T...
3	HISPANIC OR LATINO ORIGIN BY RACE: Estimate!!T...
4	HISPANIC OR LATINO ORIGIN BY RACE: Estimate!!T...
5	HISPANIC OR LATINO ORIGIN BY RACE: Estimate!!T...
6	HISPANIC OR LATINO ORIGIN BY RACE: Estimate!!T...
7	PLACE OF BIRTH BY AGE IN THE UNITED STATES: Es...
8	RELATIONSHIP BY HOUSEHOLD TYPE (INCLUDING LIVI...
9	TOTAL POPULATION: Estimate!!Total
10	TOTAL POPULATION IN OCCUPIED HOUSING UNITS BY ...
11	OCCUPANCY STATUS: Estimate!!Total!!Occupied
12	TENURE: Estimate!!Total!!Renter occupied

```
13                   OCCUPANCY STATUS: Estimate!!Total
14  RELATIONSHIP BY HOUSEHOLD TYPE (INCLUDING LIVI...
15                 SEX BY AGE: Estimate!!Total!!Female
16  HOUSEHOLD TYPE (INCLUDING LIVING ALONE): Estim...
17  HOUSEHOLD TYPE (INCLUDING LIVING ALONE): Estim...
18          OCCUPANCY STATUS: Estimate!!Total!!Vacant
19  AGGREGATE HOUSEHOLD INCOME IN THE PAST 12 MONT...
20  SEX BY WORK STATUS IN THE PAST 12 MONTHS BY US...
21  SEX BY WORK STATUS IN THE PAST 12 MONTHS BY US...
22  SEX BY WORK STATUS IN THE PAST 12 MONTHS BY US...
23  POVERTY STATUS OF INDIVIDUALS IN THE PAST 12 M...
24  POVERTY STATUS OF INDIVIDUALS IN THE PAST 12 M...
25   UNITS IN STRUCTURE: Estimate!!Total!!Mobile home
26                 UNITS IN STRUCTURE: Estimate!!Total
27  SEX BY OCCUPATION FOR THE CIVILIAN EMPLOYED PO...
28  SEX BY OCCUPATION FOR THE CIVILIAN EMPLOYED PO...
29  SOCIAL SECURITY INCOME IN THE PAST 12 MONTHS F...
30  SOCIAL SECURITY INCOME IN THE PAST 12 MONTHS F...
31  OWN CHILDREN UNDER 18 YEARS BY FAMILY TYPE AND...
32  OWN CHILDREN UNDER 18 YEARS BY FAMILY TYPE AND...
33  HOUSEHOLD INCOME IN THE PAST 12 MONTHS (IN 201...
34  PLACE OF BIRTH BY LANGUAGE SPOKEN AT HOME AND ...
35  PLACE OF BIRTH BY LANGUAGE SPOKEN AT HOME AND ...
36  PLACE OF BIRTH BY LANGUAGE SPOKEN AT HOME AND ...
37  EDUCATIONAL ATTAINMENT AND EMPLOYMENT STATUS B...
38  EDUCATIONAL ATTAINMENT AND EMPLOYMENT STATUS B...
39  INDUSTRY BY OCCUPATION FOR THE CIVILIAN EMPLOY...
40  INDUSTRY BY OCCUPATION FOR THE CIVILIAN EMPLOY...
41  INDUSTRY BY OCCUPATION FOR THE CIVILIAN EMPLOY...
42  HOUSEHOLD SIZE BY VEHICLES AVAILABLE: Estimate...
43  HOUSEHOLD SIZE BY VEHICLES AVAILABLE: Estimate...
44  MEDIAN GROSS RENT (DOLLARS): Estimate!!Median ...
45  MEDIAN VALUE (DOLLARS): Estimate!!Median value...
```

(11) USA Counties Cartographic Boundaries

```
%%script echo skipping # Comment this first line out if you wish to acquire data directly from census
# Acquire geographical data for reproduction
counties_shp = counties(cb = True, year = 2010, cache = True) # year 2012 (and 2011) cartographic bound

# Save raw data
counties_shp.to_file( here(path["drpub2"], "counties_geometries_raw.gpkg") )
```

```
skipping # Comment this first line out if you wish to acquire data directly from census
```

```
# Optionally, load data directly from the repository
counties_shp = gpd.read_file( here(path["drpub2"], "counties_geometries_raw.gpkg") )
```

```
Markdown( here(path["dmet"], "county_geom_2010_metadata.md") )
```

```
<IPython.core.display.Markdown object>
```

Prior observations

At the time of this study pre-registration, the authors had examined the Python code and data from the original study and modified the code to reproduce the original results in a current software environment.

This study is related to one prior study by the authors, a reproduction of Spielman et al.'s "Evaluating Social Vulnerability Indicators" (Spielman et al., 2020).

We have already thoroughly observed datasets (1) and (11), as they were necessary to reproduce the original study. On the other hand, we have yet to thoroughly analyze datasets (2) through (10). We have imported these datasets into Python in order to generate the data on missing data and domains in our metadata files. We have otherwise not directly inspected, transformed, visualized, or analyzed the data. All of the work we have done with the data can be seen in our `generate_RPl_metadata.ipynb` file in the metadata folder.

Bias and threats to validity

Possible threats to validity in this study include spatial dependence and nonstationarity, temporal dependence, and scale dependency.

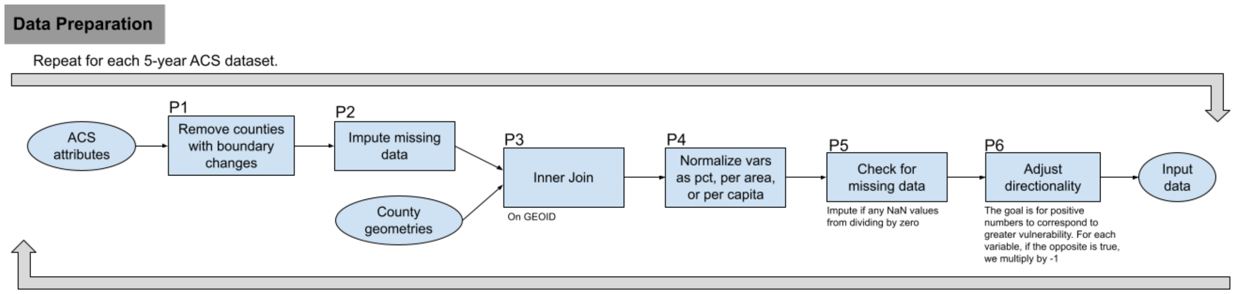
Regarding spatial dependence, we may expect nearby places to exhibit similar levels of social vulnerability as well as similar underlying causes of it. Regarding spatial nonstationarity, we may expect the causal relationships underlying social vulnerability to change over space in the model. Although spatially explicit PCA methods have been developed, the methodology used to calculate SoVI uses ordinary PCA, which does not account for spatial relationships. Thus, each county is treated as an independent observation even though there may be spatial clustering of social vulnerability and its underlying factors, and PCA is a global model that does not account for spatial nonstationarity. To this end, Spielman et al. (2020) found substantially different variable weights and rankings of SoVI scores depending on the spatial extent of input data. We do not control for spatial dependence or nonstationarity since our work is also an evaluation of SoVI's consistency based on the original study methodology.

We also anticipate temporal correlation in county SoVI scores, because demographic characteristics change over time and may be more similar in two subsequent years than in two years that are a decade apart. We account for temporal dependence by using time as an independent variable in linear regression. However, there is temporal dependence not only in social vulnerability but also in the very data we are using for our analysis. Because we use overlapping 5-year averages, four-fifths of the data used to generate each dataset is also used to generate the subsequent and previous year's datasets. The similarity of input data puts us at risk of finding SoVI to be more consistent than it really is. Unfortunately, increasing the temporal resolution to the one-year ACS estimates is not possible beause of the large amount of missing data and uncertainty in that data product and increasing the temporal extent is not possible because of changes in the census data products.

Finally, SoVI is likely scale dependent, as some input demographic variables, like race and ethnicity, show substantially greater variation at the tract level than at the county level. Differing variability depending on scale of analysis may have spillover effects on variance-based models such as PCA. For example, Eric Tate (2012) found that PCA-based social vulnerability models like SoVI exhibit high levels of scale dependency (Tate, 2012). The implication for this study is that we are at risk of finding SoVI to be consistent over time at the county level without detecting inconsistency over time at the tract level.

Data transformations

To prepare our raw data for input into the SoVI model, we need to consider only the counties that are unchanged in shape over the study window, impute for missing data, normalize the variables, and adjust their directionality such that all variables are (theoretically) directly related with social vulnerability. A draft workflow diagram for this section is displayed below.



Note on Step P1:

Since we are working with census data from ten different years and we want to track changes in SoVI scores in counties over time, we have to account for changes in county boundaries over this time. Fortunately, the Census publishes information regarding such changes on its website. The changes for the 2010 decade are available here and there were no such changes listed in the 2020 decade until 2022.

Below is a list of all counties with relevant changes between 2012 and 2021:

- Chugach Census Area, Alaska (02063). Created from 02261 in 2019.
- Copper River Census Area, Alaska (02066). Created from 02261 in 2019.
- Valdez-Cordova Census Area, Alaska (02261). Split into 02063 and 02066 in 2019.
- Petersburg Borough, Alaska (02195). Created from 02195 and part of 02105 in 2013.
- Hoonah-Angoon Census Area, Alaska (02105). Part given to 02195 in 2013.
- Bedford (independent) city, Virginia (51515). Became a town, absorbed by Bedford County (51019).
- Kusilvak Census Area, Alaska (02158). Changed name and code from Wade Hampton Census Area and 02270 in 2015.
- Oglala Lakota County, South Dakota (46102). Changed name and code from Shannon County and 46113 in 2015.
- Prince of Wales-Hyder Census Area, Alaska (02198). Added part of 02195 in 2013.
- Bedford County, Virginia (51019). Added 51515 in 2013.

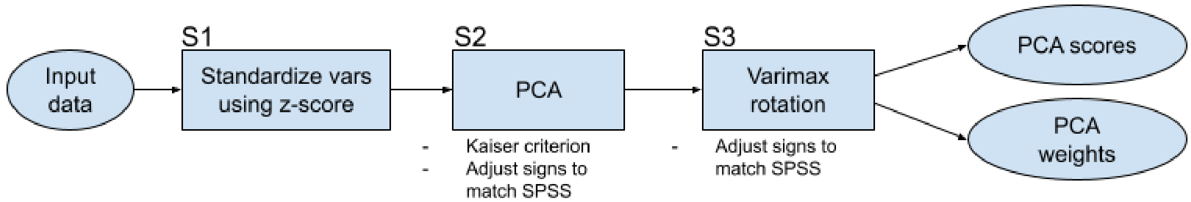
In order to generate data comparable across all ten years of interest, we simply remove these counties from each dataset containing them, except for counties 02158 and 46102, which simply changed names and codes. For counties 02158 and 46102, we just adjust their FIPS codes to match.

A final step of data transformation will be performed at the beginning of the SoVI model analysis. Each demographic variable will be standardized by calculating its z-score.

Analysis

Principal Component Analysis Spielman et al. constructed a class to conduct SPSS-style PCA with varimax rotation in Python and validated their procedure against Cutter et al.'s SPSS workflow used to calculate SoVI. Below we include a workflow diagram that shows the main operations and important outputs of their SPSS_PCA class. After that, we include their relevant code.

PCA Workflow We will later refer to this workflow as one function, SPSS_PCA.



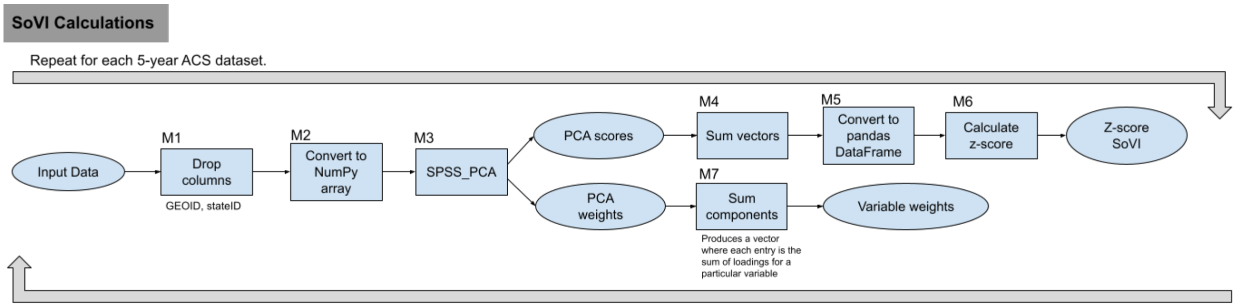

```
class SPSS_PCA:
        '''
        A class that integrates most (all?) of the assumptions SPSS imbeds in their
    implimmetation of principal components analysis (PCA), which can be found in
    thier GUI under Analyze > Dimension Reduction > Factor. This class is not
        intended to be a full blown recreation of the SPSS Factor Analysis GUI, but
        it does replicate (possibly) the most common use cases. Note that this class
        will not produce exactly the same results as SPSS, probably due to differences
        in how eigenvectors/eigenvalues and/or singular values are computed. However,
        this class does seem to get all the signs to match, which is not really necessary
        but kinda nice. Most of the approach came from the official SPSS documentation.

        References
        ----------
        ftp://public.dhe.ibm.com/software/analytics/spss/documentation/statistics/20.0/en/
    client/Manuals/IBM_SPSS_Statistics_Algorithms.pdf
        http://spssx-discussion.1045642.n5.nabble.com/Interpretation-of-PCA-td1074350.html
        http://mdp-toolkit.sourceforge.net/api/mdp.nodes.WhiteningNode-class.html
        https://github.com/mdp-toolkit/mdp-toolkit/blob/master/mdp/nodes/pca_nodes.py

        Parameters
        ----------
        inputs:  numpy array
                         n x k numpy array; n observations and k variables on each observation
        reduce:  boolean (default=False)
                         If True, then use eigenvalues to determine which factors to keep; all
                         results will be based on just these factors. If False use all factors.
        min_eig: float (default=1.0)
                         If reduce=True, then keep all factors with an eigenvalue greater than
                         min_eig. SPSS default is 1.0. If reduce=False, then min_eig is ignored.
        varimax: boolean (default=False)
                         If True, then apply a varimax rotation to the results. If False, then
                         return the unrotated results only.

        Attributes
        ----------
        z_inputs:        numpy array
                                z-scores of the input array.
        comp_mat:        numpy array
                                Component matrix (a.k.a, "loadings").
        scores:                numpy array
```

```
                        New uncorrelated vectors associated with each observation.
eigenvals_all:        numpy array
                        Eigenvalues associated with each factor.
eigenvals:        numpy array
                        Subset of eigenvalues_all reflecting only those that meet the
                        criterion defined by parameters reduce and min_eig.
weights:    numpy array
                        Values applied to the input data (after z-scores) to get the PCA
                        scores. "Component score coefficient matrix" in SPSS or
                        "projection matrix" in the MDP library.
comms:                  numpy array
                        Communalities
sum_sq_load: numpy array
                         Sum of squared loadings.
comp_mat_rot: numpy array or None
                          Component matrix after rotation. Ordered from highest to lowest
                          variance explained based on sum_sq_load_rot. None if varimax=False.
scores_rot:        numpy array or None
                        Uncorrelated vectors associated with each observation, after
                        rotation. None if varimax=False.
weights_rot: numpy array or None
                        Rotated values applied to the input data (after z-scores) to get
                        the PCA        scores. None if varimax=False.
sum_sq_load_rot: numpy array or None
                         Sum of squared loadings for rotated results. None if
                         varimax=False.

'''

def __init__(self, inputs, reduce=False, min_eig=1.0, varimax=False):

# Step S1: Standardize inputs
        z_inputs = ZSCORE(inputs)  # necessary for SPSS "correlation matrix" setting

# Step S2: Unrotated PCA
        # run base SPSS-style PCA to get all eigenvalues
        pca_node = MDP.nodes.WhiteningNode()  # settings for the PCA
        scores = pca_node.execute(z_inputs)  # base run PCA
        eigenvalues_all = pca_node.d   # rename PCA results

        # run SPSS-style PCA based on user settings
        # settings for the PCA
pca_node = MDP.nodes.WhiteningNode(reduce=reduce, var_abs=min_eig)
# run PCA  (these have mean=0, std_dev=1)
        scores = pca_node.execute(z_inputs)
weights = pca_node.v # save weights from PCA results
eigenvalues = pca_node.d # save eigenvalues from PCA results
        component_matrix = weights * eigenvalues  # compute the loadings
        # invert signs for components with mostly negative loadings
component_matrix = self._reflect(component_matrix)
        communalities = (component_matrix**2).sum(1) # compute the communalities
# compute sum of the squares of loadings, same as eigenvalues
        sum_sq_loadings = (component_matrix**2).sum(0)
# divide matrix by eigenvalues
```

```
        weights_reflected = component_matrix/eigenvalues
# calculate scores, where abs(scores)=abs(scores_reflected)
        scores_reflected = np.dot(z_inputs, weights_reflected)

# Step S3: Varimax rotation
        if varimax:
                # SPSS-style varimax rotation prep
    # normalize inputs to varimax
                c_normalizer = 1. / MDP.numx.sqrt(communalities)
    # reshape to vectorize normalization
                c_normalizer.shape = (component_matrix.shape[0],1)
    # normalize component matrix for varimax
                cm_normalized = c_normalizer * component_matrix

                # varimax rotation
                cm_normalized_varimax = self._varimax(cm_normalized)  # run varimax
                # denormalize varimax output
    c_normalizer2 = MDP.numx.sqrt(communalities)
    # reshape to vectorize denormalization
                c_normalizer2.shape = (component_matrix.shape[0],1)
    # denormalize varimax output
                cm_varimax = c_normalizer2 * cm_normalized_varimax

                # reorder varimax component matrix
                # base the ordering on sum of squared loadings
    sorter = (cm_varimax**2).sum(0)
    # add index to denote current order
                sorter = zip(sorter.tolist(), range(sorter.shape[0]))
    # sort from largest to smallest
                sorter = sorted(sorter, key=itemgetter(0), reverse=True)
    # unzip the sorted list
                sum_sq_loadings_varimax, reorderer = zip(*sorter)
    # convert to array
                sum_sq_loadings_varimax = np.array(sum_sq_loadings_varimax)
    # reorder component matrix
                cm_varimax = cm_varimax[:,reorderer]

                # varimax scores
    # invert signs for factors with mostly negative loadings
                cm_varimax_reflected = self._reflect(cm_varimax)
                varimax_weights = np.dot(cm_varimax_reflected,
                                  np.linalg.inv(np.dot(cm_varimax_reflected.T,
                                  cm_varimax_reflected))) # CM(CM'CM)^-1
                scores_varimax = np.dot(z_inputs, varimax_weights)
        else:
                comp_mat_rot = None
                scores_rot = None
                weights_rot = None

        # assign output variables
        self.z_inputs = z_inputs
        self.scores = scores_reflected
        self.comp_mat = component_matrix
        self.eigenvals_all = eigenvalues_all
```

```
        self.eigenvals = eigenvalues
        self.weights = weights_reflected
        self.comms = communalities
        self.sum_sq_load = sum_sq_loadings
        self.comp_mat_rot = cm_varimax_reflected
        self.scores_rot = scores_varimax # PCA scores output
        self.weights_rot = varimax_weights # PCA weights output
        self.sum_sq_load_rot = sum_sq_loadings_varimax

def _reflect(self, cm):
        # reflect factors with negative sums; SPSS default
        cm = copy.deepcopy(cm)
        reflector = cm.sum(0)
        for column, measure in enumerate(reflector):
                if measure < 0:
                        cm[:,column] = -cm[:,column]
        return cm

def _varimax(self, Phi, gamma = 1.0, q = 100, tol = 1e-6):
        # see http://en.wikipedia.org/wiki/Talk%3aVarimax_rotation
        # and http://stackoverflow.com/questions/17628589/
# perform-varimax-rotation-in-python-using-numpy
        p,k = Phi.shape
        R = np.eye(k)
        d=0
        for i in range(q):
                d_old = d
                Lambda = np.dot(Phi, R)
                u,s,vh = np.linalg.svd(
        np.dot(Phi.T, np.asarray(Lambda)**3 - (gamma/p) *
               np.dot(Lambda,
                      np.diag(np.diag(np.dot(Lambda.T,
                                             Lambda))))))
                R = np.dot(u,vh)
                d = np.sum(s)
                if d_old!=0 and d/d_old < 1 + tol:
                        break
        return np.dot(Phi, R)
```

Calculating SoVI At this stage, we seek to calculate the z-score standardized SoVI and variable weightings for each dataset. Below is our workflow for calculating SoVI, adapted from the workflow in our reproduction.



Internal consistency analysis We will develop a simple linear regression model for each county in the analysis, where the independent variable is time and the dependent variable is the county's z-score standardized SoVI score. To do this, we will need to manipulate our output data to generate a dataset with a row for each year, a column documenting the year, and a column for each county's z-score standardized SoVI scores. We will use the statsmodels package and its regression.linear_model.OLS¶ class to conduct the regression analysis and generate standard errors. The standard error of our year variable's regression coefficient will illustrate how consistent SoVI scores are in a particular location when controlling for time.

Theoretical consistency analysis Similar to Spielman et al.'s analysis, we will sum together all of the components for each model in order to determine the net effect of each variable on the final SoVI score. We will summarize this information in a dataframe with a row for each variable and a column for each model. For each model, we will then rank the variables with positive coefficients and negative coefficients separately, by magnitude.

Results

RPl-H1

We plan to present our results of our internal consistency analysis by producing a country-wide map of standard error coefficients. We will also calculate summary statistics of standard error coefficients, including minimum, mean, and maximum.

RPl-H2

Will will present our theoretical consistency results by producing a rank chart displaying the rank of each variable over time. We will place the ranks of positive coefficients above the X-axis and the ranks of negative coefficients below the X-axis. In this manner, the number of variables on either side of the X-axis may vary between models, but we will present information regarding both the magnitude and sign of variables in one graph.

Discussion

RPl-H1

Standard errors in simple linear regression represent the average distance between the actual data and the line of best fit. The smaller the standard errors, the better the SoVI scores fit a linear trend over time. Since standard errors use the units of the response variable and the response variable is z-score normalized, the standard errors can be interpreted as the the average deviance in SoVI scores from a linear trend over time, where the unit of measurement is one standard deviation of the SoVI score. In this manner, our standard errors will illustrate the degree to which SoVI scores are consistent when controlling for time. We would expect standard errors within one standard deviation if the SoVI model is internally consistent over time, especially when we consider the temporal dependency in the data.

RPl-H2

If SoVI is theoretically consistent, we would anticipate variable rankings and signs to be fairly consistent over time. On our figure, this would be illustrated with few lines crossing each other, few lines crossing the zero line, and narrow ranges of each variable's rankings. The more that variables switch signs, and the greater the range of each variable's ranks, the less theoretically consistent SoVI is over time.

Integrity Statement

The authors of this preregistration state that they completed this preregistration to the best of their knowledge and that no other preregistration exists pertaining to the same hypotheses and research.

Acknowledgements

- `Funding Name`: NSF Directorate for Social, Behavioral and Economic Sciences
- `Funding Title`: Transforming theory-building and STEM education through reproductions and replications in the geographical sciences
- `Award info URI`: https://www.nsf.gov/awardsearch/showAward?AWD_ID=2049837
- `Award number`: BCS-2049837

This report is based upon the template for Reproducible and Replicable Research in Human-Environment and Geographical Sciences, DOI:%5B10.17605/OSF.IO/W29MQ](https://doi.org/10.17605/OSF.IO/W29MQ)

References

- Cutter, S. L., Boruff, B. J., & Shirley, W. L. (2003). Social Vulnerability to Environmental Hazards. Social Science Quarterly, 84(2), 242–261. https://doi.org/10.1111/1540-6237.8402002
- Spielman, S. E., Tuccillo, J., Folch, D. C., Schweikert, A., Davies, R., Wood, N., & Tate, E. (2020). Evaluating Social Vulnerability Indicators: Criteria and their Application to the Social Vulnerability Index. Natural Hazards, 100(1), 417–436. https://doi.org/10.1007/s11069-019-03820-z
- Tate, E. (2012). Social vulnerability indices: A comparative assessment using uncertainty and sensitivity analysis. Natural Hazards, 63(2), 325–347. https://doi.org/10.1007/s11069-012-0152-2
- United States Census Bureau. (2021, October 8). Substantial Changes to Counties and County Equivalent Entities: 1970-Present. https://www.census.gov/programs-surveys/geography/technical-documentation/county-changes.2010.html#list-tab-957819518