Part 2: Generating a layered pseudo-3D map using R

Pete Skach and Eric Olle

10/14/2022

Part 2: Generating a layered pseudo-3D map using R

The goal of this document is to modify and use a rotate_sf() function written by Stephan Junger and Denis Cohen in the “Method Bytes” blog. The function was modified to allow the user to change the shear matrix and the angle of rotation (theta). Additionally, the rotation calculation was slightly modified to approximate the theta requested by the user. This is meant to modify an excellent function written by Junger and Cohen and be added at a later date to another package (trauma directors tool kit tdtk).

This is the second part of a three-part series on basic geo-spatial in R using census data. In part one census data was plotted for different variables and then graphed. Part two looks at rotating and seperating the layers in a pseudo 3D projection. Finally, in part 3 a bivariate color scheme will be used a generated to show how to plot two variable in a low, moderate and high categories. Census data was chosen since it is easy to find basic statistics that are matched to the necessary geometry files. A lot of the work is derived from existing references such as: TidyCensus, CensusAPI and blog posts.

Why is it important to use a rotated projection? In geo-spatial analysis it is easy to get overwhelmed by multiple variables that can be projected in separate plots (later blog post) or as a bivariate color sheme. In this basic but not optimal example, the number of individual at different level of the poverty line were extracted and displayed. This was done using the tidycensus package to interface with the 2020 5 year-American Community Survey data. This data contains different subsets of the poverty line (i.e. < 0.5, 0.5 to < 1.0 and 1.0 to < 1.5 - times the poverty line) and does not fit into a standard bi or tri variate color scheme. The separation and rotation of the layers allows for a rapid visualization of how the number of individuals at different levels of the poverty line changes on a census tract level.

The basic equation used to convert a 2D to pseudo3D image is as follows:

$$\psi_{3D} = \begin{vmatrix} 1 & \lambda \\ \mu & 1 \end{vmatrix} \times \begin{vmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{vmatrix} + (\, X_{shift}, Y_{shift} \,)$$

The rotate_sf() takes an angle in degrees and converts to radians. The shear matrix uses values lambda and mu to shift parallel to the Y or X, axis.

Document summary:

Generatin a layerd pseudu-3D map using R Written by: Pete Skach and Eric W. Olle CC-BY-SA license for the r-markdown document. GPLv3 for the code. Make sure you visit the Junger and Cohen blog (reference below). No Warranty Provided Created: Oct 2022 Edited for blog: October 14, 2022

The GitHub for this project contains the R-markdown and the data used to generate this blog. The data and the markdown document are CC-BY-SA.

Loadiing the packages

```
 library(tidyverse)  # The lazy way.  Should load individual packages.
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──
## ✔ ggplot2 3.3.6     ✔ purrr   0.3.4
## ✔ tibble  3.1.7     ✔ dplyr   1.0.9
## ✔ tidyr   1.2.0     ✔ stringr 1.4.0
## ✔ readr   2.1.2     ✔ forcats 0.5.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
 library(magrittr) # sometimes need for the pipe and dual pipe ops
##
## Attaching package: 'magrittr'
## The following object is masked from 'package:purrr':
##
##     set_names
## The following object is masked from 'package:tidyr':
##
##     extract
 library(tigris)
## To enable caching of data, set `options(tigris_use_cache = TRUE)`
## in your R script or .Rprofile.
 library(tidycensus)
 options(tigris_use_cache = TRUE)
```

Once the packages are loaded the next step is to get the US Census Data with the geospatial geometry. Check out the tidycensus package for more information on how to retrieve the data. Make sure you have the your census API token correctly configured or no data will be retreived.

Getting the Census data

This is using the data from a 5-year American Community Survey but can be a range of different sources including the 10-year full census. For information on the variable list see:

https://api.census.gov/data/2020/acs/acs5/variables.html (https://api.census.gov/data/2020/acs/acs5/variables.html)

```
WV_poverty <- get_acs(
  state = "WV",
  geography = "tract",
  variables = c(total = "C18131_001E",
                lt05 = "C18131_002E",
                gt5to99 = "C18131_005E",
                gt1to15 = "C18131_008E"),
  geometry = TRUE,
  output = "wide",
  year = 2020)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
names(WV_poverty)
```

```
##  [1] "GEOID"       "NAME"        "total"       "C18131_001M" "lt05"
##  [6] "C18131_002M" "gt5to99"     "C18131_005M" "gt1to15"     "C18131_008M"
## [11] "geometry"
```

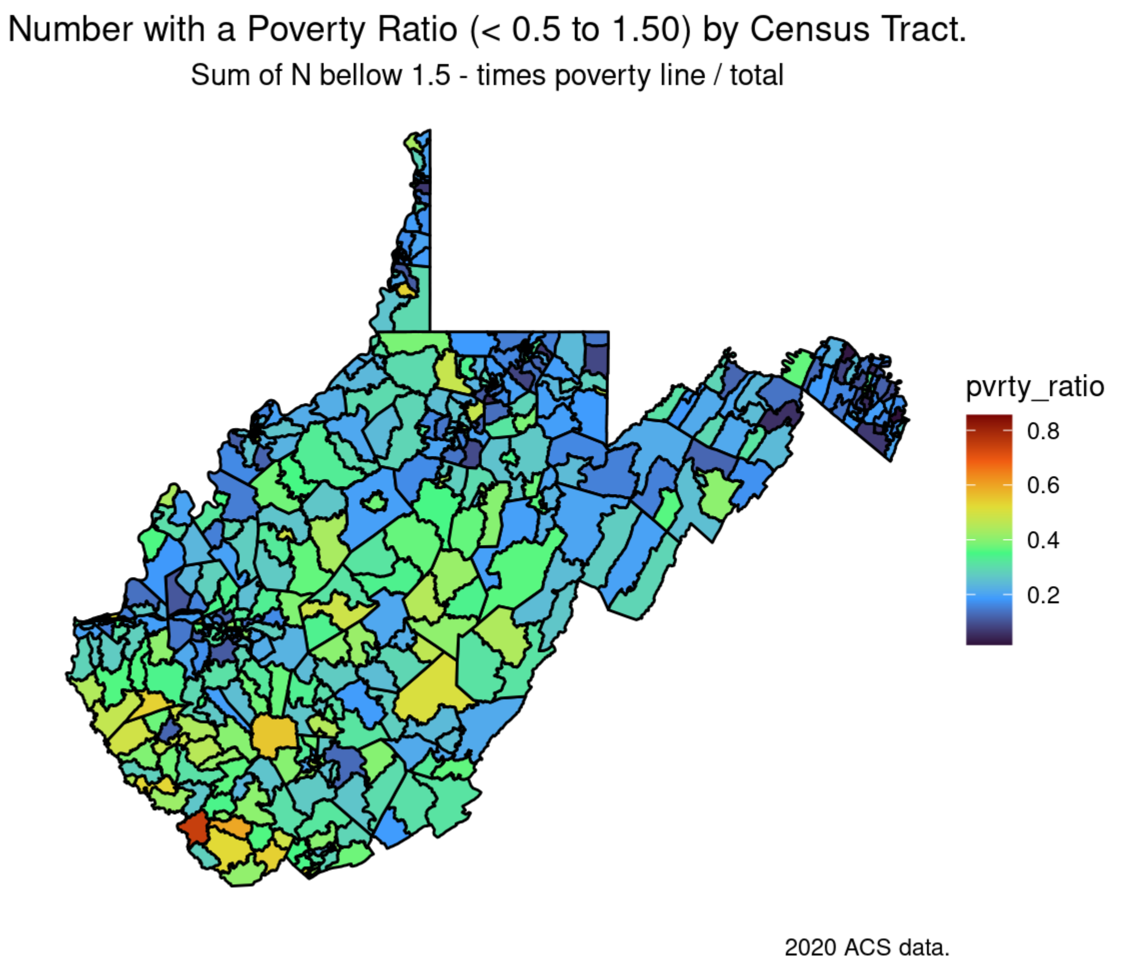
Looking at the column names shows that the geomotry was loaded along with the data with margin of error. This can be verified with a simple "head()" command. This will also verify that it is a spatial dataframe.

```
head(WV_poverty)
```

```
## Simple feature collection with 6 features and 10 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: -82.54137 ymin: 38.39577 xmax: -79.80879 ymax: 40.05918
## Geodetic CRS:  NAD83
##         GEOID                                            NAME total
## 1 54001965700 Census Tract 9657, Barbour County, West Virginia  4932
## 2 54001965500 Census Tract 9655, Barbour County, West Virginia  3845
## 3 54069001300      Census Tract 13, Ohio County, West Virginia  1352
## 4 54011001400    Census Tract 14, Cabell County, West Virginia  2332
## 5 54011001900    Census Tract 19, Cabell County, West Virginia  2405
## 6 54099005100     Census Tract 51, Wayne County, West Virginia  1891
##   C18131_001M lt05 C18131_002M gt5to99 C18131_005M gt1to15 C18131_008M
## 1         506  377         205     355         262     577         258
## 2         457  286         167     530         351     168         118
## 3         258   79          58      79          54      73          54
## 4         474  576         303     421         264     179         104
## 5         305   87          59     207         106     182         132
## 6         183   78          59     288         152     211          89
##                         geometry
## 1 POLYGON ((-80.07606 39.1016...
## 2 POLYGON ((-80.22631 39.1235...
## 3 POLYGON ((-80.72345 40.0347...
## 4 POLYGON ((-82.43736 38.4168...
## 5 POLYGON ((-82.40455 38.4074...
## 6 POLYGON ((-82.54137 38.3963...
```

Next is to load a modified rotate_sf() orginally developed by Junger and Cohen (see link at the end for the initial version).

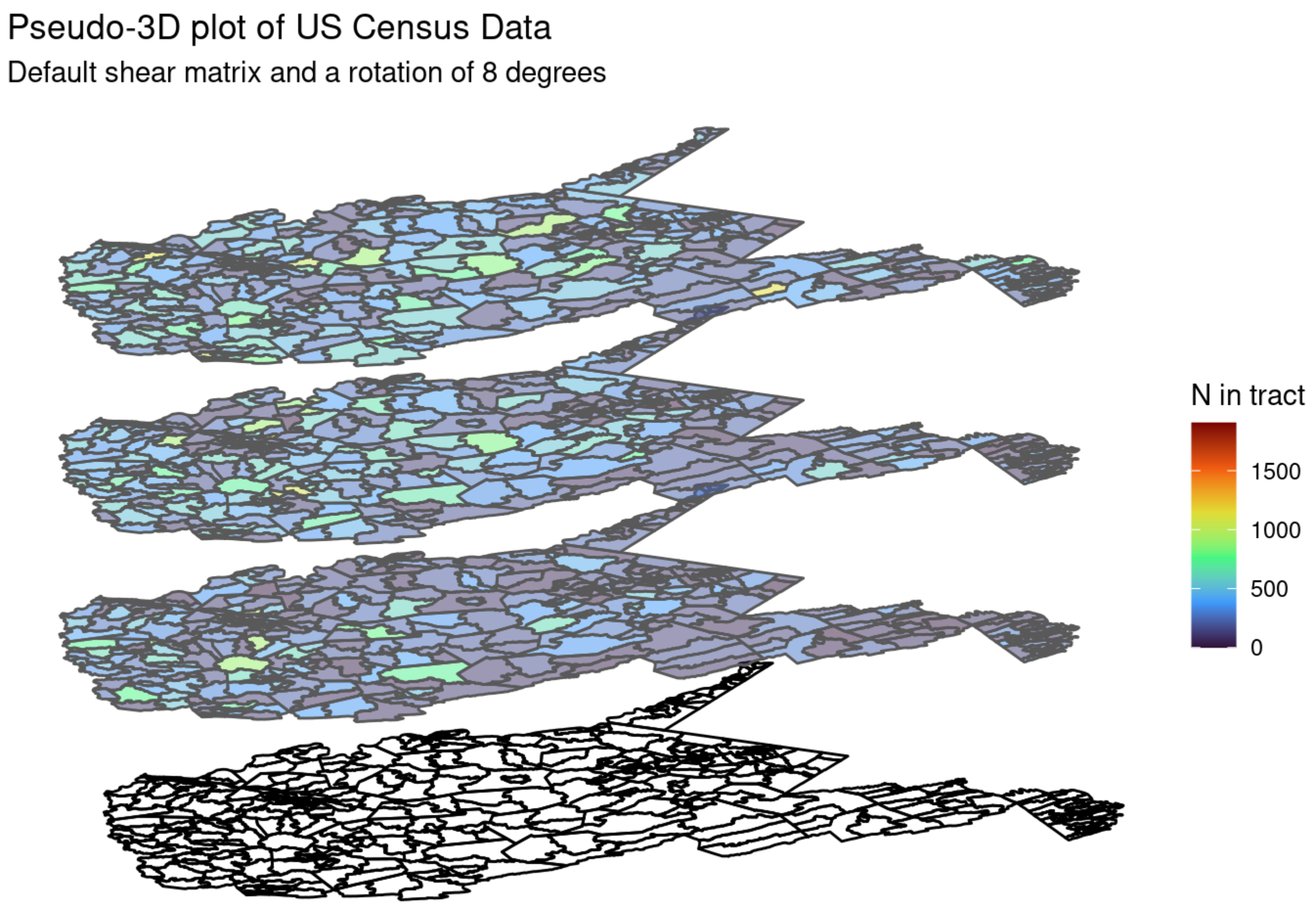
```
WV_poverty %>%
    rowwise() %>%
    mutate(pvrty_est = sum(c(lt05, gt5to99, gt1to15))) %>%
    mutate(pvrty_ratio = pvrty_est/total) -> WV_poverty_sum


WV_poverty_sum %>%
    ggplot(aes(fill =  pvrty_ratio)) +
    geom_sf(color = "black") +
    theme_void() +
    scale_fill_viridis_c(option = "turbo") +
        theme(plot.title = element_text(hjust = 0.5),
          plot.subtitle = element_text(hjust = 0.5)) +
    labs(title = "Number with a Poverty Ratio (< 0.5 to 1.50) by Census Tract.",
         caption = "2020 ACS data.",
         subtitle = "Sum of N bellow 1.5 - times poverty line / total")
```


Number with a Poverty Ratio (< 0.5 to 1.50) by Census Tract.
Sum of N bellow 1.5 - times poverty line / total
pvrty_ratio
0.8
0.6
0.4
0.2
2020 ACS data.

This a a probability based upon the calculated "total" variable from the US Census data. These are summed. This may however not be the best way to summarize the data due to the Margin of Error (MOE) associated with US Census ACS estimates. Additionally, sometimes one or more variables need to be show concurrently but it is not amenable to a faceted or side-by-side display. With a "flat-state" displayed and rotate_sf() function is loaded it is time to "play around." The next sections will look at rotating and seprating the layers along with the role of theta, lambda and mu in the rotation and shear matrices.

Pseudo-3D with Basic Junger and Cohen defaults

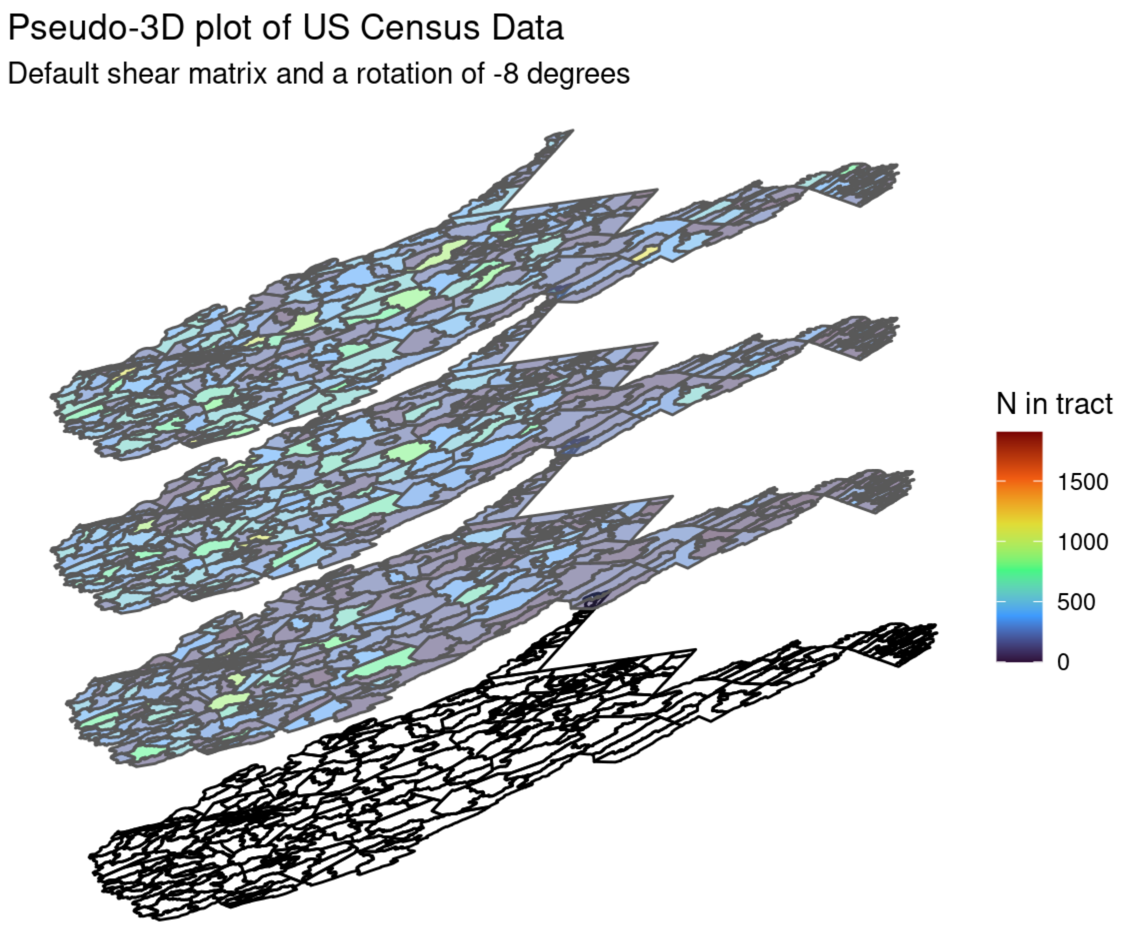
```
ggplot() +
    geom_sf(data = rotate_sf(WV_poverty, theta = 8), fill = "white", color = "black",
alpha = 0.1) +
    geom_sf(data = rotate_sf(WV_poverty,theta = 8, y_add = 2, x_add = -0.5), aes(fill
= lt05),
            alpha = .5) +
     geom_sf(data = rotate_sf(WV_poverty,theta = 8, y_add = 4, x_add = -0.5), aes(fil
l = gt5to99),
             alpha = .5) +
    geom_sf(data = rotate_sf(WV_poverty,theta = 8, y_add = 6, x_add = -0.5), aes(fill
= gt1to15),
            alpha = .5) +
    scale_fill_viridis_c(option = "turbo") +
    theme_void() +
   labs(title = "Pseudo-3D plot of US Census Data",
         subtitle = "Default shear matrix and a rotation of 8 degrees",
         fill = "N in tract")
```


Pseudo-3D plot of US Census Data
Default shear matrix and a rotation of 8 degrees
N in tract
1500
1000
500
0

In this plot the Junger and Cohen defaults are used with an similar theta as their intial work (i.e. pi/20 rad versus 8 degrees). The same shear matrix.

Modifing the theta (added functionality)

```
ggplot() +
    geom_sf(data = rotate_sf(WV_poverty, theta =-8), fill = "white", color = "black",
alpha = 0.1) +
    geom_sf(data = rotate_sf(WV_poverty,theta = -8, y_add = 2, x_add = -0.3), aes(fil
l = lt05),
            alpha = 0.5) +
     geom_sf(data = rotate_sf(WV_poverty,theta = -8, y_add = 4, x_add = -0.5), aes(fi
ll = gt5to99),
             alpha = 0.5) +
    geom_sf(data = rotate_sf(WV_poverty,theta = -8, y_add = 6, x_add = -0.5), aes(fil
l = gt1to15),
            alpha = 0.5) +
    scale_fill_viridis_c(option = "turbo") +
    theme_void() +
   labs(title = "Pseudo-3D plot of US Census Data",
         subtitle = "Default shear matrix and a rotation of -8 degrees",
         fill = "N in tract")
```


Pseudo-3D plot of US Census Data
Default shear matrix and a rotation of -8 degrees
N in tract
1500
1000
500
0

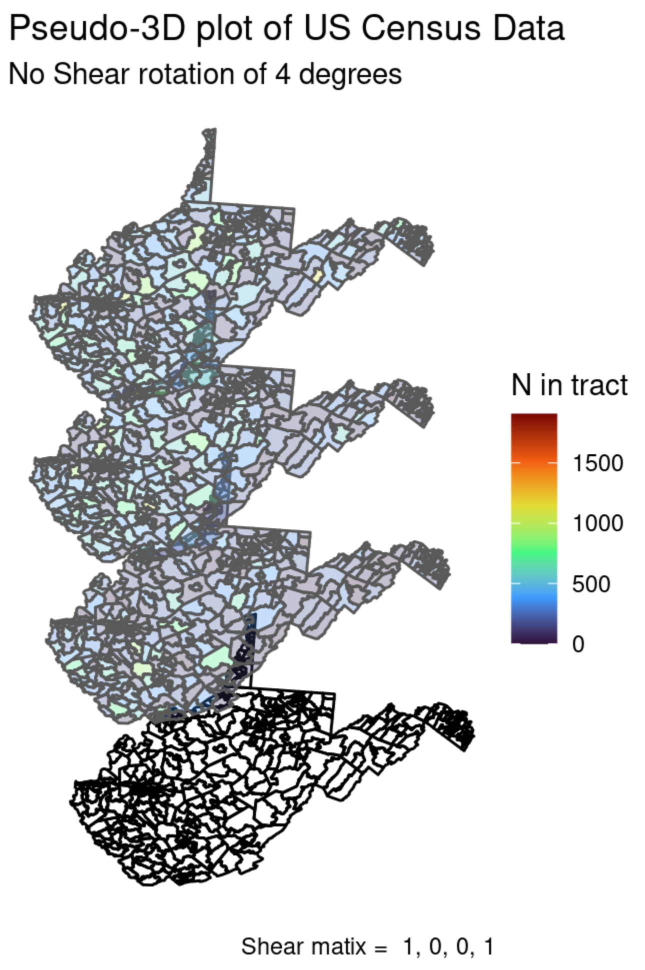
Modifying the Shear matrix

The shear matrix sets the amount the X and Y values are “sheared” parallel to the X or Y axis.

Modifying the shear matrix to 0 lambda and mu

Notice how setting the UR and LL values to 0 “flatten” the data.

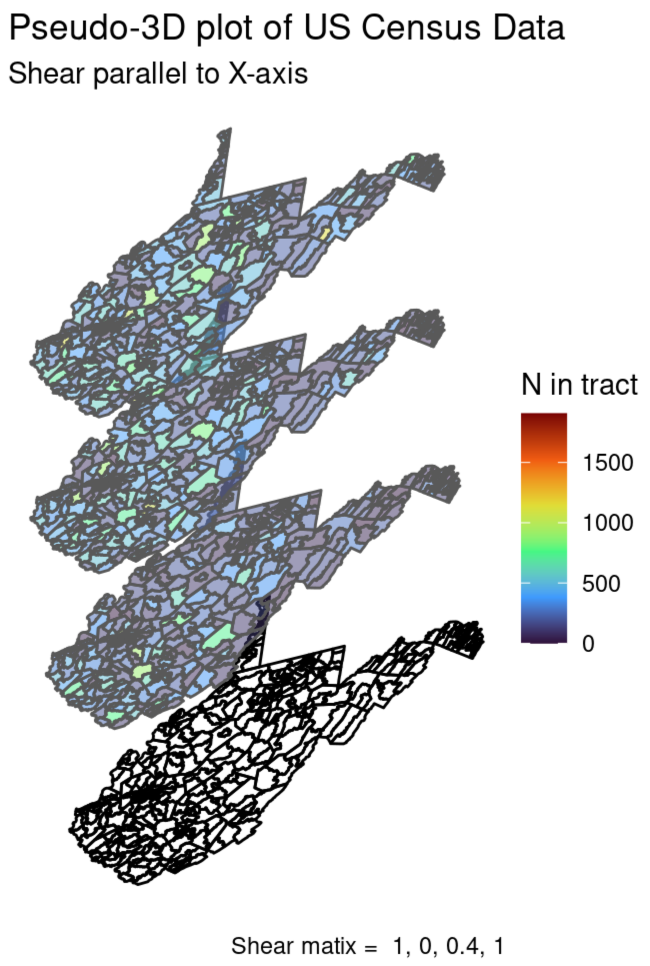
```
new_shear <- c(1, 0, 0, 1)


ggplot() +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 4), fill =
"white", color = "black", alpha = 0.1) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  4, y_add
= 2, x_add = -0.3), aes(fill = lt05),
            alpha = 0.3) +
     geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  4, y_add
= 4, x_add = -0.5), aes(fill = gt5to99),
             alpha = 0.3) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 4, y_add =
6, x_add = -0.5), aes(fill = gt1to15),
            alpha = 0.3) +
    scale_fill_viridis_c(option = "turbo") +
    theme_void() +
     labs(title = "Pseudo-3D plot of US Census Data",
         subtitle = "No Shear rotation of 4 degrees",
         caption =  paste("Shear matix = " ,paste(new_shear, collapse = ", ")),
         fill = "N in tract")
```


Pseudo-3D plot of US Census Data
No Shear rotation of 4 degrees
N in tract
1500
1000
500
0
Shear matix = 1, 0, 0, 1

Modifying the lambda only

Adding a positive value to the the lambda variable will shift the values parallel to the X-axis.

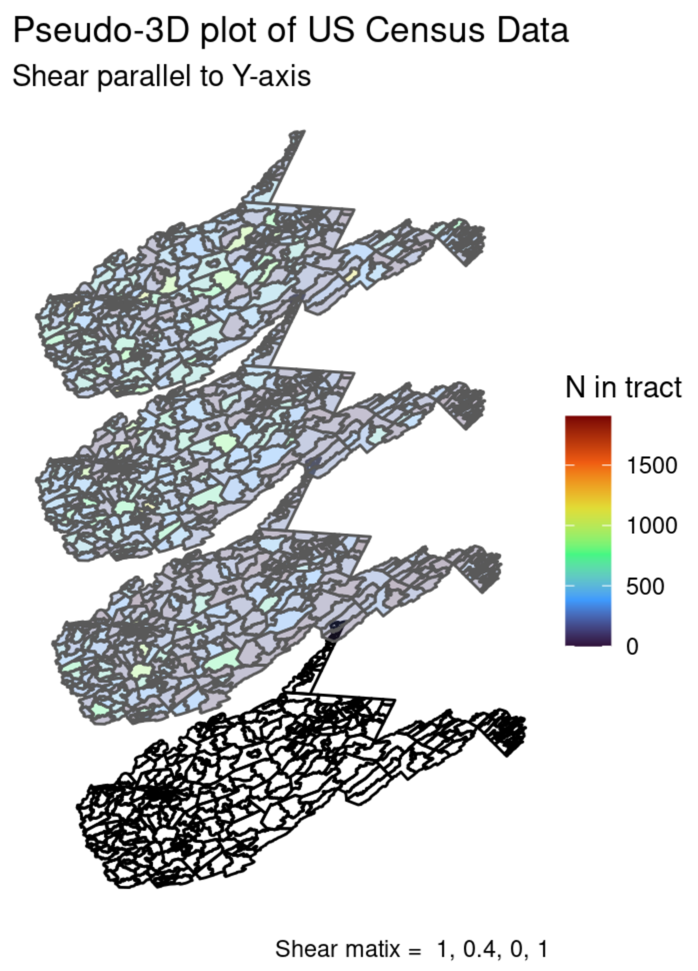
```r
new_shear <- c(1, 0, 0.4, 1) # Changing the matrix




ggplot() +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 8), fill =
"white", color = "black", alpha = 0.1) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  8, y_add
= 2, x_add = -0.3), aes(fill = lt05),
            alpha = 0.5) +
     geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  8, y_add
= 4, x_add = -0.5), aes(fill = gt5to99),
             alpha = 0.5) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 8, y_add =
6, x_add = -0.5), aes(fill = gt1to15),
            alpha = 0.5) +
    scale_fill_viridis_c(option = "turbo") +
    theme_void()  +
     labs(title = "Pseudo-3D plot of US Census Data",
         subtitle = "Shear parallel to X-axis",
         caption =  paste("Shear matix = " ,paste(new_shear, collapse = ", ")),
         fill = "N in tract")
```



Modifying mu only

This should shift the values parallel to the Y-axis.

```
new_shear <- c(1, 0.4, 0, 1)



ggplot() +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 4), fill =
"white", color = "black", alpha = 0.1) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  4, y_add
= 2, x_add = -0.3), aes(fill = lt05),
            alpha = 0.3) +
     geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  4, y_add
= 4, x_add = -0.5), aes(fill = gt5to99),
             alpha = 0.3) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 4, y_add =
6, x_add = -0.5), aes(fill = gt1to15),
            alpha = 0.3) +
    scale_fill_viridis_c(option = "turbo") +
    theme_void()  +
     labs(title = "Pseudo-3D plot of US Census Data",
         subtitle = "Shear parallel to Y-axis",
         caption =  paste("Shear matix = " ,paste(new_shear, collapse = ", ")),
         fill = "N in tract")
```

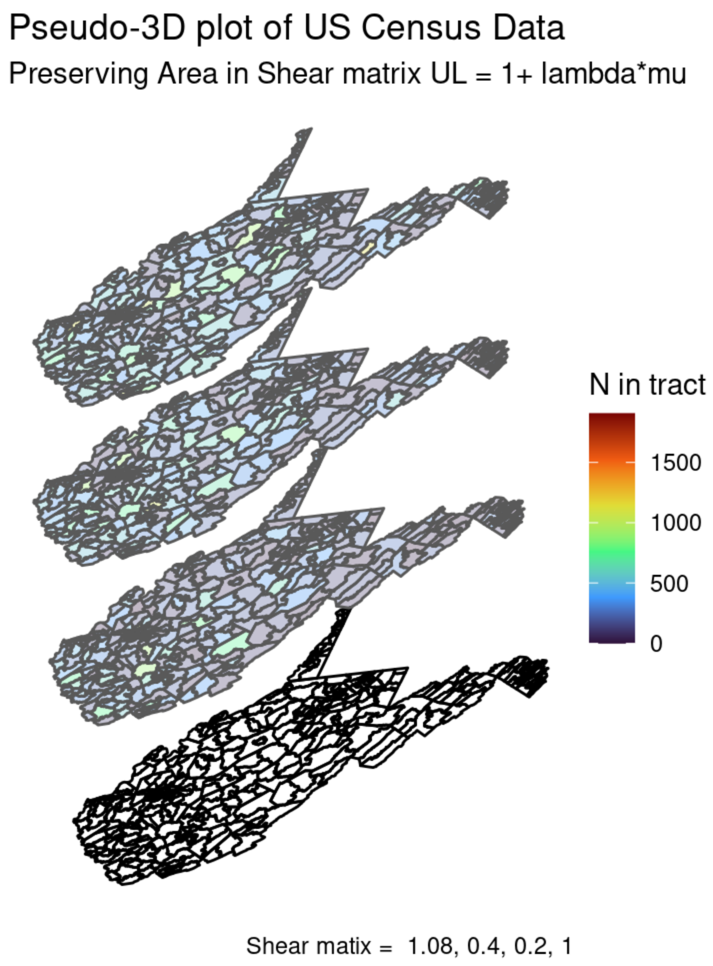


To combine both do a basic composition matrix of [1+(mu*lambda), mu, lambda, 1]. Although a better method maybe to “play around” with the values. To play around to get the desired projection may not preserve the area.

Modifing shear matrix to maintain equal area

This is done be setting the UL value of the shear matrix to 1 + (lambda * mu)

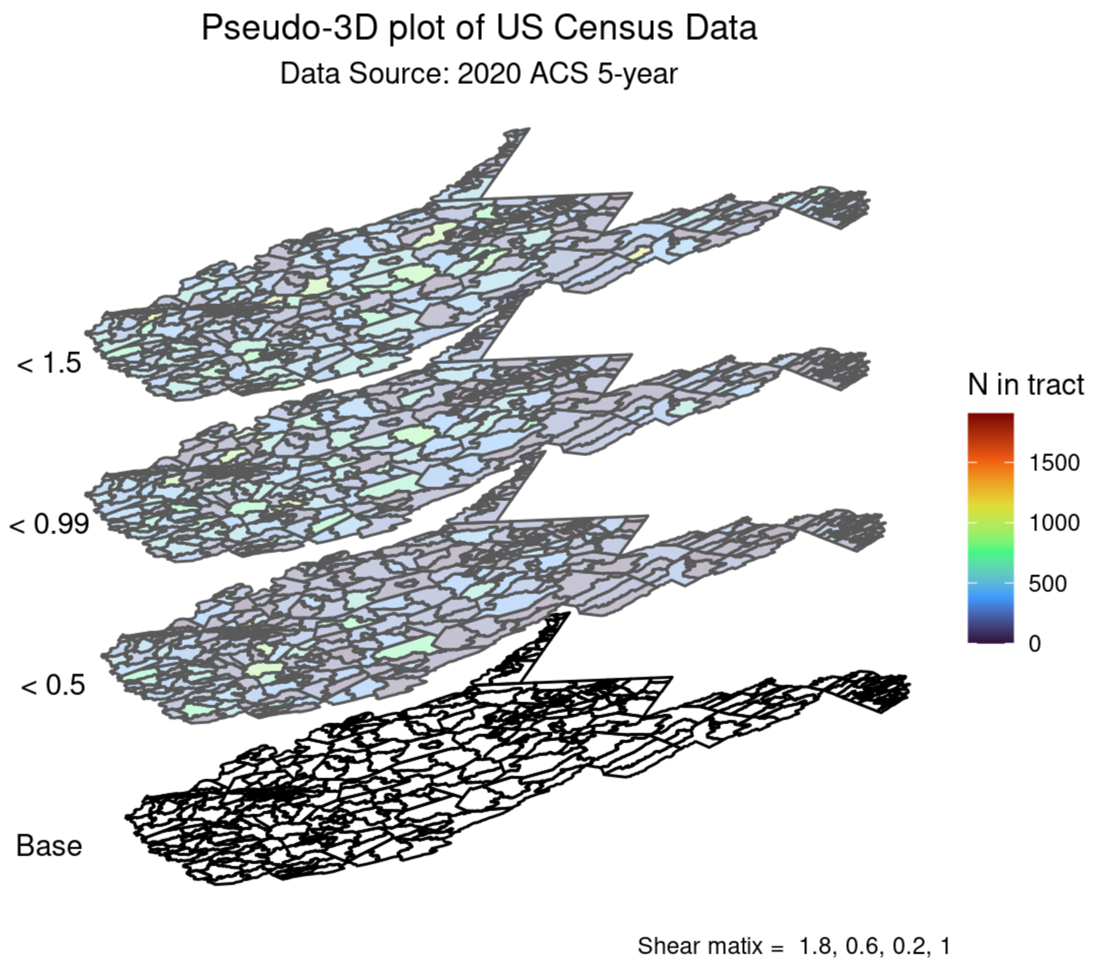
```
new_shear <- c(1.08, 0.4, 0.2, 1)




ggplot() +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 4), fill =
"white", color = "black", alpha = 0.1) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  4, y_add
= 2, x_add = -0.3), aes(fill = lt05),
            alpha = 0.3) +
     geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  4, y_add
= 4, x_add = -0.5), aes(fill = gt5to99),
             alpha = 0.3) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 4, y_add =
6, x_add = -0.5), aes(fill = gt1to15),
            alpha = 0.3) +
    scale_fill_viridis_c(option = "turbo") +
    theme_void() +
     labs(title = "Pseudo-3D plot of US Census Data",
         subtitle = "Preserving Area in Shear matrix UL = 1+ lambda*mu",
         caption =  paste("Shear matix = " ,paste(new_shear, collapse = ", ")),
         fill = "N in tract")
```


Pseudo-3D plot of US Census Data
Preserving Area in Shear matrix UL = 1+ lambda*mu
N in tract
1500
1000
500
0
Shear matix = 1.08, 0.4, 0.2, 1

In one of the iterations of a shear matrix the following seemed to work well. It slightly modified the upper left to be 1+(lambda + mu). This may not maitain the area but seemed to provides a good visualization of the state.

Modifying the shear matrix to provide “optimal” views.

```
new_shear <- c(1.8, 0.6, 0.2, 1)

ggplot() +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 4), fill =
"white", color = "black", alpha = 0.1) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  4, y_add
= 2, x_add = -0.3), aes(fill = lt05),
            alpha = 0.3) +
     geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta =  4, y_add
= 4, x_add = -0.5), aes(fill = gt5to99),
             alpha = 0.3) +
    geom_sf(data = rotate_sf(WV_poverty, shear_values = new_shear, theta = 4, y_add =
6, x_add = -0.5), aes(fill = gt1to15),
            alpha = 0.3) +
    scale_fill_viridis_c(option = "turbo") +
    theme_void() +
    theme(plot.title = element_text(hjust = 0.5),
          plot.subtitle = element_text(hjust = 0.5))  +
    annotate("text", x = -125, y = 30, label = "Base") +
    annotate("text", x = -125, y = 32, label = " < 0.5") +
    annotate("text", x = -125, y = 34, label = "< 0.99") +
    annotate("text", x = -125, y = 36, label = "< 1.5") +
    labs(title = "Pseudo-3D plot of US Census Data",
         subtitle = "Data Source: 2020 ACS 5-year",
         caption =  paste("Shear matix = " ,paste(new_shear, collapse = ", ")),
         fill = "N in tract")
```


Pseudo-3D plot of US Census Data
Data Source: 2020 ACS 5-year
< 1.5
< 0.99
< 0.5
Base
N in tract
1500
1000
500
0
Shear matix = 1.8, 0.6, 0.2, 1

Conclusion

The initial rotate_sf() by Junger and Cohen is an excellent function for adding “depth” to a 2D plot. This allows for separation of layer and create an plot that can display multiple values in a “stacked” versus “faceted” method. This version of the rotate_sf() function builds on their excellent work by adding user option to rotate based upon degrees that are later converted to rad. Additionally, this function allows the end user to change the shear matrix anc create a plot optimal for the data.

References

The web page that the has an excellent explanation plus the initial rotate_sf() prior adding basic functionality and changing the rotation calculation.

https://www.mzes.uni-mannheim.de/socialsciencedatalab/article/geospatial-data/#d-plots-1 (https://www.mzes.uni-mannheim.de/socialsciencedatalab/article/geospatial-data/#d-plots-1)

Fenglin Guo, Peng Hu, Shenyue Ji, Yiduo Bai, Xiaohong Xiao, and Yuping Wang “Design of pseudo-3D visualization in mobile GIS”, Proc. SPIE 6753, Geoinformatics 2007: Geospatial Information Science, 67530A (25 July 2007); https://doi.org/10.1117/12.761353 (https://doi.org/10.1117/12.761353)