ACS DATA CONVERSION FROM CENSUS

GEOGRAPHIES TO NEIGHBORHOOD

& COMMUNITY BOUNDARIES

A THESIS IN

Medical Bioinformatics

Presented to the faculty of the University.

of Missouri-Kansas City in partial fulfilment of the

requirements for the degree.

MASTER OF SCIENCE

by

MOUNIKA REDDY JAKKIDI

B.Pharm., Jawaharlal Nehru Technological University-HYD., 2018

Kansas City, Missouri

2023

ABSTRACT

To be written later

APPROVAL PAGE

To be written later

CONTENTS

ABSTRACT……………………………………………………………….......……..iii

LIST OF ILLUSTRATIONS……………………………………………….…...……vi

LIST OF TABLES……………………………………………………..………..…...vii

Chapter

1. INTRODUCTION………………………………………………………….…...1

2. REVIEW OF LITERATURE……………………………………………....…...3

3. METHODOLOGY………………………………………………………...…..14

4. RESULTS……………………………………………….………………….….18

5. DISCUSSION……………...……………………………………………….….19

Appendix

A. DATA DICTIONARY .........................................................................................20

B. PROGRAM CODE ...............................................................................................35

REFERENCE LIST ......................................................................................................87

ILLUSTRATIONS

Figure Page

1. Standard Hierarchy of Census Geographic Entities………………………….3
2. Digital Representation of Geographic Features………………………………11

TABLES

Table Page

1. Example of GEOID…………………………………………………………….8

CHAPTER 1

INTRODUCTION

Geographical information is essential while conducting a census. They don't just count people; they count people in certain geographic regions since it becomes the basis for conducting censuses and analyzing the data (Ratcliffe, 2019). While the Census Bureau creates its own geographic divisions, it also maintains a list of those that are utilized by other regional, state, and federal organizations. Using Census Bureau data successfully requires an understanding of spatial connections. The main source of demographic information for the United States Census Bureau is the American Community Survey (ACS), which is done on a regular basis every 10 years. In order to give a clear overview of the characteristics and needs of many regions across the country, this information is frequently merged into several geographic regions, such as census block groups, tracts, and counties (US Census Bureau, 2023).

R programming provides advanced geographic data analysis and visualization tools, including the distribution of ACS data from census regions to neighborhood and community districts. using R packages like "sf" and "dplyr" to transform the ACS data in order to reallocate data at the neighborhood and community levels. The "ggplot2" and "sf" functions may be used to create geographic visualizations showing the distribution of the data among neighborhoods after it has been reallocated (The most trusted IDE for Open Source Data Science, 2022).

Although ACS data is available for census geographies such as tracts and block groups, it is not available for blocks. However, there are communities and neighborhoods that require specific information about their geographic boundaries, which may not always align with the boundaries drawn by the Bureau of Census (Rossiter, 2021). To address this issue, we are using a weighting scheme to convert ACS data from one geography, such as (tract, or block group) and converting them into the different relevant boundaries for the community or neighborhood (Saalfeld, 1998). This is done by allocating the data based on how much the tract or block group falls inside or outside the specific community using a number of housing units or people in various blocks. This approach ensures that the data is appropriately allocated, providing a more accurate representation of the community’s population.

CHAPTER 2

LITERATURE REVIEW

Taking a census revolves around geography. They don’t just count people; they count people in specific geographic areas because it serves as the foundation for conducting censuses and tabulating their results (Rossiter, 2021). The Census Bureau develops its own geographic regions but also keeps track of certain geographic regions that are used by various regional, state and federal organizations. Understanding geographical links are essential for effectively utilizing Census Bureau data. The connections between the census Bureau’s maintained legal, administrative, and statistical borders are shown in the standard Hierarchy of Census Geographic Entities (Bureau, 2023).

Diagram

Description automatically generated

Figure1: Standard Hierarchy of Census Geographic Entities

**Blocks** are statistical regions that have physical boundaries like roads, rivers, and train tracks. All tabulated data from a decennial census is based on these. The United States is all covered by blocks. Blocks can be as small as a single city block or can potentially cover hundreds of square kilometres in rural regions(Rossiter, 2021). Blocks are given four-digit census block numbers ranging from 0000 to 9999 to identify them specifically inside census tracts located within states and counties(Bureau, 2021). The block group is identified by the first digit of the census block number. All blocks with water area are designed to have block numbers starting with a zero.

**Block Groups (BGs)**- used to show data and regulate block numbering- are statistical divisions of census tracts typically considered to comprise between 600 and 3000 individuals. A block group is a collection of blocks with the same first digit of their four-digit census block number located in the same census tract (Bureau, 2021). Block groups often cover a continuous region. At least one BG is present in every census tract and each BG has a specific number inside the tract. In conventional census geographic hierarchy, BGs may cross any other geographic entity’s boundaries but never those of a state, county, or census tract. The normal county-based census tracts and block groups established for the same region may not be at all similar to the tribal census tracts and block groups.

**Census Tracts**-Small, comparatively permanent statistical subdivisions of a county or statistically comparable units are known as census tracts(Bureau, 2021). For the purpose of presenting statistical data, census tracts serve as a reliable collection of geographic units. Populations in census tracts typically range from 1,200 to 8,000, with 4,000 being the ideal number. A census tract typically covers a single, continuous region, however, the size of a census tract can vary greatly depending on the density of the nearby population (Rossiter, 2021). In the typical census geographic hierarchy, state and county borders are always considered to be census tract borders. Tribal census tracts are special geographic units that are designated on federally recognized American Indian reservations and off-reservation trust lands(Kedron et al., 2019). The normal county-based census tracts established for the same region may not be at all similar to the tribal census tracts.

**Census Region**-For the purpose of presenting census statistics, the United States is divided into Census Regions, which are collections of states and the District of Columbia(Rossiter, 2021). The four census regions are the Northeast, the Midwest, the South, and the West. There are two or more census divisions in each of the four census regions. A single-digit census code serves as the identifier for each census area. Puerto Rico and the U.S. Territories do not fall under any census divisions or regions (Bureau, 2021). There are nine census divisions, and a single-digit census code is used to identify each one.

* Geographical entities are given codes by the Census Bureau and other federal agencies to make the organization, display, and sharing of statistical data and other information easier. Individual entities may be clearly identified according to geographic entity codes. Federal Information Processing Series (FIPS), National Standard (NS), and Census Bureau codes are among the several categories of geographic entity codes included in Census Bureau data products.0
* These codes were originally known as Federal Information Processing Standards (FIPS), or Federal Information Processing. For geographic entities governed by FIPS, the Census Bureau still maintains and issues codes (United States Department of Housing and Urban Development 2012). States, counties, congressional districts, core-based statistical areas, places, county subdivisions, sub-minor civil divisions, consolidated cities, estates, and all varieties of American Indian, Alaska Native, and Native Hawaiian regions are among the geographical entities included by FIPS.

**Core Based Statistical Areas** (CBSAs) are made up of the county or counties, or comparable entities, that are connected to at least one core (defined as an urban area by the Census Bureau) with at least 10 000 people, as well as any neighboring counties that are highly integrated socially and economically with the core as evidenced by commuting ties to the core-related counties. Metropolitan statistical areas and micropolitan statistical regions are referred to jointly as "core-based statistical areas."

Most states refer to their main geographical divisions as counties. An eight-digit National Standard (NS) code and a three-character alphabetically based Federal Information Processing Series (FIPS) code are given to each county or statistically comparable organization. The main divisions of counties and comparable entities are called county subdivisions. They can be categorized as either legal or statistical and include census county divisions, census subareas, minor civil divisions, and unorganized territories. A five-character Federal Information Processing Series (FIPS) number and an eight-digit National Standard (NS) code are given to each county subdivision based on the state's alphabetical order (American National Standards Institute (ANSI) and Federal Information Processing Series (FIPS) codes 2022).

**County**-In most states, counties are used to describe the fundamental legal divisions. These divisions are referred to as parishes in Louisiana. Alaska is a state without counties. One or more incorporated places that are apart from any county organization make up the principal divisions of four states (Maryland, Missouri, Nevada and Virginia). In order to improve the presentation of the statistics, these incorporated locations are know as independent cities which are recognized as equal entities(Bureau, 2021).

To offer comparable geographic units at the county level of the geographic hierarchy for these states, the counties in Massachusetts were all dissolved as operational governmental bodies. But the Census Bureau continues to publish data for these historical entities and portrays them as inoperative legal entities in data products. An eight-digit National Standard (NS) code and a three-character alphabetically based Federal Information Processing Series (FIPS) code are given to each county or statistically comparable organization(American National Standards Institute (ANSI) and Federal Information Processing Series (FIPS) codes 2022). The main divisions of counties and comparable entities are called county subdivisions. They can be categorized as either legal or statistical and include census county divisions, census subareas, minor civil divisions, and unorganized territories. A five-character Federal information processing series (FIPS) number and an eight-digit National Standard (NS) code are given to each county subdivision based on the state’s alphabetical order.

The Census Bureau, in collaboration with state, tribal and local agencies, draws the boundaries of census county divisions for statistical reasons. Census County Divisions are not governmental entities and have no legal purpose. Census tract borders and CCD boundaries often follow observable features. Each CCD’s name is derived from a locality, county, or well-known local name that designates its location.

Table 1: Example of GEOID

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Area Type | GEOID Structure | Number Of Digits | Example Of Geographic Area | Example GEOID |
| State | STATE | 2 | Kansas | 20 |
|  |
| County | STATE+COUNTY | 2+3=5 | Johnson County, KS | 20091 |  |
|  |
| Census Tract | STATE+COUNTY+TRACT | 2+3+6=11 | Census Tract 042302, Johnson County, KS | 20091042302 |  |
|  |
|  |
|  |
| Block Group | STATE+COUNTY+TRACT+ BLOCK GROUP | 2+3+6+1=12 | Block Group 3 in Census Tract 042302, Johnson County, KS. | 2.0091E+11 |  |
|  |
|  |
|  |
| Block\* | STATE+COUNTY+TRACT+ BLOCK GROUP+BLOCK | 2+3+6+4=15 (Note: some blocks also contain a one character suffix (A, B, C, etc.) | Block number is 001, Block Group 3 in Census Tract 042302, Johnson County, KS. | 2.0091E+14 |  |

Since the block group code is represented by the first digit of a census block code, the block group code is not included in the census block GEOID code.

**American Community Study (ACS)**: The American Community Study (ACS) is an annual survey that gathers data on social, economic, housing, and demographic aspects of our country and its citizens(Bureau, American Community Survey (ACS) 2023). Communities may utilize this information as a valuable tool to track their changes. By using the ACS form, people are assisting in making sure that the best data is used when making choices regarding the future of their community. Decision-makers need a complete picture of their population in order to distribute limited resources in an efficient and effective manner. Over 3.5 million homes are contacted nationwide by the Census Bureau each year to take part in the ACS. Due to their quick release in the year after they are gathered, ACS data are always very accurate.

The ACS produces periodic estimations, which are snapshots of the population and housing characteristics over a given time of data collection. These are the estimates for one year and five years.

**1-Year Estimates**

The 2005 ACS 1-Year Estimates were released for the first time in 2006. This is 12 months of data collection. Based on data gathered from the American Community Survey, these estimates include data for regions with populations of 65,000 or more, the information is based on numerous demographic, social, economic, and housing features of the people in the United States (ACS). These estimates are helpful for research, policymaking, and decision-making in a variety of sectors, including, but not limited to, public health, education, transportation, and urban planning.

**1-Year Additional Estimates**

The 1-Year Additional Estimates contain information on regions having a population of 20,000 or more people and are composed of condensed versions of frequently used ACS tables. These estimates, which cover the 2014 ACS 1-year supplemental estimates, were initially released in 2016. Researchers, politicians, and decision-makers in various disciplines, including public health, education, and transportation, can benefit from the estimations by receiving important information.5-Year Projections

**5-Year Projections**

Based on 60 months of data collecting, the ACS 5-Year Projections are a detailed set of projections that include all geographical areas. These estimates for the period 2005–2009 were first published in 2010. These estimates are very helpful for long-term planning, decision-making, and research in many areas, these estimates are very helpful (American Community survey 2017-2021 5-year - census.gov).

**Mapping Files:** Geographical data, which is spatial data recorded into a file format, is contained in GIS mapping files. The U.S. Census Bureau's geographic spatial data is represented by TIGER, which stands for Topologically Integrated Geographic Encoding and Referencing system.

Shapefiles (Partnership, TIGER/Line) are some examples.

**Shapefile**- Maps are made using shapefiles, which are digital representations of geographic features including lakes, landmarks, highways, and borders. The non-topological geometry and attribute data for the spatial features of a data collection are stored in a shapefile. Environmental Systems Research Institute\* (Esri) shapefile format is what the Census Bureau uses to produce shapefile layers(Esri Shapefile Reader/writer).

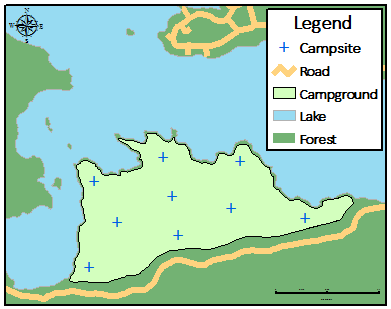


Figure2: Digital Representation of Geographic Features

**TIGER/LINE SHAPEFILES:** The Master Address File (MAF)/Topologically Integrated Geographic Encoding and Referencing (TIGER) System of the Census Bureau is where the TIGER/Line Shapefiles are extracted (MTS). The fifty states, the District of Columbia, Puerto Rico, and the Island regions (American Samoa, the Commonwealth of the Northern Mariana Islands, Guam, and the United States Virgin Islands) are all represented in the shapefiles. The shapefiles comprise point features, linear features like highways and hydrography, and polygon borders of geographic regions and features.

The TIGER/Line Files were originally made available in 1989. The American Standard Code for Information Interchange (ASCII) format fixed tables or record types contained in these files provide the first comprehensive street centerline coverage of the United States, Puerto Rico, and the Island Areas. The Census Bureau initially created the TIGER Database using the 1980 Geographic Base Files/Dual Independent Map Encoding (GBF/DIME Files), the U.S. Geological Survey (USGS) 1:100,000-scale Digital Line Graph (DLG), USGS 1:24,000-scale quadrangles, and a variety of other maps for selected areas outside the contiguous 48 states (predecessor to the current MTS). Throughout the 1990s and 2000s, the Census Bureau periodically issued ASCII 2-4 versions of the TIGER/Line Files(Walker, 2016). starting with the 2007 update, the shapefile file format replaced ASCII as the default for the TIGER/Line Files.

With the use of partner data, aerial photography, and fieldwork, the Census Bureau continuously updates and improves the MTS. Through a variety of partner initiatives, the Census Bureau receives updates on borders, features, and addresses from partners in the federal, state, and municipal governments (Her & Yu, 2021). To increase the geographic accuracy of the road network, the Census Bureau performed a significant reconfiguration of the MTS in the 2000s. The Census Bureau has since imposed quality requirements for data sources used to update the MTS.

In addition, shapefiles for the community districts and neighborhoods in the Kansas City metropolitan area. The two geographies, community district, and neighborhood have slightly different sources. The neighborhood geography was originally developed by the Kansas City Missouri planning department in the 1980s as part of the user-defined geography initiative of the US Census Department. These neighborhood boundaries, though contested at the margin in some places, were based on the pre-existing (pre-1980) social geography. The Wyandotte County neighborhoods have a similar history. The North Kansas City, Independence, and Raytown neighborhoods were developed as part of the KC Health CORE initiative. Slight changes were made to the North Kansas City shapefiles to make them continuous (minimal gaps, no overlaps).

The Community District geography was developed as part of the Center for Economic Information's neighborhood and urban development work. Following the work of the Chicago School of Urban Sociology (Burgess, McKenzie, etc.), CEI recognized the need for mid-level geography, between the neighborhood and city level, that was rooted in the living patterns of residents. The construction of the community district geography is discussed at length in Dr Bowles’s dissertation and a slightly modified version of the community district geography is used by the KCMO planning department.

CHAPTER 3

METHODOLOGY

**R-**R is a language and environment for visual design and statistical computation. It is a GNU project(GNU is an operating software which is free software and it respects user’s freedom) that is comparable to the S language and environment that John Chambers and colleagues created at Bell Laboratories (previously AT&T, now Lucent Technologies). R might be thought of as an alternative S implementation. Although there are some significant changes, much of the code created for S works well in R. R offers a wide range of graphical and statistical tools, including time-series analysis, classification, clustering, and linear and nonlinear modelling(What is R? 2012). It is also very extendable. R offers an Open-Source alternative for those interested in participating in statistical methods research, which frequently uses the S language as its preferred vehicle.

The simplicity with which well-designed charts of publication quality may be created using R, complete with mathematical symbols and equations when necessary is one of its strengths. The user still has complete power despite careful consideration being given to the visuals' minor design decisions' defaults. R is accessible as Free Software under the conditions of the GNU General Public License from the Free Software Foundation (Ihaka & Gentleman, 1996). On a wide range of UNIX platforms and related systems (including Linux), Windows, and MacOS, it builds and executes. R is frequently used as a statistics tool by users. It is considered a setting in which statistical methods are applied. Through packages, R can (simply) be expanded. The R distribution comes with roughly eight packages, and the CRAN family of websites offers many more that cover a fairly broad spectrum of contemporary statistics.

**R Studio-**R studio now Posit. An integrated development environment (IDE) for R and Python is called RStudio. A console, a syntax-highlighted editor with direct code execution support, and tools for graphing, history, debugging, and workspace management are all included. RStudio runs on desktop computers and is offered in both open-source and paid editions (Windows, Mac, and Linux). The following are some of the features that RStudio offers: local RStudio access, Code completion, clever indentation, syntax highlighting, and immediate running of R code from the source editor (The most trusted IDE for Open Source Data Science 2022). Go directly to the definition of the function. Using Projects, you can easily manage many working folders. R help and documentation are integrated. View Python data, render and publish Python content, and share Python objects with R. An interactive debugger is used to find and correct faults. Tools for package creation in plenty. Publishing applications and reports is simple.

**Packages:**

core tidyverse: It is likely to utilize these packages in regular data analysis because they are part of the core tidyverse (Campbell, 2019). The following packages are part of the core tidyverse as of version 1.3.0:

ggplot2 - The Grammar of Visuals is the foundation for ggplot2, a framework for declaratively constructing graphics. We should provide ggplot2 the data, instruct it on how to map variables to looks and what graphical primitives to employ, and it handles the rest.

dplyr- It offers a language for data manipulation, offering a standardized collection of verbs to address the most typical problems with data manipulation.

tidyr- We may access tidy data with the use of a number of functions provided by tidyr. Data that is organized and uniform in form is called tidy data.

readr-A quick and easy technique to read rectangular data is provided by readr (like csv, tsv, and fwf). It is built to flexibly analyze a variety of data kinds that are seen in the field, but also cleanly failing when data changes unexpectedly.

tibble-The data frame is modernized in tibble tibble by preserving what time has shown to work and discarding what has not. Tibbles are data frames that are slack and grumpy; they do less and moan more, requiring you to deal with issues sooner, usually resulting in cleaner, more expressive code (Campbell, 2019).

stringr-Working with strings is made as simple as possible with the help of stringr, which offers a unified collection of functions. It is developed on top of stringi, which offers quick, accurate implementations of common string manipulations using the ICU C library. Visit Docs.

tidycensus-With the help of the R package tidycensus, users may interact with a subset of the US Census Bureau's data APIs and receive tidyverse-ready data frames with the choice of basic feature geometry.

An R package called tidycensus (K. Walker and Herman 2021), which was originally made available in 2017, was created to make it easier to get and use US Census Bureau population data in the R environment (Bivand, 2022). The bundle has two different objectives. In order to facilitate the process of extracting insights from US Census data, tidycensus intends to first make Census data accessible to R users in a tidyverse-friendly manner. Second, the program is intended to make it easier for geographic Census data analysts to wrangle data. To aid with mapping and geographical analysis, tidycensus allows R users to request geometry along with characteristics for their Census data. Through its APIs and other data download sites, the US Census Bureau offers a wide variety of datasets to the user community.

sf- In late October 2016, the sf1 package was initially made available on CRAN. It implements the "Simple Features" formal standard, which outlines a paradigm for the storing and retrieval of spatial geometries like points, lines, and polygons (Pebesma, 2018). When a feature geometry satisfies specific requirements, it is said to be simple. Simple polygons, for instance, cannot have spikes or hanging vertices, nor can they self-intersect. Simple Characteristics lack explicit knowledge of their neighbors or other spatially related features and are independent (Machlis, 2019). Both web standards like GeoJSON and spatial databases like PostGIS have largely accepted this standard. A basic data frame with a particular column that holds the details for the geometry coordinates is how spatial objects are stored in sf. A list the same size as the number of rows in the data frame makes up that special column.

CHAPTER 4

RESULTS

To be written later

CHAPTER 5

DISCUSSION

To be written later

**APPENDIX - A**

Data dictionary (block)

---

Data Dictionary: tl\_2020\_xxxxx\_tabblock20.zip

authors: Satya Golla and Steve Simon

copyright:

Not directly stated, but this file should

be in the public domain.

creation date: Unknown

description:

Shapefiles for every blocks for every county

in the United States. Replace xxxxx with the

FIPS code for each county.

download:

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/20\_KANSAS/20091/tl\_2020\_20091\_tabblock20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/20\_KANSAS/20103/tl\_2020\_20103\_tabblock20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/20\_KANSAS/20209/tl\_2020\_20209\_tabblock20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/29\_MISSOURI/29037/tl\_2020\_29037\_tabblock20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/29\_MISSOURI/29047/tl\_2020\_29047\_tabblock20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/29\_MISSOURI/29095/tl\_2020\_29095\_tabblock20.zip

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows: varies by county

columns: 15

source:

U.S. Census Bureau

https://www.census.gov/

vars:

STATEFP20:

label: State FIPS code

scale: two digit integer with leading zero

COUNTYFP20:

label: County FIPS code

scale: three digit integer with leading zeros

TRACTCE20:

label: Census tract number

scale: six digit integer with leading zeros

BLOCKCE20:

label: Census block number

scale: four digit integer with leading zeros

GEOID20:

label: Geographic ID

scale: fifteen digit integer with leading zeros

NAME20:

label: Census block

scale: Alphanumeric block name

MTFCC20:

label: MAF/TIGER feature class code (G2200)

scale: five digits

values: <https://www2.census.gov/geo/pdfs/maps-> data/data/tiger/tgrshp2020/TGRSHP2020\_TechDoc\_E.pdf

UR20:

label: 2020 Census urban/rural indicator(2020 Urban Areas are not yet defined)

UACE20:

label: 2020 Census urban area code (2020 Urban Areas are not yet defined)

UATYPE20:

label: 2020 Census urban area type(2020 Urban Areas are not yet defined)

FUNCSTAT20:

label: 2020 Census functional status

values: Joint-use areas are identified uniquely by MTFCC G2170. An “A” in the functional status

(FUNCSTAT) field identifies federal AIR/ORTL joint-use areas, while an “S” in the field represents

joint-use OTSAs

ALAND:

label: Land area

unit: square meters

WATER:

label: Water area

unit: square meters

INTPTLAT:

label: Latitude of an interior point

INTPTLON:

label: Longitude of an interior point

---

Data dictionary (block group)

---

Data\_dictionary: tl\_2020\_xxxxx\_bg20.zip

authors: Satya Golla and Steve Simon

copyright:

Not directly stated, but this file should

be in the public domain.

creation\_date: Unknown

description:

Shapefiles for every block groups for every county

in the United States. Replace xxxxx with the

FIPS code for each county.

download:

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/20\_KANSAS/20091/tl\_2020\_20091\_bg20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/20\_KANSAS/20103/tl\_2020\_20103\_bg20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/20\_KANSAS/20209/tl\_2020\_20209\_bg20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/29\_MISSOURI/29037/tl\_2020\_29037\_bg20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/29\_MISSOURI/29047/tl\_2020\_29047\_bg20.zip

https://www2.census.gov/geo/tiger/TIGER2020PL/STATE/29\_MISSOURI/29095/tl\_2020\_29095\_bg20.zip

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows: varies by county

columns: 12

source:

U.S. Census Bureau

https://www.census.gov/

vars:

STATEFP20:

label: State FIPS code

scale: two digit integer with leading zero

COUNTYFP20:

label: County FIPS code

scale: three digit integer with leading zeros

TRACTCE20:

label: Census tract number

scale: six digit integer with leading zeros

BLOCKCE20:

label: Census block number

scale: four digit integer with leading zeros

GEOID20:

label: Geographic ID

scale: fifteen digit integer with leading zeros

NAME20:

label: Census block

scale: Alphanumeric block name

MTFCC20:

label: MAF/TIGER feature class code (G2200)

scale: five digit

values: https://www2.census.gov/geo/pdfs/maps-data/data/tiger/tgrshp2020/TGRSHP2020\_TechDoc\_E.pdf

FUNCSTAT20:

label: 2020 Census functional status

values: Joint-use areas are identified uniquely by MTFCC G2170. An “A” in the functional status

(FUNCSTAT) field identifies federal AIR/ORTL joint-use areas, while an “S” in the field represents

joint-use OTSAs

ALAND:

label: Land area

unit: square meters

WATER:

label: Water area

unit: square meters

INTPTLAT:

label: Latitude of an interior point

INTPTLON:

label: Longitude of an interior point

---

Data dictionary (tract)

---

data\_dictionary: tr.RData

authors: Satya Golla and Steve Simon

copyright:

Not directly stated, but this file should

be in the public domain.

creation\_date: Unknown

description:

Simple feature (sf) for tracts in a seven county region

of the Kansas City metropolitan area. Converted from

census shapefiles described in tract-data-dictionary.

format:

proprietary: Simple Feature (sf)

missing-value-code: not needed

rows: 570

columns: 4

source:

U.S. Census Bureau and modified by

store-tract-information.Rmd

vars:

tr\_id

label: Tract identifier; a concatenation of state, county and tract FIPS code

scale: eleven digit integer with leading zeros

tr\_name

label: Tract name

tr\_area

label: Tract area

unit: square meters

geometry:

label: Simple feature (sf) geometry

---

Data dictionary (county)

---

data\_dictionary: tl\_2021\_us\_county.zip

authors: Satya Golla and Steve Simon

copyright:

Not directly stated, but this file should

be in the public domain.

creation\_date: Unknown

description:

Shapefiles for every county in the United

States.

download:

https://www2.census.gov/geo/tiger/TIGER2021/COUNTY/

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows:

columns: 17

source:

U.S. Census Bureau

vars:

STATEFP:

label: State FIPS code

scale: two digit integer with leading zero

COUNTYFP:

label: County FIPS code

scale: three digit integer with leading zeros

COUNTYNS:

label: Current county GNIS code

scale:

GEOID:

label: (Geographic ID) County identifier; a concatenation of Current state FIPS code and county FIPS code

NAME:

label: Current county name

NAMELSAD:

label: (Longer county name)Current name and the translated legal/statistical area description for county

LSAD:

label: Current legal/statistical area description code for consolidated city

CLASSFP:

label: Current FIPS class code

MTFCC:

label: MAF/TIGER feature class code (G4020)

CSAFP:

label: Current combined statistical area code

CBSAFP:

label: Current metropolitan statistical area/micropolitan statistical

area code

METDIVFP:

label: Current metropolitan division code

FUNCSTAT:

label: Current functional status

ALAND:

label: Land area

unit: square meters

WATER:

label: Water area

unit: square meters

INTPTLAT:

label: Latitude of an interior point

INTPTLON:

label: Longitude of an interior point

chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www2.census.gov/geo/pdfs/maps-data/data/tiger/tgrshp2019/TGRSHP2019\_TechDoc.pdf

---

Data dictionary (community districts) **:**

---

data\_dictionary: Community districts.zip

authors: Mounika Reddy Jakkidi and Steve Simon

copyright: Uncertain.

creation date: Unknown

description:

Shapefiles for community districts in the

Kansas City metropolitan area.

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows: 59

columns: 4

source:

See doc/email-wilson-2022-03-03.pdf

vars:

cd\_id:

label: Sequential id

cd\_Name:

label: Name of community district

cd\_area:

label:Area of community district

geometry:

label:sfc\_polygon(object of POLYGON geometries)

---

Data dictionary (community districts weights)**:**

---

data\_dictionary: Community districts.zip

authors: Mounika Reddy Jakkidi and Steve Simon

copyright: Uncertain.

creation\_date: Unknown

description:

Shapefiles for community districts in the

Kansas City metropolitan area.

source:

See doc/email-Wilson-2022-03-03.pdf

bg\_counts:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows: 1236

columns: 6

vars:

bg\_id:

label: block group id

cd\_id:

label: community district id

people\_in:

label: people inside census block

Units\_in:

label: housing units inside the block

people:

label: number of people

units:

label: number of housing units

bl\_counts:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows: 18,864

columns: 9

source:

See doc/email-wilson-2022-03-03.pdf

vars:

bl\_id:

label: block id

cd\_Name:

label: Name of the community district

cd\_id:

label: Sequential id

bl\_prop\_in:

label: proportion of block inside community district

people:

label: number of people

people\_in:

label: people inside census block

units:

label: number of housing units

Units\_in:

label: housing units inside a block

bl\_list:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows:17,547

columns:5

vars:

bg\_id:

label: block group id

bl\_id:

label: block id

tr\_id:

label: tract id

people:

label: number of people

people\_in:

label: people inside census block

units:

label: number of housing units

---

Data dictionary (community district intersection)**:**

---

data\_dictionary: Community districts.zip

authors: Mounika Reddy Jakkidi and Steve Simon

copyright: Uncertain.

creation\_date: Unknown

description:

Shapefiles for community districts in the

Kansas City metropolitan area.

source:

See doc/email-Wilson-2022-03-03.pdf

bg\_cd\_intersection:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows: 1236

columns: 4

vars:

bg\_id:

label: block group id

cd\_name:

label:label: Name of the community district

cd\_id:

label: community district id

bg\_prop\_in:

label: proportion of block group inside

bl\_cd\_intersection:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows: 15,748

columns: 5

vars:

bg\_id:

label: block group id

bl\_id:

label: block id

cd\_id:

label: community district id

cd\_name:

label: Name of the community district

bl\_prop\_in:

label: proportion of block inside

---

Data dictionary (neighborhoods)**:**

---

data\_dictionary: neighborhoods.zip

authors: Mounika Reddy Jakkidi,Satya Golla and Steve Simon

copyright: Uncertain.

creation\_date: Unknown

description:

Shapefiles for neighborhoods in the

Kansas City metropolitan area.

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows: 375

columns: 10

source:

See doc/email-wilson-2022-03-03.pdf

vars:

NID:

label: Numeric id

AreaName:

label: Name of neighborhood

nbr\_id:

label: Sequential id

community district:

label: Which community district this belongs to

id:

label: unique id field

label\_long:

label: Unknown

shid:

label: Unknown

id **for** Shapefile

Shape\_Length:

label: Perimeter of neighborhood

unit: Unknown

Shape\_Area:

label: Area of neighborhood

unit: Unknown

---

Data dictionary (neighborhoods weights)**:**

---

data\_dictionary: neighborhoods weights.zip

authors: Mounika Reddy Jakkidi and Steve Simon

copyright: Uncertain.

creation\_date: Unknown

description:

calculating census blocks intersect with which neighbourhoods.

source:

See doc/email-wilson-2022-03-03.pdf

bg\_counts:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows:2048

columns:6

vars:

bg\_id:

label: block group id

NID:

label: Numeric id

people\_in:

label: people inside census block

Units\_in:

label:housing units inside block

people:

label:number of people

units:

label:number of housing units

bl\_counts:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows:25,336

columns:9

vars:

bg\_id:

label: block group id

bl\_id:

label:block id

NID:

label: Numeric id

AreaName:

label: Name of neighborhood

bl\_prop\_in:

label: proportion of block inside

people:

label: number of people

people\_in:

label: people inside census block

units:

label: number of housing units

Units\_in:

label: housing units inside a block

bl\_list:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows:19,859

columns:5

vars:

bg\_id:

label: block group id

bl\_id:

label: block id

tr\_id:

label: tract id

people:

label: number of people

units:

label: number of housing units

tr\_counts:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows:1246

columns:6

NID:

label: Numeric id

tr\_id:

label: tract id

people:

label: number of people

people\_in:

label: people inside census block

units:

label: number of housing units

Units\_in:

label: housing units inside block

---

Data dictionary (neighborhoods intersection)**:**

---

data\_dictionary: neighborhoods intersection.zip

authors: Mounika Reddy Jakkidi and Steve Simon

copyright: Uncertain.

creation\_date: Unknown

description:

blocks that partially or completely intersect with neighborhoods

source:

See doc/email-Wilson-2022-03-03.pdf

bg\_nbd\_intersection:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows:2657

columns:4

vars:

bg\_id:

label: block group id

NID:

label: Numeric id

AreaName:

label: Name of neighborhood

bg\_prop\_in:

label: proportion of block group inside neighborhood

bl\_nbd\_intersection:

format:

proprietary: Shapefile (zip)

missing-value-code: not needed

rows:22,324

columns:5

vars:

bg\_id:

label: block group id

bl\_id:

label: block id

AreaName:

label: Name of neighborhood

NID:

label: Numeric id

bl\_prop\_in:

label: proportion of block inside neighborhood

---

**APPENDIX - B**

CODE:  
Download acs data:  
---

title: "Download ACS data using tidycensus"

author: "Satya Golla, Mounika Reddy,and Steve Simon"

date: "Created 2022-09-04"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program downloads information from

the American Community Survey and displays

it on a map.

information about variables in the ACS can be found here

https://api.census.gov/data/2020/acs/acs5/groups.html

### Load relevant files

```{r setup}

# Important abbreviations:

# bg = block group

# bl = block

# cd = community district

# co = county

# nbd=neighborhood

# tr = tract

library(glue)

library(magrittr)

library(sf)

library(tidyverse)

library(tidycensus)

library(dplyr)

path\_name <- "../data/"

```

### Contents of bg

```{r}

load(glue("{path\_name}bg.RData"))

glimpse(bg)

```

### Contents of bl

```{r}

load(glue("{path\_name}bl.RData"))

glimpse(bl)

```

### Contents of cd

```{r}

load(glue("{path\_name}cd.RData"))

glimpse(cd)

```

### Contents of co

```{r}

load(glue("{path\_name}co.RData"))

glimpse(co)

```

### Subset co to seven counties

```{r}

clist <- c(

"20091",

"20103",

"20209",

"29037",

"29047",

"29095",

"29165")

co %>%

filter(GEOID %in% clist) -> co

glimpse(co)

```

### Get id values for all community districts

```{r}

cd %>%

tibble %>%

distinct(cd\_id) %>%

arrange(cd\_id) %>%

pull(cd\_id) -> cd\_list

```

### Pull state information from ACS

The examples here are loosely based on

the [basic usage vignette][walk1] for tidycensus.

[walk1]: https://walker-data.com/tidycensus/articles/basic-usage.html

### Housing units--occupied and unoccupied

```{r}

# Note: Data Profiles (DP) are available in the five year

# ACS data only down to the tract level.

vlist<- c("DP04\_0016", "DP04\_0024", "DP04\_0025", "DP04\_0026")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "state",

variables = vlist,

year = 2020)

get\_acs(

geography = "county",

variables = vlist,

state="MO",

year = 2020)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

### B01001 tables

https://pearson.socialexplorer.com/data/ACS2016/metadata/?ds=SE&table=B01001

```{r}

# Note: Detail tables (B) are available in the five year

# ACS data down to the census block group level.

vlist <- c("B01001\_003", "B01001\_027")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

### B13002 tables

https://api.census.gov/data/2020/acs/acs5/groups/B13002.html

```{r}

# But some estimates are still only available in the

# five year ACS survey at the tract level or higher.

vlist <- c("B13002\_001", "B13002\_002", "B13002\_011")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B21001"tables

https://data.census.gov/cedsci/table?q=Cedar**%**20Rapids**%**20veterans&tid=ACSDT5Y2020.B21001

```{r}

vlist <- c("B21001\_001", "B21001\_002", "B21001\_003")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B25001" tables

https://data.census.gov/cedsci/table?q=housing%20units&tid=ACSDT5Y2020.B25001

```{r}

vlist <- c("B25001\_001", "B25002\_001", "B25002\_002", "B25002\_003")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B25002"

https://data.census.gov/cedsci/table?q=Table%20B25002&tid=ACSDT1Y2019.B25002&hidePreview=true

```{r}

vlist <- c("B25002\_002", "B25003\_001", "B25003\_002", "B25003\_003")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B25124"

https://api.census.gov/data/2020/acs/acs5/groups/B25124.html

```{r}

vlist <- c("B25124\_002", "B25124\_003", "B25124\_010", "B25124\_017", "B25124\_024", "B25125\_031")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

```{r}

vlist <- c("B25124\_038", "B25124\_039", "B25124\_046", "B25124\_053", "B25124\_060", "B25124\_067")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B28001"

https://data.census.gov/cedsci/all?q=B28001&d=ACS**%**201-Year**%**20Estimates**%**20Detailed**%**20Tables

```{r}

vlist <- c("B28001\_001", "B28001\_002", "B28001\_011")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B25003"

https://api.census.gov/data/2020/acs/acs5/groups/B25003.html

```{r}

vlist <- c("B25003\_001E", "B25003\_002E", "B25003\_003E")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B09001"

https://api.census.gov/data/2020/acs/acs5/groups/B09001.html

```{r}

vlist <- c("B09001\_001", "B09001\_002", "B09001\_003","B09001\_005")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B17001"

https://api.census.gov/data/2020/acs/acs5/groups/B17001.html

```{r}

vlist <- c("B17001\_001", "B17001\_002", "B17001\_003","B17001\_017","B17001\_018")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B27001" tables

https://api.census.gov/data/2020/acs/acs5/groups/B27001.html

```{r}

vlist <- c("B27001\_001", "B27001\_002", "B27001\_030")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

###"B992701"tables

https://api.census.gov/data/2020/acs/acs5/groups/B992701.html

```{r}

vlist <- c("B992701\_001", "B992701\_002", "B992701\_003")

load\_variables(2020, "acs5", cache = TRUE) %>%

filter(name %in% vlist)

get\_acs(

geography = "tract",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

get\_acs(

geography = "cbg",

variables = vlist,

state="MO",

county="Jackson",

year = 2020)

```

Store block information:

---

title: "Store block information"

author: "Satya Golla and Steve Simon"

date: "Created 2022-03-12"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will read in census shapefiles for census blocks, convert them into the simple features (sf) format, and store the results for later use.

### General information

The starting point for downloading most Census shapefiles is [here][c\_general]. The ftp site is a bit more difficult to navigate, but leads you directly to the files that you want.

It helps to know the FIPS (Federal Information Processing System) codes **for** Kansas (20) and Missouri (29).

Files with a postfix of "20" either in the filename or among some of the variable names indicates that the data or shapefile is from the 2020 Census. A postfix of "10" indicates that the data or shapefile is from the 2010 Census.

The important counties for this program are

Cass County, MO (29037)

Clay County, MO (29047)

Jackson County, MO (29095)

Johnson County, KS (20091)

Leavenworth County, KS (20103)

Platte County, MO (29165)

Wyandotte County, KS (20209)

Some of the variables in the shape files are rather cryptic. An explanation of MTFCC20 appears [here][mtfcc].

### US census blocks

A census block is the smallest geographical unit used by the United States Census Bureau for tabulation of 100-percent data. There are files for census blocks for individual counties in each state.

Use the st\_read function from the [sf package][r\_sf] to read in the data. You may need to convert the coordinate reference system. That uses the st\_crs function, also of the sf package. Census blocks are uniquely identified by a two digit state FIPS code, a three digit county FIPS code, a six digit tract code, and a four digit block code. These are combined into a fifteen digit code, GEOID.

### Read block shapefiles for each county

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../data"

```

### Johnson County, KS (20091)

```{r read-johnson-county}

file\_name <- "bl/tl\_2020\_20091\_tabblock20.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> s20091

glimpse(s20091)

s20091 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Leavenworth County, KS (20103)

```{r read-leavenworth-county}

file\_name <- "bl/tl\_2020\_20103\_tabblock20.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> s20103

glimpse(s20103)

s20103 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Miami County, KS (20121)

Miami County was included **in** earlier programs, but it is not needed **for** any of the work envisioned **in** this project.

### Wyandotte County, KS (20209)

```{r read-wyandotte-county}

file\_name <- "bl/tl\_2020\_20209\_tabblock20.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> s20209

glimpse(s20209)

s20209 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Cass County, MO (29037)

```{r read-cass-county}

file\_name <- "bl/tl\_2020\_29037\_tabblock20.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> s29037

glimpse(s29037)

s29037 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Clay County, MO (29047)

```{r read-clay-county}

file\_name <- "bl/tl\_2020\_29047\_tabblock20.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> s29047

glimpse(s29047)

s29047 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Jackson County, MO (29095)

```{r read-jackson-county}

file\_name <- "bl/tl\_2020\_29095\_tabblock20.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> s29095

glimpse(s29095)

s29095 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Platte County, MO (29165)

```{r read-platte-county}

file\_name <- "bl/tl\_2020\_29165\_tabblock20.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> s29165

glimpse(s29165)

s29165 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Combine files

```{r combine}

s20091 %>%

bind\_rows(s20103) %>%

bind\_rows(s20209) %>%

bind\_rows(s29037) %>%

bind\_rows(s29047) %>%

bind\_rows(s29095) %>%

bind\_rows(s29165) -> bl

```

### Rename variables

```{r rename}

bl %>%

mutate(bl\_id=substr(GEOID20, 1, 15)) %>%

rename(bl\_name=NAME20) %>%

mutate(

bl\_area=

as.numeric(st\_area(bl))) %>%

select(

bl\_id,

bl\_name,

bl\_area,

geometry) -> bl

glimpse(bl)

```

### Store as an RData file

Placing the file **in** an RData format will speed up future use of this file.

```{r save}

save(bl, file="../data/bl.RData")

```

An alternate set of shape files appear under the names

+ TIGER2020/TABBLOCK/tl\_2020\_xx\_tabblock10.zip

These are files from the 2010 Census.

<!---Key census links go here--->

[r\_sf]: https://r-spatial.github.io/sf/

[r read\_acs]: https://cran.r-project.org/package=acs

[c\_general]: https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html

[c\_counties]: https://www2.census.gov/geo/tiger/TIGER2021/COUNTY/tl\_2021\_us\_county.zip

[c\_puma\_ks]: https://www2.census.gov/geo/tiger/TIGER2021/PUMA/tl\_2021\_20\_puma10.zip

[c\_puma\_mo]:https://www2.census.gov/geo/tiger/TIGER2021/PUMA/tl\_2021\_29\_puma10.zip

[c\_tract\_ks]: https://www2.census.gov/geo/tiger/TIGER2021/TRACT/tl\_2021\_20\_tract.zip

[c\_tract\_mo]: https://www2.census.gov/geo/tiger/TIGER2021/TRACT/tl\_2021\_29\_tract.zip

[c\_bg]: https://www2.census.gov/geo/tiger/TIGER2021/BG/

[c\_bg\_ks]: https://www2.census.gov/geo/tiger/TIGER2021/BG/tl\_2021\_20\_bg.zip

[c\_bg\_mo]: https://www2.census.gov/geo/tiger/TIGER2021/BG/tl\_2021\_29\_bg.zip

Insert links **for** census block groups and census blocks.

[c\_redistricting]: https://www.census.gov/programs-surveys/decennial-census/about/rdo/summary-files.html

[c\_help]: https://www.census.gov/data/academy/courses/2020-census-redistricting-data.html

[c\_redistricting\_ks]: https://www2.census.gov/programs-surveys/decennial/2020/data/01-Redistricting\_File--PL\_94-171/Kansas/ks2020.pl.zip

[c\_redistricting\_mo]: https://www2.census.gov/programs-surveys/decennial/2020/data/01-Redistricting\_File--PL\_94-171/Missouri/mo2020.pl.zip

[c\_r\_code]: https://www2.census.gov/programs-surveys/decennial/rdo/about/2020-census-program/Phase3/SupportMaterials/2020PL\_R\_import\_scripts.zip

[mtfcc]: https://www2.census.gov/geo/pdfs/maps-data/data/tiger/tgrshp2009/TGRSHP09AF.pdf

---

Store block group information:

---

title: "Store block group information"

author: "Satya Golla and Steve Simon"

date: "Created 2022-03-14"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will read in census block group files and store them in an RData file for later use.

### General information

A census block group is a subdivision of a Census tract. These can be further subdivided into Census blocks. Census blocks are uniquely identified by a two digit state FIPS code, a three digit county FIPS code, a six digit tract code, and a single digit block group code. These are combined into a twelve digit code, GEOID.

The starting point for downloading most Census shapefiles is [here][c\_general]. The ftp site is a bit more difficult to navigate, but leads you directly to the files that you want.

There are files for census block groups for [Kansas][c\_bg\_ks] and [Missouri][c\_bg\_mo].

[c\_general]: https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html

[c\_bg\_ks]: https://www2.census.gov/geo/tiger/TIGER2021/BG/tl\_2021\_20\_bg.zip

[c\_bg\_mo]: https://www2.census.gov/geo/tiger/TIGER2021/BG/tl\_2021\_29\_bg.zip

The important counties for this program are

Cass County, MO (29037)

Clay County, MO (29047)

Jackson County, MO (29095)

Johnson County, KS (20091)

Leavenworth County, KS (20103)

Platte County, MO (29165)

Wyandotte County, KS (20209)

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../data"

```

### Kansas block groups

```{r read-ks}

co\_list <- c(

"091",

"103",

"209")

file\_name <- "bg/tl\_2021\_20\_bg.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) %>%

filter(COUNTYFP %in% co\_list) -> bg\_ks

glimpse(bg\_ks)

bg\_ks %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Missouri block groups

```{r read-mo}

co\_list <- c(

"037",

"047",

"095",

"165")

file\_name <- "bg/tl\_2021\_29\_bg.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) %>%

filter(COUNTYFP %in% co\_list) -> bg\_mo

glimpse(bg\_mo)

bg\_mo %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Combine Kansas and Missouri

```{r combine-block-groups}

bg\_ks %>%

bind\_rows(bg\_mo) %>%

rename(bg\_id=GEOID) %>%

rename(bg\_name=NAMELSAD) -> bg

bg %>%

mutate(

bg\_area=

as.numeric(st\_area(bg))) %>%

select(

bg\_id,

bg\_name,

bg\_area,

geometry) -> bg

glimpse(bg)

bg %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("mo-ks-tracts")

```

### Store as RData

```{r save}

save(bg, file="../data/bg.RData")

```

Store tract information:

---

title: "Store tract information"

author: "Satya Golla and Steve Simon"

date: "Created 2022-03-14"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

```{r setup}

library(sf)

library(tidyverse)

```

### General information

### Census tracts

A Census tract is a region usually within a single county that contains roughly 4,000 people.

The starting point **for** downloading most Census shapefiles is [here][c\_general]. There are files **for** census tracts **for** [Kansas][c\_tract\_ks] and [Missouri][c\_tract\_mo].

[c\_general]: https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html

[c\_tract\_ks]: https://www2.census.gov/geo/tiger/TIGER2021/TRACT/tl\_2021\_20\_tract.zip

[c\_tract\_mo]: https://www2.census.gov/geo/tiger/TIGER2021/TRACT/tl\_2021\_29\_tract.zip

The important counties **for** this program are

Cass County, MO (29037)

Clay County, MO (29047)

Jackson County, MO (29095)

Johnson County, KS (20091)

Leavenworth County, KS (20103)

Platte County, MO (29165)

Wyandotte County, KS (20209)

```{r read-tract-ks}

co\_list <- c(

"091",

"103",

"209")

path\_name <- "../data"

file\_name <- "tr/tl\_2021\_20\_tract.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) %>%

filter(COUNTYFP %in% co\_list) -> tr\_ks

glimpse(tr\_ks)

tr\_ks %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("ks-tract")

```

```{r read-tract-mo}

co\_list <- c(

"037",

"047",

"095",

"165")

path\_name <- "../data"

file\_name <- "tr/tl\_2021\_29\_tract.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) %>%

filter(COUNTYFP %in% co\_list) -> tr\_mo

glimpse(tr\_mo)

tr\_mo %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("mo-tract")

```

```{r combine-tracts}

### Combine Kansas and Missouri

tr\_ks %>%

bind\_rows(tr\_mo) %>%

rename(tr\_id=GEOID) %>%

rename(tr\_name=NAMELSAD) -> tr

tr %>%

mutate(

tr\_area=

as.numeric(st\_area(tr))) %>%

select(

tr\_id,

tr\_name,

tr\_area,

geometry) -> tr

glimpse(tr)

tr %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("mo-ks-tracts")

```

### Store as RData

```{r save}

save(tr, file="../data/tr.RData")

```

Store redistricting information:

---

title: "Store redistricting information"

author: "Satya Golla and Steve Simon"

date: "Created 2022-03-08."

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will read **in** census redistricting files

which provide population and housing counts **for**

every census block **in** every state **in** the U.S.

### General information

The starting point **for** data and information about

the redistricting files is [here][c\_red].

This program is based on pl\_geohd\_2020\_dar.r,

which is part of [this zip file][c\_rcode].

[c\_red]: https://www.census.gov/programs-surveys/decennial-census/about/rdo/summary-files.html

[c\_rcode]: https://www2.census.gov/programs-surveys/decennial/rdo/about/2020-census-program/Phase3/SupportMaterials/2020PL\_R\_import\_scripts.zip

An important contact at City Hall **in** Kansas City, Missouri is Steve Lebowski. He has done similar work every time new Census information is available.

### Setup

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../data/"

load(paste0(path\_name, "bl", ".RData"))

load(paste0(path\_name, "nbd", ".RData"))

vnames <- c(

"FILEID",

"STUSAB",

"SUMLEV",

"GEOVAR",

"GEOCOMP",

"CHARITER",

"CIFSN",

"LOGRECNO",

"GEOID",

"GEOCODE",

"REGION",

"DIVISION",

"STATE",

"STATENS",

"COUNTY",

"COUNTYCC",

"COUNTYNS",

"COUSUB",

"COUSUBCC",

"COUSUBNS",

"SUBMCD",

"SUBMCDCC",

"SUBMCDNS",

"ESTATE",

"ESTATECC",

"ESTATENS",

"CONCIT",

"CONCITCC",

"CONCITNS",

"PLACE",

"PLACECC",

"PLACENS",

"TRACT",

"BLKGRP",

"BLOCK",

"AIANHH",

"AIHHTLI",

"AIANHHFP",

"AIANHHCC",

"AIANHHNS",

"AITS",

"AITSFP",

"AITSCC",

"AITSNS",

"TTRACT",

"TBLKGRP",

"ANRC",

"ANRCCC",

"ANRCNS",

"CBSA",

"MEMI",

"CSA",

"METDIV",

"NECTA",

"NMEMI",

"CNECTA",

"NECTADIV",

"CBSAPCI",

"NECTAPCI",

"UA",

"UATYPE",

"UR",

"CD116",

"CD118",

"CD119",

"CD120",

"CD121",

"SLDU18",

"SLDU22",

"SLDU24",

"SLDU26",

"SLDU28",

"SLDL18",

"SLDL22",

"SLDL24",

"SLDL26",

"SLDL28",

"VTD",

"VTDI",

"ZCTA",

"SDELM",

"SDSEC",

"SDUNI",

"PUMA",

"AREALAND",

"AREAWATR",

"BASENAME",

"NAME",

"FUNCSTAT",

"GCUNI",

"POP100",

"HU100",

"INTPTLAT",

"INTPTLON",

"LSADC",

"PARTFLAG",

"UGA")

```

### Read Kansas redistricting files

Setting SUMLEV=="750" produces information **for** individual census blocks. Setting it to SUM=="150" produce information **for** individual block groups.

```{r read-ks}

file\_name <- "red/ksgeo2020.pl"

path\_name %>%

paste0(file\_name) %>%

read.delim(

header=FALSE,

colClasses="character",

sep="|") -> red1

names(red1) <- vnames

red1 %>%

filter(SUMLEV == "750") -> red1

red1 %>%

head

```

### Read Missouri redistricting files

```{r read-mo}

file\_name <- "red/mogeo2020.pl"

path\_name %>%

paste(file\_name, sep="/") %>%

read.delim(

header=FALSE,

colClasses="character",

sep="|") -> red2

names(red2) <- vnames

red2 %>%

filter(SUMLEV == "750") -> red2

red2 %>%

head

```

### Store as an RData file

Placing the file **in** an RData format will speed up future use of this file.

```{r store}

clist <- c(

"20091",

"20103",

"20209",

"29037",

"29047",

"29095",

"29165")

red1 %>%

bind\_rows(red2) %>%

filter(BLOCK != "") %>%

filter(

paste0(STATE, COUNTY)

%in% clist) %>%

mutate(POP100=as.numeric(POP100)) %>%

mutate(HU100=as.numeric(HU100)) %>%

rename(people=POP100) %>%

rename(units=HU100) %>%

rename(land\_area=AREALAND) %>%

rename(water\_area=AREAWATR) %>%

rename(block\_lat=INTPTLAT) %>%

rename(block\_lon=INTPTLON) %>%

mutate(

st\_id=

str\_sub(GEOCODE, 1, 2)) %>%

mutate(

co\_id=

str\_sub(GEOCODE, 1, 5)) %>%

mutate(

tr\_id=

str\_sub(GEOCODE, 1, 11)) %>%

mutate(

bg\_id=

str\_sub(GEOCODE, 1, 12)) %>%

mutate(

bl\_id=

str\_sub(GEOCODE, 1, 15)) %>%

select(

tr\_id,

bg\_id,

bl\_id,

people,

units) -> red

glimpse(red)

```

```{r save}

save(red, file="../data/red.RData")

```

Store county information:

---

title: "Store county information"

author: "Satya Golla and Steve Simon"

date: "Created 2022-03-14"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

The important counties **for** this program are

Cass County, MO (29037)

Clay County, MO (29047)

Jackson County, MO (29095)

Johnson County, KS (20091)

Leavenworth County, KS (20103)

Miami County, KS (20121)

Wyandotte County, KS (20209)

```{r setup}

library(sf)

library(tidyverse)

```

```{r read-county}

path\_name <- "../Data"

file\_name <- "co/tl\_2021\_us\_county.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> all\_counties

head(all\_counties)

all\_counties %>%

filter(STATEFP==20) %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("ks-counties")

all\_counties %>%

filter(STATEFP==29) %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("mo-counties")

all\_counties %>%

filter(STATEFP==20|STATEFP==29) -> co

co %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("mo-ks-counties")

```

Refer to the data dictionary, county-data-dictionary.yaml **for** information about where to find this data and what all the variables represent.

This program will read **in** census county files and store them **in** an RData file **for** later use.

### Rename variables

For simplicity, we reduce the number of variables and keep only those needed **for** later analyses. The variables **in** the reduced dataset are

+ co\_id

+ co\_name

+ co\_area

+ geometry

### Store as an RData file

Placing the file **in** an RData format will speed up future use of this file.

```{r save}

save(co, file="../Data/co.RData")

```

Store puma information:

---

title: "Store puma information"

author: "Satya Golla and Steve Simon"

date: "Created 2022-03-14"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

```{r setup}

library(sf)

library(tidyverse)

```

### Public Use Microdata Areas (PUMA)

A PUMA is a region within a state that contains no less than 100,000 people. There are files **for** PUMAs **for** [Kansas][c\_puma\_ks] and [Missouri][c\_puma\_mo]. The various functions described **in** the US counties shapefiles section will be used here.

```{r read-puma-ks}

path\_name <- "../Data"

file\_name <- "puma/tl\_2021\_20\_puma10.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> all\_puma\_ks

head(all\_puma\_ks)

all\_puma\_ks %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("ks-puma")

```

variable **in** the file

+ Variables:

+ "STATEFP10"

+ "PUMACE10"

+ "GEOID10"

+ "NAMELSAD10"

+ "MTFCC10"

+ "FUNCSTAT10"

+ "ALAND10"

+ "AWATER10"

+ "INTPTLAT10"

+ "INTPTLON10"

+ "geometry"

```{r read-puma-mo}

path\_name <- "../data"

file\_name <- "puma/tl\_2021\_29\_puma10.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> all\_puma\_mo

head(all\_puma\_mo)

all\_puma\_mo %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("mo-puma") -> puma

```

variable **in** the file

+ Variables:

+ "STATEFP10"

+ "PUMACE10"

+ "GEOID10"

+ "NAMELSAD10"

+ "MTFCC10"

+ "FUNCSTAT10"

+ "ALAND10"

+ "AWATER10"

+ "INTPTLAT10"

+ "INTPTLON10"

+ "geometry"

```{r combine-pumas}

all\_puma\_ks %>%

bind\_rows(all\_puma\_mo) -> mo\_ks\_puma

mo\_ks\_puma %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("mo-ks-puma")

```

### Community districts

This shapefile, Community\_District, was provided by Neal Wilson. The various functions described **in** the US counties shapefiles section will be used here.

```{r read-Community-Districts}

file\_name <- "cd/Community\_District.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> Community\_Districts

head(Community\_Districts)

Community\_Districts %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("Community\_Districts")

Community\_Districts %>%

st\_transform(crs=st\_crs(mo\_ks\_puma)) -> cd

```

#### Intersection of pumas with community districts

Each community district will reside entirely inside a single county, and it is obvious to anyone familiar with the Kansas City metropolitan area which county is would be, so this is done mostly **for** the sake of continuity and completeness.

It is important that both the pumas and the community districts use a common coordinate reference system. It is unclear WHICH coordinate reference system is best, but this is controlled by the st\_crs function of the sf package.

The st\_intersects function will evaluate to TRUE **if** two shapefiles intersect. The st\_area function will give the area of a shapefile and the st\_intersection function will produce a new shapefile from the intersection of two shapefiles.

```{r intersect-puma-Community-Districts}

for(i in 1:2) {

mo\_ks\_puma %>%

st\_intersects( cd[i,], sparse=FALSE) -> intersect\_matrix

total\_area <- st\_area(mo\_ks\_puma[intersect\_matrix,]) / 1000000

intersect\_area <- st\_area(st\_intersection(mo\_ks\_puma[intersect\_matrix,],cd[i,])) / 1000000

print(intersect\_area/total\_area)

map\_position <- st\_centroid(mo\_ks\_puma[intersect\_matrix, ])

ggplot(data=mo\_ks\_puma[intersect\_matrix, ], aes()) +

geom\_sf(fill=NA, col="darkgreen") +

geom\_sf(data=cd[i,], fill=NA, col="darkred")+

ggtitle(paste(c(cd$NAME[i],mo\_ks\_puma$GEOID10[intersect\_matrix]),collapse = ", ")) -> map

print(map)

}

```

### Neighborhoods

This shapefile, Neighborhoods, was provided by Neal Wilson. The various functions described **in** the US counties shapefiles section will be used here.

```{r read-Neighborhoods, eval = FALSE}

file\_name <- "nbd/Neighborhoods.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> Neighborhoods

head(Neighborhoods)

Neighborhoods %>%

ggplot() +

geom\_sf(aes()) +

ggtitle("Neighborhoods")

Neighborhoods %>%

st\_transform(crs=st\_crs(mo\_ks\_puma)) -> nbd

```

```{r intersect-puma-Neighborhoods}

for(i in 1:2) {

mo\_ks\_puma %>%

st\_transform(crs=st\_crs(mo\_ks\_puma)) -> nbd

st\_intersects(nbd[i,], sparse=FALSE) -> intersect\_matrix

print(intersect\_matrix)

ggplot(data=nbd, aes()) +

geom\_sf(fill="lightgreen", col="darkgreen") +

geom\_sf(data=nbd[intersect\_matrix, ], fill="pink", col="darkred")

}

```

### Store as an RData file

Placing the file **in** an RData format will speed up future use of this file.

```{r save}

save(puma, file="../data/puma.RData")

```

Store community information:

---

title: "Store community information"

author: "Satya Golla and Steve Simon"

date: "Created 2022-03-11"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will read **in** shapefiles **for** community districts **in** the Kansas City metropolitan area, convert them into the simple features (sf) format, and store the results **for** later use.

### General information

An email from Neal Wilson dated 2022-03-03 and found **in** the doc folder outlines some of the details about the origins of this file. It was originally sent **in** an email dated 2021-10-04.

### Read community district shapefiles

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../data"

```

```{r read}

file\_name <- "cd/Community\_District.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> shape1

head(shape1)

shape1 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Align coordinate reference system

Use the same coordinate reference system that is found **in** the U.S. Census Bureau's shapefiles. The st\_zm function removes an unneeded z dimension.

```{r align}

load("../Data/bl.RData")

shape1 %>%

rename(cd\_id=CD\_ID) %>%

rename(cd\_name=CD\_NAME) %>%

mutate(

cd\_area=

as.numeric(st\_area(shape1))) %>%

select(

cd\_id,

cd\_name,

cd\_area,

geometry) %>%

st\_transform(crs=st\_crs(bl)) %>%

st\_zm -> cd

glimpse(cd)

```

### Store as an RData file

Placing the file in an RData format will speed up future use of this file.

```{r store}

save(cd, file="../data/cd.RData")

```

Store neighborhood information:

---

title: "Store neighborhood information"

author: "Satya Golla and Steve Simon"

date: "Created 2022-03-11"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will read **in** shapefiles **for** neighborhoods **in** the Kansas City metropolitan area, convert them into the simple features (sf) format, and store the results **for** later use.

### General information

An email from Neal Wilson, dated 2022-03-03 and found **in** the doc folder, outlines some of the details about the origins of this file. It was originally sent **in** an email dated 2021-10-04.

### Read neighboorhood shapefiles

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../data"

```

```{r read}

file\_name <- "nbd/Neighborhoods.shp"

path\_name %>%

paste(file\_name, sep="/") %>%

st\_read(stringsAsFactors=FALSE) -> shape1

head(shape1)

shape1 %>%

ggplot() +

geom\_sf(aes()) +

ggtitle(file\_name)

```

### Align coordinate reference system

Use the same coordinate reference system that is found **in** the U.S. Census Bureau's shapefiles. The st\_zm function removes an unneeded z dimension.

```{r align}

load("../Data/bl.RData")

shape1 %>%

mutate(label=AreaName) %>%

st\_transform(crs=st\_crs(bl)) %>%

st\_zm -> nbd

```

### Store as an RData file

Placing the file in an RData format will speed up future use of this file.

```{r store-counties}

save(nbd, file="../Data/nbd.RData")

```

Calculate community districts-weights:

---

title: "Calculate community districts weights"

author: "Satya Golla, Mounika Jakkidi, and Steve Simon"

date: "Created 2022-03-11"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will calculate which census blocks intersect with which community districts. It relies on information stored **in**

wd

+ bg.RData

+ bl.RData

+ cd.RData

+ cd-intersections.RData

+ red.RData

### General information

For information about various files, refer to the files that stored the relevant data (e.g., store\_block\_information.Rmd).

It helps to know the FIPS (Federal Information Processing System) codes **for** Kansas (20) and Missouri (29).

The important counties **for** this program are

Cass County, MO (29037)

Clay County, MO (29047)

Jackson County, MO (29095)

Johnson County, KS (20091)

Leavenworth County, KS (20103)

Wyandotte County, KS (20209)

### Setup

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../data/"

```

### Load key files

Block groups

```{r load-bg}

load(paste0(path\_name, "bg", ".RData"))

glimpse(bg)

```

Blocks

```{r load-bl}

load(paste0(path\_name, "bl", ".RData"))

glimpse(bl)

```

Community districts

```{r load-cd}

load(paste0(path\_name, "cd", ".RData"))

glimpse(cd)

```

Community district intersections

```{r load-cd-intersections}

load(paste0(path\_name, "cd-intersections", ".RData"))

glimpse(bg\_cd\_intersection)

glimpse(bl\_cd\_intersection)

```

```{r load-redistricting}

load(paste0(path\_name, "red", ".RData"))

glimpse(red)

```

### Counts

```{r display-nice-counts}

# This code will compute the count of a vector

# and display it with commas if it is greater

# than 999.

nice\_count <- function(x) {

x %>%

unique %>%

length %>%

format(big.mark=",")

}

```

There are 7 counties **in** the Kansas City metropolitan area.

There are `r nice\_count(cd$cd\_id)` community districts **in** the 7 counties.

There are

`r nice\_count(bg$bg\_id)`

block groups **in** the 7 counties.

`r nice\_count(bg\_cd\_intersection$bg\_id)`

of these intersect partially or completely with one or more community districts.

There are

`r nice\_count(bl$bl\_id)`

blocks **in** the 7 counties.

`r nice\_count(bl\_cd\_intersection$bl\_id)`

of these intersect partially or completely with one or more community districts.

### Get block group list

Identify the list of all block groups that partially or completely intersect with any community district.

```{r intersect-bg}

bg\_cd\_intersection %>%

pull(bg\_id) %>%

unique -> bg\_list

glimpse(bg\_list)

```

### Get block list

Now get a list of all census blocks that lie inside any of these block groups. Merge with red to get population and housing unit counts.

```{r pull-blocks}

bl %>%

pull(bl\_id) %>%

str\_sub(1, 12) %>%

bind\_cols(bl$bl\_id) %>%

set\_names(c("bg\_id", "bl\_id")) %>%

filter(bg\_id %in% bg\_list) %>%

inner\_join(red,

by=c("bg\_id", "bl\_id")) -> bl\_list

glimpse(bl\_list)

```

### Calculate counts and areas of intersections

```{r calculate-proportions}

bl\_cd\_intersection %>%

full\_join(bl\_list,

by=c("bg\_id", "bl\_id")) %>%

replace\_na(list(bl\_prop\_in=0)) %>%

replace\_na(list(cd\_id=0)) %>%

mutate(

people\_in=

round(people\*bl\_prop\_in)) %>%

mutate(

units\_in=

round(units\*bl\_prop\_in)) %>%

select(

bg\_id,

bl\_id,

cd\_name,

cd\_id,

bl\_prop\_in,

people,

people\_in,

units,

units\_in) -> bl\_counts

glimpse(bl\_counts)

```

### Calculate counts for block groups

```{r count-bl}

red %>%

group\_by(bg\_id) %>%

summarize(

people=sum(people),

units=sum(units)) -> count\_total

glimpse(count\_total)

bl\_counts %>%

group\_by(bg\_id, cd\_id) %>%

summarize(

people\_in=sum(people\_in),

units\_in=sum(units\_in)) %>%

filter(cd\_id > 0) -> count\_in

glimpse(count\_in)

count\_in %>%

inner\_join(count\_total, by="bg\_id") %>%

filter(cd\_id > 0) -> bg\_counts

glimpse(bg\_counts)

```

### Save the information in an RData file.

```{r save}

save(

bg\_counts,

bg\_list,

bl\_counts,

bl\_list,

file="../Data/cd-weights.RData")

```

Calculate intersection of blocks and community districts:

---

title: "Calculate intersection of blocks and community districts"

author: "Satya Golla ,Mounika Jakkidi, and Steve Simon"

date: "Created 2022-03-11"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will calculate which census blocks intersect with which community districts. It relies on information stored **in**

+ bg.RData

+ bl.RData

+ cd.RData

### General information

For information about various files, refer to the files that stored the relevant data (e.g., store\_block\_information.Rmd).

It helps to know the FIPS (Federal Information Processing System) codes **for** Kansas (20) and Missouri (29).

The important counties **for** this program are

Cass County, MO (29037)

Clay County, MO (29047)

Jackson County, MO (29095)

Johnson County, KS (20091)

Leavenworth County, KS (20103)

Wyandotte County, KS (20209)

### Load relevant files

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../data/"

load(paste0(path\_name, "bg", ".RData"))

load(paste0(path\_name, "bl", ".RData"))

load(paste0(path\_name, "cd", ".RData"))

```

### Intersect blocks and community districts

Identify the list of all blocks that partially or completely intersect with any community district. Calculate the block area and add it back to the data frame.

```{r intersect-bl}

bl %>%

st\_intersection(cd) -> bl\_cd\_intersection

bl\_cd\_intersection %>%

st\_area %>%

as.numeric %>%

tibble %>%

set\_names("in\_area") %>%

bind\_cols(bl\_cd\_intersection) %>%

mutate(

bg\_id=str\_sub(bl\_id, 1, 12)) %>%

mutate(

bl\_prop\_in=in\_area/bl\_area) %>%

mutate(

bl\_prop\_in=round(bl\_prop\_in, 3)) %>%

select(

bg\_id,

bl\_id,

cd\_id,

cd\_name,

bl\_prop\_in) -> bl\_cd\_intersection

```

### Intersect block groups and community districts

Identify the list of all block groups that partially or completely intersect with any community district.

```{r intersect-bg}

bg %>%

st\_intersection(cd) -> bg\_cd\_intersection

bg\_cd\_intersection %>%

st\_area %>%

as.numeric %>%

tibble %>%

set\_names("in\_area") %>%

bind\_cols(bg\_cd\_intersection) %>%

mutate(bg\_prop\_in=in\_area/bg\_area) %>%

mutate(

bg\_prop\_in=round(bg\_prop\_in, 3)) %>%

select(

bg\_id,

cd\_name,

cd\_id,

bg\_prop\_in) -> bg\_cd\_intersection

```

### Save the information in an RData file.

```{r save}

save(

bl\_cd\_intersection,

bg\_cd\_intersection,

file="../Data/cd-intersections.RData")

```

Calculate neighborhood -weights:

---

title: "Calculate neighborhood weights"

author: "Satya Golla, Mounika Jakkkidi and Steve Simon"

date: "Created 2022-03-11"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will calculate which census blocks intersect with which neighborhoods. It relies on information stored **in**

+ bg.RData

+ bl.RData

+ nbd.RData

+ nbd-intersections.RData

+ red.RData

### General information

For information about various files, refer to the files that stored the relevant data (e.g., store\_block\_information.Rmd).

It helps to know the FIPS (Federal Information Processing System) codes **for** Kansas (20) and Missouri (29).

The important counties **for** this program are

Cass County, MO (29037)

Clay County, MO (29047)

Jackson County, MO (29095)

Johnson County, KS (20091)

Leavenworth County, KS (20103)

Wyandotte County, KS (20209)

### Setup

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../Data/"

```

### Load key files

Block groups

```{r load-bg}

load(paste0(path\_name, "bg", ".RData"))

glimpse(bg)

```

Blocks

```{r load-bl}

load(paste0(path\_name, "bl", ".RData"))

glimpse(bl)

```

neighborhoods

```{r load-nbd}

load(paste0(path\_name, "nbd", ".RData"))

glimpse(nbd)

```

neighborhoods intersections

```{r load-nbd-intersections}

load(paste0(path\_name, "nbd-intersections", ".RData"))

glimpse(bg\_nbd\_intersection)

glimpse(bl\_nbd\_intersection)

```

```{r load-redistricting}

load(paste0(path\_name, "red", ".RData"))

glimpse(red)

```

### Counts

```{r display-nice-counts}

# This code will compute the count of a vector

# and display it with commas if it is greater

# than 999.

nice\_count <- function(x) {

x %>%

unique %>%

length %>%

format(big.mark=",")

}

```

There are 7 counties **in** the Kansas City metropolitan area.

There are `r nice\_count(nbd$NID)` neighborhoods **in** the 7 counties.

There are

`r nice\_count(bg$bg\_id)`

block groups **in** the 7 counties.

`r nice\_count(bg\_nbd\_intersection$bg\_id)`

of these intersect partially or completely with one or more neighborhoods.

There are

`r nice\_count(bl$bl\_id)`

blocks **in** the 7 counties.

`r nice\_count(bl\_nbd\_intersection$bl\_id)`

of these intersect partially or completely with one or more neighborhoods.

### Get block group list

Identify the list of all block groups that partially or completely intersect with any neighborhoods.

```{r intersect-bg}

bg\_nbd\_intersection %>%

pull(bg\_id) %>%

unique -> bg\_list

glimpse(bg\_list)

```

### Get block list

Now get a list of all census blocks that lie inside any of these block groups. Merge with red to get population and housing unit counts.

```{r pull-blocks}

bl %>%

pull(bl\_id) %>%

str\_sub(1, 12) %>%

bind\_cols(bl$bl\_id) %>%

set\_names(c("bg\_id", "bl\_id")) %>%

filter(bg\_id %in% bg\_list) %>%

inner\_join(red,

by=c("bg\_id", "bl\_id")) -> bl\_list

glimpse(bl\_list)

```

### Calculate counts and areas of intersections

```{r calculate-proportions}

bl\_nbd\_intersection %>%

full\_join(bl\_list,

by=c("bg\_id", "bl\_id")) %>%

replace\_na(list(bl\_prop\_in=0)) %>%

replace\_na(list(NID=0)) %>%

mutate(

people\_in=

round(people\*bl\_prop\_in)) %>%

mutate(

units\_in=

round(units\*bl\_prop\_in)) %>%

select(

bg\_id,

bl\_id,

NID,

AreaName,

bl\_prop\_in,

people,

people\_in,

units,

units\_in) -> bl\_counts

glimpse(bl\_counts)

```

### Calculate counts for block groups

```{r count-bl}

red %>%

group\_by(bg\_id) %>%

summarize(

people=sum(people),

units=sum(units)) -> count\_total

glimpse(count\_total)

bl\_counts %>%

group\_by(bg\_id, NID) %>%

summarize(

people\_in=sum(people\_in),

units\_in=sum(units\_in)) %>%

filter(NID > 0) -> count\_in

glimpse(count\_in)

count\_in %>%

inner\_join(count\_total, by="bg\_id") %>%

filter(NID > 0) -> bg\_counts

glimpse(bg\_counts)

```

###sum across blockgroups to get tracts counts

```{r}

bg\_counts %>%

mutate(tr\_id=str\_sub(bg\_id,1,11)) %>%

group\_by(NID,tr\_id) %>%

summarize(

people=sum(people),

people\_in=sum(people\_in),

units=sum(units),

units\_in=sum(units\_in)) -> tr\_counts

glimpse(tr\_counts)

```

### Save the information in an RData file.

```{r save}

save(

bg\_counts,

bl\_counts,

tr\_counts,

file="../Data/nbd-weights.RData")

```

Calculate neighborhood intersection:

---

title: "Calculate intersection of blocks and neighorhoods"

author: "Satya Golla, Mounika Jakkidi, and Steve Simon"

date: "Created 2022-03-11"

output: html\_document

knit: (function(inputFile, encoding) {

rmarkdown::render(inputFile, encoding = encoding,

output\_dir = "../results", output\_format = "all") })

---

This program will calculate which census blocks intersect with neighorhoods. It relies on information stored **in**

+ bg.RData

+ bl.RData

+ nbd.RData

### General information

For information about various files, refer to the files that stored the relevant data (e.g., store\_block\_information.Rmd).

It helps to know the FIPS (Federal Information Processing System) codes **for** Kansas (20) and Missouri (29).

The important counties **for** this program are

Cass County, MO (29037)

Clay County, MO (29047)

Jackson County, MO (29095)

Johnson County, KS (20091)

Leavenworth County, KS (20103)

Wyandotte County, KS (20209)

### Load relevant files

```{r setup}

library(sf)

library(tidyverse)

path\_name <- "../Data/"

load(paste0(path\_name, "bg", ".RData"))

load(paste0(path\_name, "bl", ".RData"))

load(paste0(path\_name, "nbd", ".RData"))

```

### Intersect blocks and neighorhoods

Identify the list of all blocks that partially or completely intersect with neighborhoods. Calculate the block area and add it back to the data frame.

```{r intersect-bl}

Sys.time()

bl %>%

st\_intersection(nbd) -> bl\_nbd\_intersection

glimpse(bl\_nbd\_intersection)

bl\_nbd\_intersection %>%

st\_area %>%

as.numeric %>%

tibble %>%

set\_names("in\_area") %>%

bind\_cols(bl\_nbd\_intersection) %>%

mutate(

bg\_id=str\_sub(bl\_id, 1, 12)) %>%

mutate(

bl\_prop\_in=in\_area/bl\_area) %>%

mutate(

bl\_prop\_in=round(bl\_prop\_in, 3)) %>%

select(

bg\_id,

bl\_id,

AreaName,

NID,

bl\_prop\_in) -> bl\_nbd\_intersection

Sys.time()

glimpse(bl\_nbd\_intersection)

```

### Intersect block groups and neighorhoods

Identify the list of all block groups that partially or completely intersect with neighborhoods.

```{r intersect-bg}

Sys.time()

bg %>%

st\_intersection(nbd) -> bg\_nbd\_intersection

glimpse(bg\_nbd\_intersection)

bg\_nbd\_intersection %>%

st\_area %>%

as.numeric %>%

tibble %>%

set\_names("in\_area") %>%

bind\_cols(bg\_nbd\_intersection) %>%

mutate(bg\_prop\_in=in\_area/bg\_area) %>%

mutate(

bg\_prop\_in=round(bg\_prop\_in, 3)) %>%

select(

bg\_id,

NID,

AreaName,

bg\_prop\_in) -> bg\_nbd\_intersection

Sys.time()

glimpse(bg\_nbd\_intersection)

```

### Save the information in an RData file.

```{r save}

save(

bl\_nbd\_intersection,

bg\_nbd\_intersection,

file="../Data/nbd-intersections.RData")

```

**REFERENCES:**

American Community survey 2017-2021 5-year - census.gov. (n.d.). Retrieved April 1, 2023, from <https://www2.census.gov/programs-surveys/acs/tech_docs/pums/2017_2021ACS_PUMS_User_Guide.pdf>

About the 2023 118 congressional district Tiger/Line shapefiles. (n.d.). Retrieved April 1, 2023, from.`<https://www2.census.gov/geo/pdfs/maps/data/data/tiger/tgrshp_rd18/TGRSHPRD18_TechDoc_Ch3.pdf>

Bivand, R. (2022). R packages for Analyzing Spatial Data: A comparative case study with Areal Data. *Geographical Analysis*, *54*(3), 488–518. <https://doi.org/10.1111/gean.12319>

Bureau, U. S. C. (2023, March 3). *Data*. Census Website. Retrieved March 31, 2023, from https://www.census.gov/data/

Bureau, U. S. C. (2021, October 8). *Understanding Geographic Identifiers (geoids)*. Census.gov. Retrieved March 31, 2023, from https://www.census.gov/programs-surveys/geography/guidance/geo-identifiers.html

Bureau, U. S. C. (n.d.). Explore census data. Retrieved March 31, 2023, from <https://data.census.gov/webpages?g=050XX00US10005>

Bureau, U. S. C. (2023, March 16). *American Community Survey (ACS)*. Census.gov. Retrieved March 31, 2023, from https://www.census.gov/programs-surveys/acs/

Bureau, U. S. C. (2022, March 16). *American National Standards Institute (ANSI) and Federal Information Processing Series (FIPS) codes*. Census.gov. Retrieved March 31, 2023, from https://www.census.gov/library/reference/code-lists/ansi.html

Campbell, M. (2019). Essential R packages: Tidyverse. *Learn RStudio IDE*, 63–72. https://doi.org/10.1007/978-1-4842-4511-8\_6

Esri Shapefile Reader/writer. (n.d.). Retrieved March 31, 2023, from https://docs.safe.com/fme/html/FME\_Desktop\_Documentation/FME\_ReadersWriters/shapefile/shapefile.htm

Her, Y. G., & Yu, Z. (2021). Mapping the US census data using the Tiger/Line Shapefiles. *EDIS*, *2021*(3). <https://doi.org/10.32473/edis-ae557-2021>

Ihaka, R., & Gentleman, R. (1996). R: A language for data analysis and Graphics. *Journal of Computational and Graphical Statistics*, *5*(3), 299–314. https://doi.org/10.1080/10618600.1996.10474713

Kedron, P., Frazier, A. E., Trgovac, A. B., Nelson, T., & Fotheringham, A. S. (2019). Reproducibility and replicability in geographical analysis. *Geographical Analysis*, *53*(1), 135–147. <https://doi.org/10.1111/gean.12221>

Machlis, S. (2019, December 19). *How to do spatial analysis in R with SF*. InfoWorld. Retrieved March 31, 2023, from <https://www.infoworld.com/article/3505897/how-to-do-spatial-analysis-in-r-with-sf.html>

Prewitt, K. (2010). The U.S. Decennial Census: Politics and Political Science. *Annual Review of Political Science*, *13*(1), 237–254. <https://doi.org/10.1146/annurev.polisci.031108.095600>

Pebesma, E. (2018). Simple features for r: Standardized support for Spatial Vector Data. *The R Journal*, *10*(1), 439. https://doi.org/10.32614/rj-2018-009

Ratcliffe, M. R. (2019). Providing context and meaning to statistical data: The Work of Census Bureau Geographers. *The Geography Teacher*, *16*(3), 107–111. https://doi.org/10.1080/19338341.2019.1619609

Rossiter, K. (2021, October 8). "understanding geographic relationships: Counties, places, tracts and more". The United States Census Bureau. Retrieved March 31, 2023, from <https://www.census.gov/newsroom/blogs/random-samplings/2014/07/understanding-geographic-relationships-counties-places-tracts-and-more.html>

Saalfeld, A. (n.d.). Sorting spatial data for sampling and other geographic applications - geoinformatica. SpringerLink. Retrieved March 31, 2023, from https://link.springer.com/article/10.1023/A:1009741021887#citeas

The most trusted IDE for Open Source Data Science. Posit. (2022, November 23). Retrieved March 31, 2023, from https://posit.co/products/open-source/rstudio/

United States Department of Housing and Urban Development. (2012). *The Encyclopedia of Housing*. https://doi.org/10.4135/9781452218380.n256

*What is R?* R. (n.d.)(2012). Retrieved March 31, 2023, from https://www.r-project.org/about.html

Walker, K. E. (2016, August 11). Tigris: An R package to access and work with geographic data from the US Census Bureau. Home. Retrieved March 31, 2023, from <https://repository.tcu.edu/handle/116099117/43087>