There’s a Package for That?

R Packages that Support Analysis of Common Federal Datasets

Kelsey Farson Gray

Senior Data Scientist, Insight Policy Research

insight POLICY RESEARCH

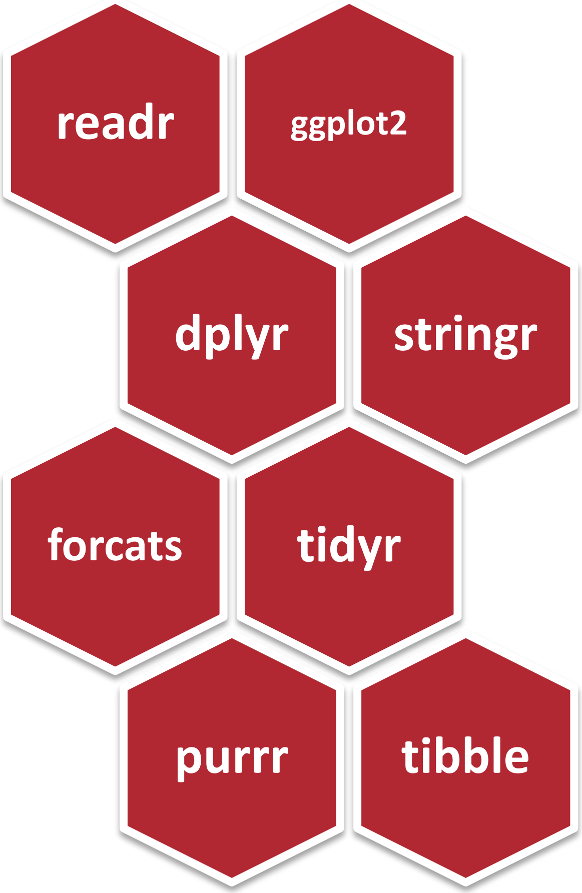
- **What is an R package?**
  - Collection of functions structured to support a process
  - Well-documented
  - Available for download

- **Accessing packages**

```
install.packages(“<package>”)
library(<package>)
```

- **Popular packages**

Tidyverse



www.tidyverse.org

POP QUIZ:

How many R packages are there?

130

1,300

13,000

13,200

How can **R packages** support analyses of common federal data sets?

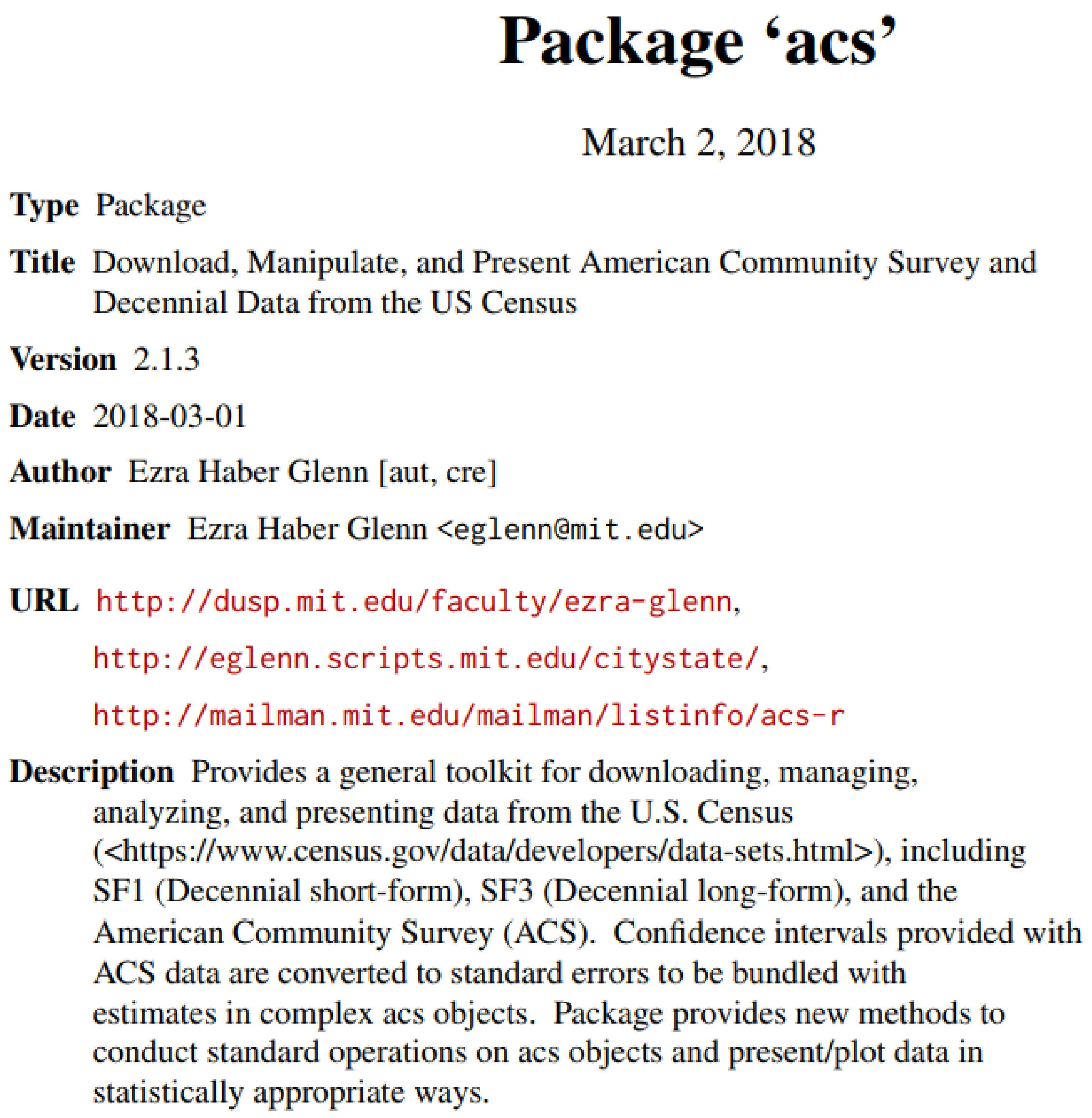
Today's examples

- Leverage **Application Program Interfaces (APIs)** to access data
  - Allows communication between software, **not** the Graphical User Interface (GUI)
  - **Example**: OpenTable
- Expedite routine analyses
- Enhance reproducibility / documentation

How do I get started?

- **Know what you're looking for** (e.g., data set, data year, table and variable names)
- **Search available packages**
  - How do I find new packages? (CRAN, GitHub)
  - How do I interpret R packages?
- **Explore available APIs** for commonly used data
  - Request API keys, if needed

Examples



Package ‘acs’

March 2, 2018

Type Package

Title Download, Manipulate, and Present American Community Survey and Decennial Data from the US Census

Version 2.1.3

Date 2018-03-01

Author Ezra Haber Glenn [aut, cre]

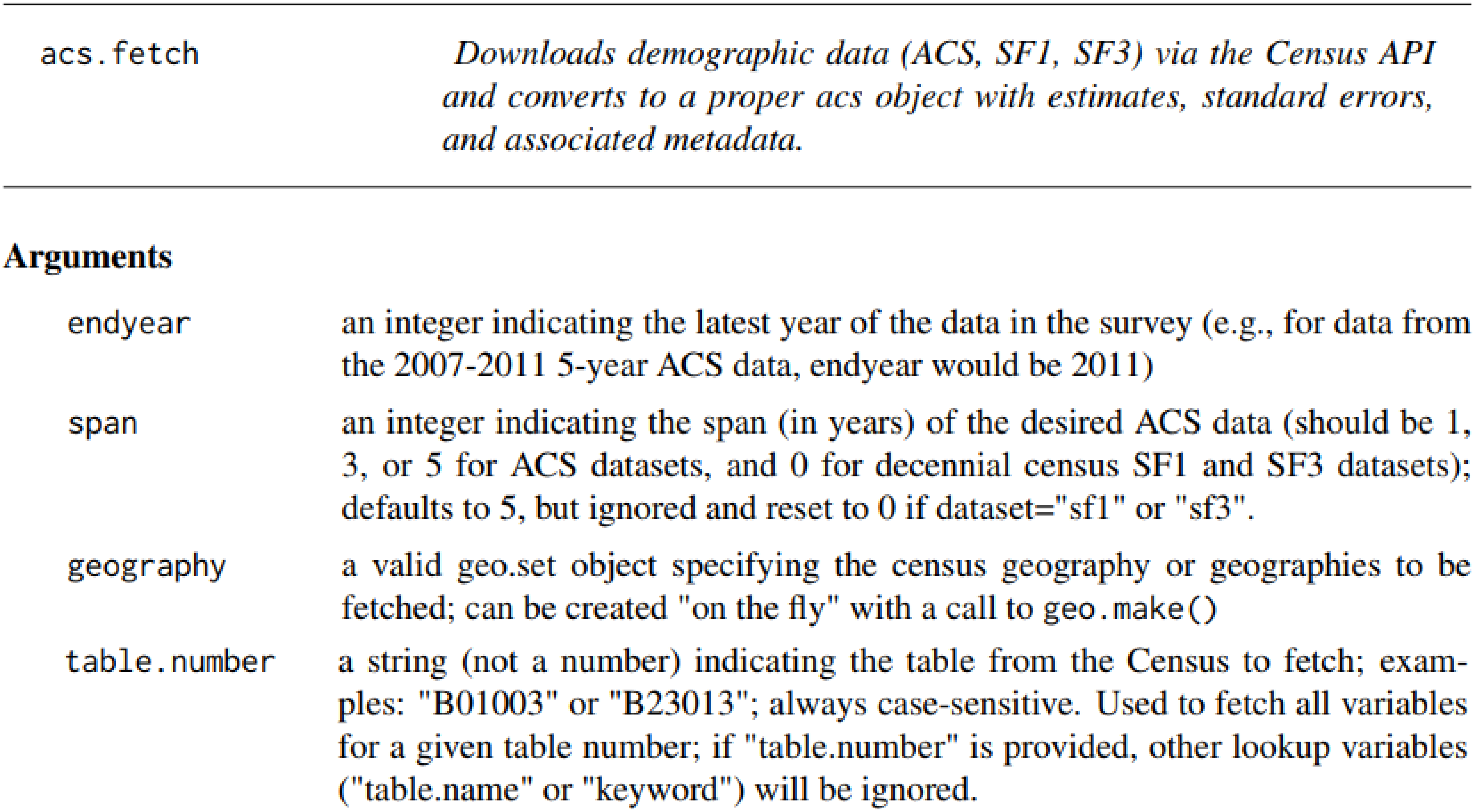
Maintainer Ezra Haber Glenn <eglenn@mit.edu>

URL http://dusp.mit.edu/faculty/ezra-glenn,

http://eglenn.scripts.mit.edu/citystate/,

http://mailman.mit.edu/mailman/listinfo/acs-r

Description Provides a general toolkit for downloading, managing, analyzing, and presenting data from the U.S. Census (<https://www.census.gov/data/developers/data-sets.html>), including SF1 (Decennial short-form), SF3 (Decennial long-form), and the American Community Survey (ACS). Confidence intervals provided with ACS data are converted to standard errors to be bundled with estimates in complex acs objects. Package provides new methods to conduct standard operations on acs objects and present/plot data in statistically appropriate ways.



`acs.fetch` *Downloads demographic data (ACS, SF1, SF3) via the Census API and converts to a proper acs object with estimates, standard errors, and associated metadata.*

Arguments

`endyear` an integer indicating the latest year of the data in the survey (e.g., for data from the 2007-2011 5-year ACS data, endyear would be 2011)

`span` an integer indicating the span (in years) of the desired ACS data (should be 1, 3, or 5 for ACS datasets, and 0 for decennial census SF1 and SF3 datasets); defaults to 5, but ignored and reset to 0 if dataset="sf1" or "sf3".

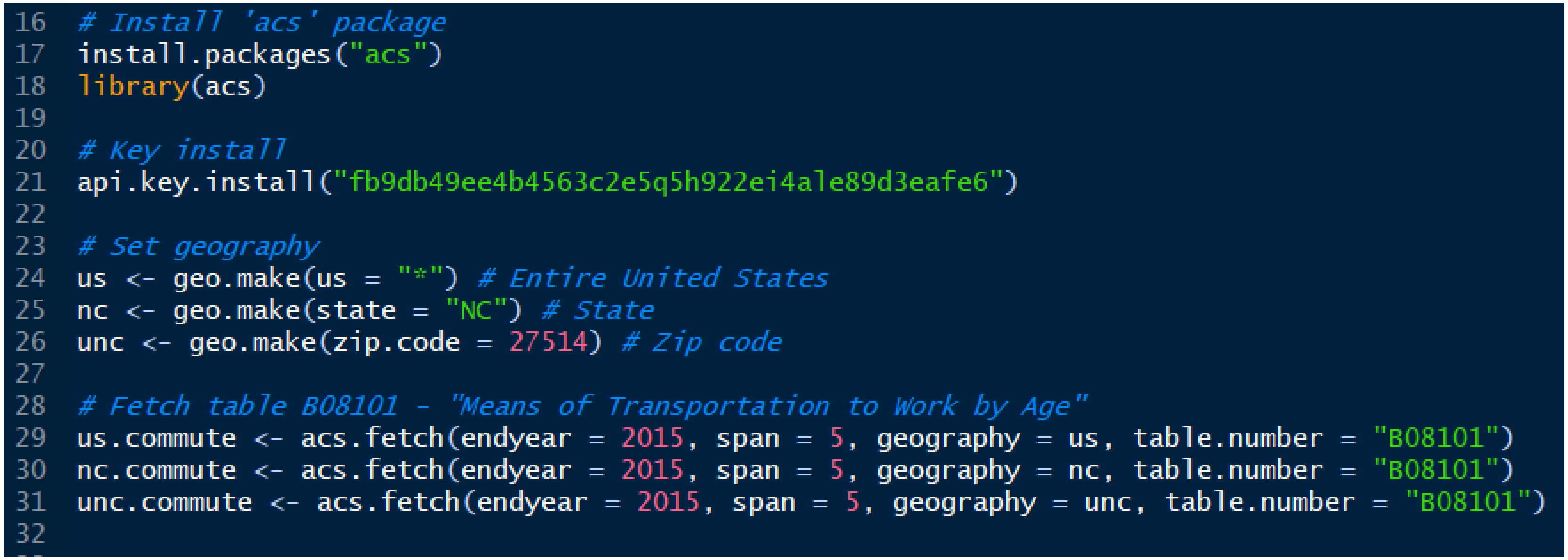
`geography` a valid geo.set object specifying the census geography or geographies to be fetched; can be created "on the fly" with a call to `geo.make()`

`table.number` a string (not a number) indicating the table from the Census to fetch; examples: "B01003" or "B23013"; always case-sensitive. Used to fetch all variables for a given table number; if "table.number" is provided, other lookup variables ("table.name" or "keyword") will be ignored.

NOTE: Additional arguments are accepted

How does mode of transportation to work vary within the United States?

Example: Code for fetching data through ‘acs’ package



```r
# Install 'acs' package
install.packages("acs")
library(acs)

# Key install
api.key.install("fb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Set geography
us <- geo.make(us = "*") # Entire United States
nc <- geo.make(state = "NC") # State
unc <- geo.make(zip.code = 27514) # Zip code

# Fetch table B08101 - "Means of Transportation to Work by Age"
us.commute <- acs.fetch(endyear = 2015, span = 5, geography = us, table.number = "B08101")
nc.commute <- acs.fetch(endyear = 2015, span = 5, geography = nc, table.number = "B08101")
unc.commute <- acs.fetch(endyear = 2015, span = 5, geography = unc, table.number = "B08101")
```

Let’s break it down…

```
# Install 'acs' package
install.packages("acs")
library(acs)
```

```
# Key install (request from Census)
api.key.install("fb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Set geography
us <- geo.make(us = "*") # Entire United States
nc <- geo.make(state = "NC") # State
unc <- geo.make(zip.code = 27514) # Zip code

# Fetch table B08101 - "Means of Transportation to Work by Age”
us.commute <- acs.fetch(endyear = 2015,
                        span = 5,
                        geography = us,
                        table.number = "B08101")
```

Let’s break it down…

```
# Install 'acs' package
install.packages("acs")
library(acs)

# Key install (request from Census)
api.key.install("fb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Set geography
us <- geo.make(us = "*") # Entire United States
nc <- geo.make(state = "NC") # State
unc <- geo.make(zip.code = 27514) # Zip code

# Fetch table B08101 - "Means of Transportation to Work by Age”
us.commute <- acs.fetch(endyear = 2015,
                        span = 5,
                        geography = us,
                        table.number = "B08101")
```

Let’s break it down…

```
# Install 'acs' package
install.packages("acs")
library(acs)

# Key install (request from Census)
api.key.install("fb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Set geography
us <- geo.make(us = "*") # Entire United States
nc <- geo.make(state = "NC") # State
unc <- geo.make(zip.code = 27514) # Zip code
```

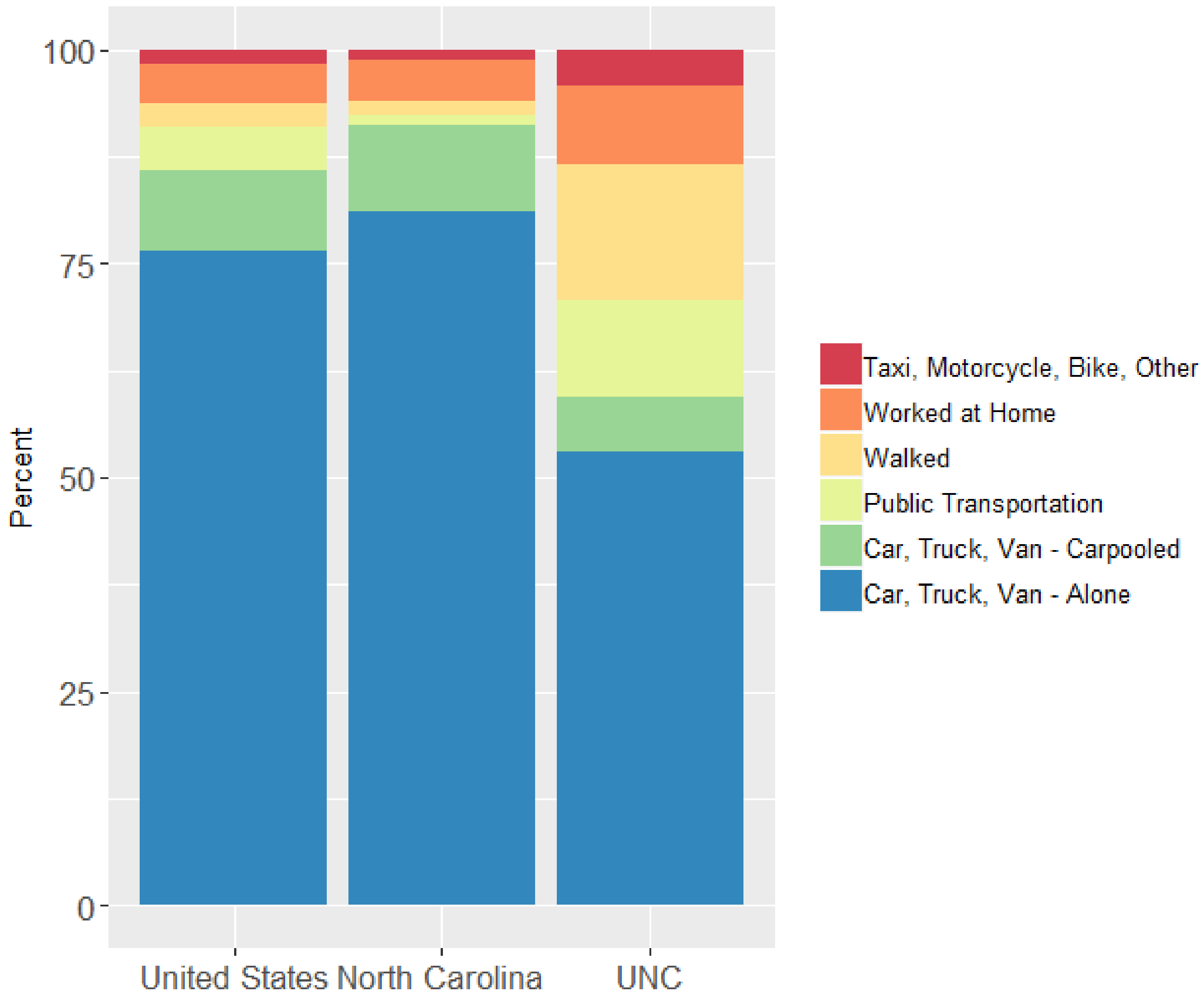
```
# Fetch table B08101 - "Means of Transportation to Work by Age”
us.commute <- acs.fetch(endyear = 2015,
                        span = 5,
                        geography = us,
                        table.number = "B08101")
```

Let’s break it down…

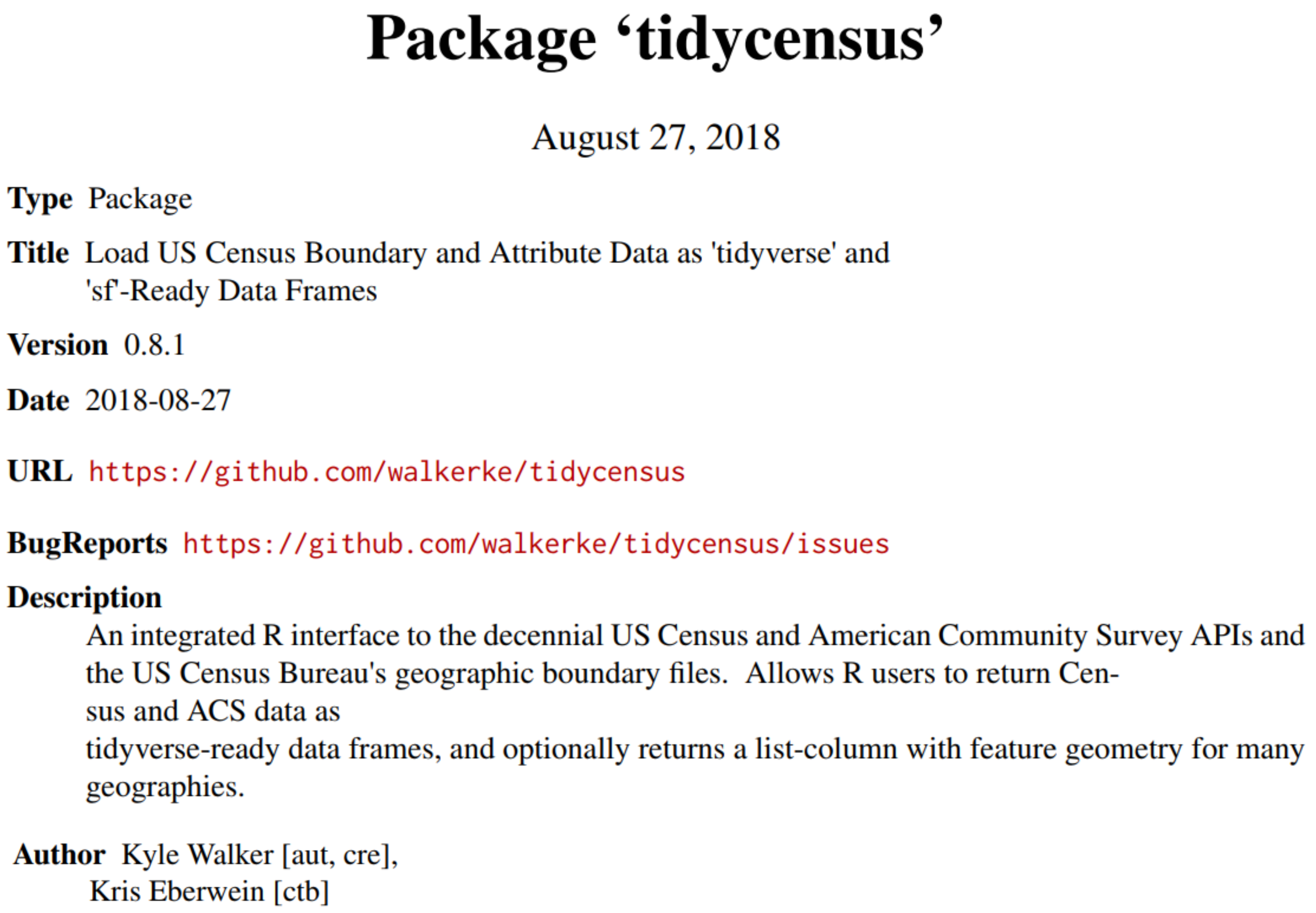
```
# Install 'acs' package
install.packages("acs")
library(acs)

# Key install (request from Census)
api.key.install("fb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Set geography
us <- geo.make(us = "*") # Entire United States
nc <- geo.make(state = "NC") # State
unc <- geo.make(zip.code = 27514) # Zip code
```

```
# Fetch table B08101 - "Means of Transportation to Work by Age”
us.commute <- acs.fetch(endyear = 2015,
                        span = 5,
                        geography = us,
                        table.number = "B08101")
```

Figure 1. Means of Transportation to Work by Location



Source: 2011-2015 American Community Survey 5-Year Estimates



Package ‘tidycensus’

August 27, 2018

Type Package

Title Load US Census Boundary and Attribute Data as 'tidyverse' and
'sf'-Ready Data Frames

Version 0.8.1

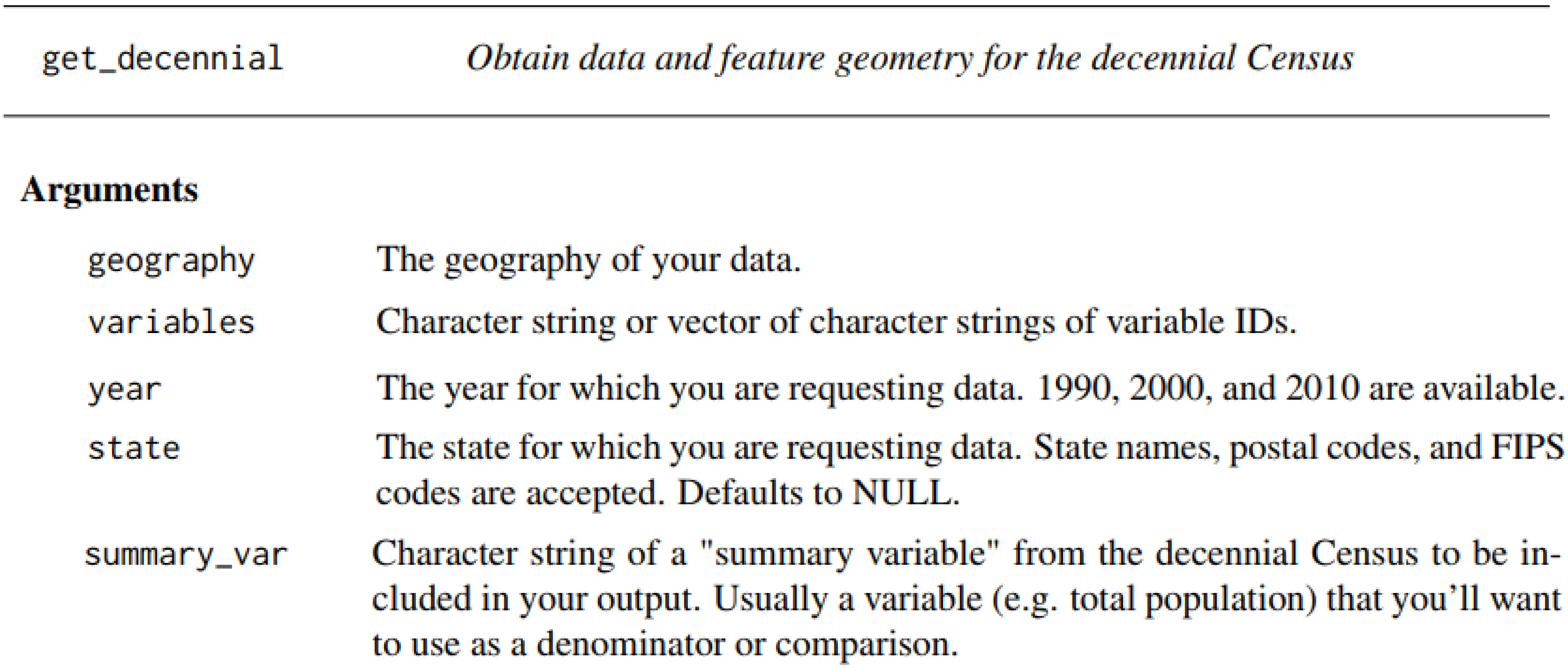
Date 2018-08-27

URL https://github.com/walkerke/tidycensus

BugReports https://github.com/walkerke/tidycensus/issues

Description
An integrated R interface to the decennial US Census and American Community Survey APIs and the US Census Bureau's geographic boundary files. Allows R users to return Census and ACS data as
tidyverse-ready data frames, and optionally returns a list-column with feature geometry for many geographies.

Author Kyle Walker [aut, cre],
Kris Eberwein [ctb]



get_decennial *Obtain data and feature geometry for the decennial Census*

Arguments

geography — The geography of your data.

variables — Character string or vector of character strings of variable IDs.

year — The year for which you are requesting data. 1990, 2000, and 2010 are available.

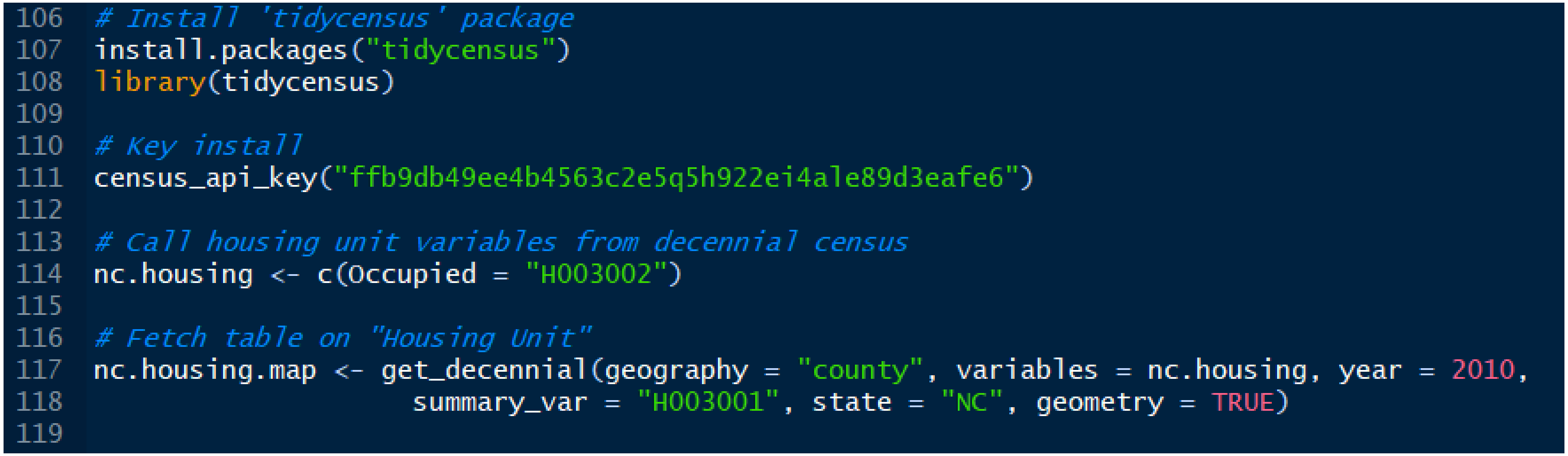
state — The state for which you are requesting data. State names, postal codes, and FIPS codes are accepted. Defaults to NULL.

summary_var — Character string of a "summary variable" from the decennial Census to be included in your output. Usually a variable (e.g. total population) that you'll want to use as a denominator or comparison.

NOTE: Additional arguments are accepted

How many housing units are occupied by county?

Example: Code for fetching data through ‘tidycensus’ package



```r
# Install 'tidycensus' package
install.packages("tidycensus")
library(tidycensus)

# Key install
census_api_key("ffb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Call housing unit variables from decennial census
nc.housing <- c(Occupied = "H003002")

# Fetch table on "Housing Unit"
nc.housing.map <- get_decennial(geography = "county", variables = nc.housing, year = 2010,
                    summary_var = "H003001", state = "NC", geometry = TRUE)

```

Let’s break it down…

```
# Install 'tidycensus' package
install.packages("tidycensus")
library(tidycensus)
```

```
# Key install
census_api_key("ffb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Call housing unit variables from decennial census
housing <- c(Occupied = "H003002",
             Vacant   = "H003003")

# Fetch table on "Housing Unit"
nc.housing.map <- get_decennial(geography = "county",
                                variables = housing,
                                year = 2010,
                                summary_var = "H003001",
                                state = "NC",
                                geometry = TRUE)
```

Let’s break it down…

```
# Install 'tidycensus' package
install.packages("tidycensus")
library(tidycensus)

# Key install
census_api_key("ffb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Call housing unit variables from decennial census
housing <- c(Occupied = "H003002",
             Vacant   = "H003003")

# Fetch table on "Housing Unit"
nc.housing.map <- get_decennial(geography = "county",
                                variables = housing,
                                year = 2010,
                                summary_var = "H003001",
                                state = "NC",
                                geometry = TRUE)
```

Let’s break it down…

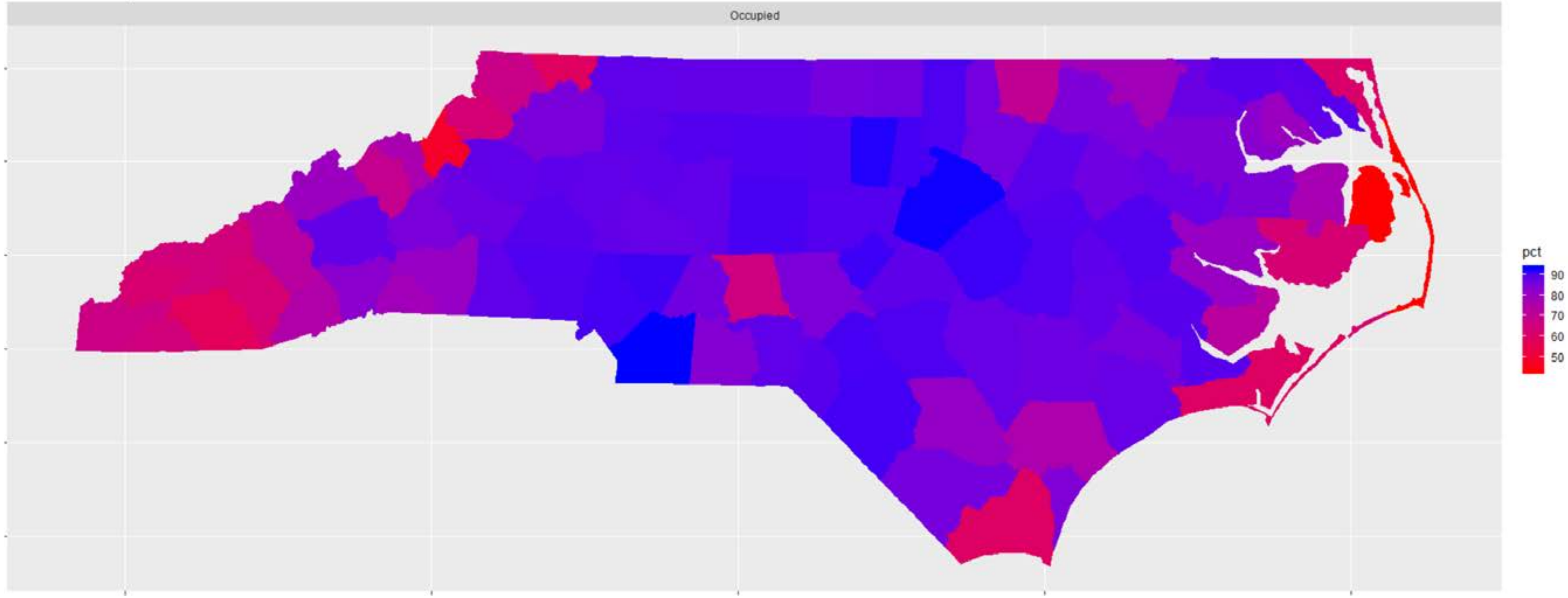
```
# Install 'tidycensus' package
install.packages("tidycensus")
library(tidycensus)

# Key install
census_api_key("ffb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Call housing unit variables from decennial census
housing <- c(Occupied = "H003002",
             Vacant   = "H003003")

# Fetch table on "Housing Unit"
nc.housing.map <- get_decennial(geography = "county",
                                variables = housing,
                                year = 2010,
                                summary_var = "H003001",
                                state = "NC",
                                geometry = TRUE)
```

Let’s break it down…

```
# Install 'tidycensus' package
install.packages("tidycensus")
library(tidycensus)

# Key install
census_api_key("ffb9db49ee4b4563c2e5q5h922ei4ale89d3eafe6")

# Call housing unit variables from decennial census
housing <- c(Occupied = "H003002",
             Vacant   = "H003003")
```

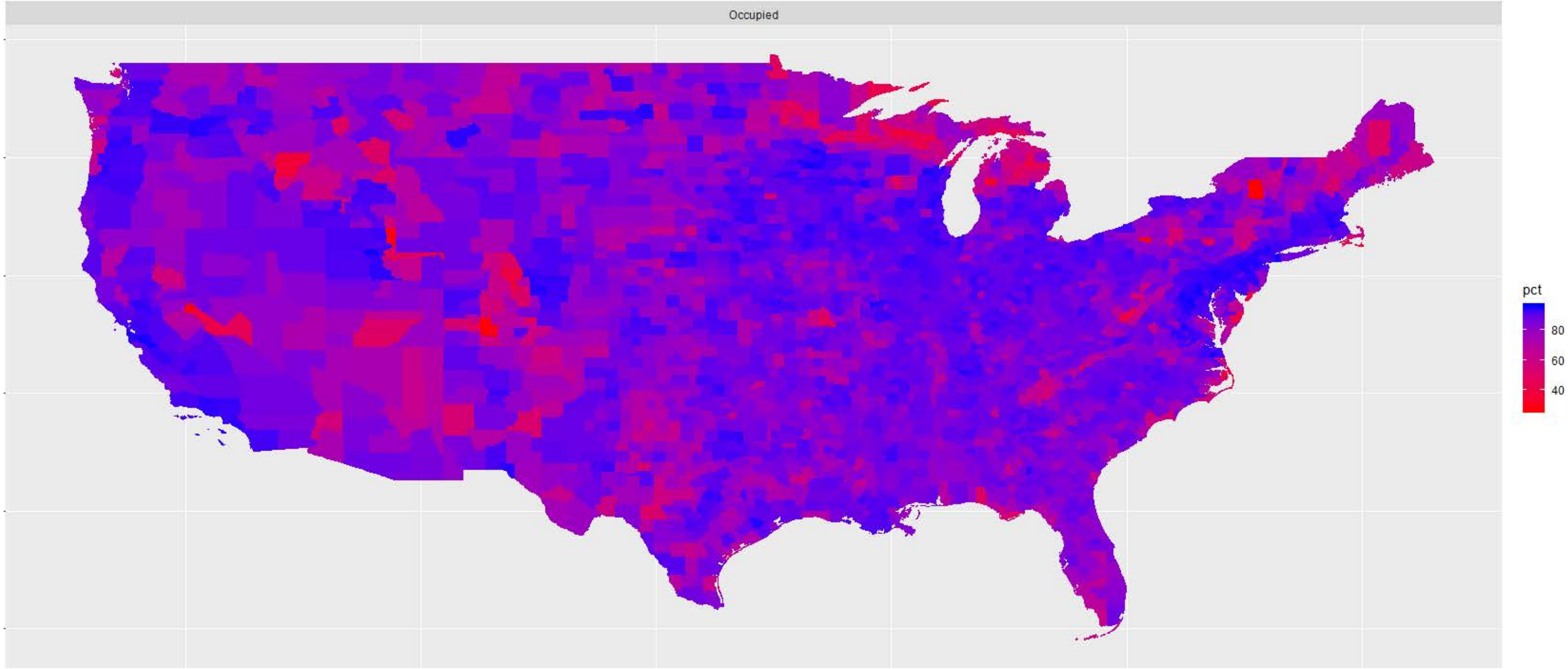
```
# Fetch table on "Housing Unit"
nc.housing.map <- get_decennial(geography = "county",
                                variables = housing,
                                year = 2010,
                                summary_var = "H003001",
                                state = "NC",
                                geometry = TRUE)
```

Figure 2. Percent of Occupied Housing Units in North Carolina, 2010 Census



Source: 2010 Decennial Census, U.S. Census Bureau

Figure 3. Percent of Occupied Housing Units in the Continental U.S., 2010 Census



Source: 2010 Decennial Census, U.S. Census Bureau

Additional R Packages

- **blsAPI:** Allows users to request data for one or multiple series through the BLS API. **https://cran.r-project.org/web/packages/blsAPI/blsAPI.pdf**

- **rHealthDataGov:** Access and select subsets of data from HealthData.gov. **https://cran.r-project.org/web/packages/rHealthDataGov/rHealthDataGov.pdf**

- **medicare:** Package for obtaining and cleaning Medicare public use files. **https://cran.r-project.org/web/packages/medicare/medicare.pdf**

- **iPUMSR:** Import census, survey, and geographic data from IPUMS **https://cran.r-project.org/web/packages/ipumsr/ipumsr.pdf**

GitHub

- **CDC Vital Statistics:** **https://github.com/Mikuana/vitalstatistics**
- **Department of Education:** **https://github.com/UrbanInstitute/education-data-package-r**
  - Integrated Postsecondary Education Data System (IPEDS)
  - Common Core of Data (CCD)
- **Bureau of Labor Statistics:** **https://github.com/keberwein/blscrapeR**

Questions?

Kelsey Farson Gray

kgray@insightpolicyresearch.com