STUDY SERIES
(Computing #2022-01)

Browsing the 2010 Census SF2 Summary File with R

Andrew M. Raim,
James A. Livsey,
Kyle M. Irimata

Center for Statistical Research & Methodology
Research and Methodology Directorate
U.S. Census Bureau
Washington, D.C. 20233

Report Issued: August 15, 2022

Browsing the 2010 Census SF2 Summary File with R

Andrew M. Raim, James A. Livsey, and Kyle M. Irimata
Center for Statistical Research & Methodology, U.S. Census Bureau*

2022-08-15 08:30:44

Abstract

The U.S. Census Bureau releases tabulations based on the decennial census to the public in multiple formats. Data provided on the Census Bureau website may be especially useful to researchers in secure computing environments where network access is restricted. However, users may find the format of this data difficult to use compared to other formats such as the website https://data.census.gov. This paper provides a tutorial for experienced R users to consume census data from the website, specifically focusing on summary File SF2 from the 2010 decennial census. We introduce the `sfreader` package which provides utility functions and helper tables to facilitate use of the data. Examples demonstrate the organization of tables, geographies, and races among the 2010 SF2 files. Use of `sfreader` and widely adopted R packages such as `tidyverse` and `sf` ("Simple Features") is demonstrated to navigate and process the data.

Contents

1 Introduction

The U.S. Census Bureau conducts a census of the United States population every ten years to support congressional redistricting and provide information on basic demographic characteristics of both individuals and households. In the 2010 decennial census, population data were primarily released through Summary File 1 (SF1) and Summary File 2 (SF2) products. Each product is composed of 100% of the population of the U.S. and provides tabulations on demographic characteristics including age, sex, ethnicity, race, household relationships, and residential tenure. SF1 provides summaries of the major race groups, as required by Public

*andrew.raim@census.gov, james.a.livsey@census.gov, kyle.m.irimata@census.gov, U.S. Census Bureau, Washington, DC, 20233, U.S.A. Second and third authors contributed equally; order was determined by a call to the R `sample` function. Disclaimer: This article is released to inform interested parties of ongoing research and to encourage discussion of work in progress. Any views expressed are those of the authors and not those of the U.S. Census Bureau.

Law 94-171 (PL94) in support of legislative redistricting, including such categories as White, Black, American Indian/Alaska Native (AIAN), Asian/Pacific Islander, and "some other race". The PL94-171 redistricting data are also released in a separate census redistricting data product. In comparison, SF2 includes additional detailed race groups beyond these major race groups, as well as AIAN tribal breakdowns. Tables similar to those released in SF2 for a more comprehensive set of tribal groupings are released in the American Indian and Alaska Native Summary File (AIANSF) for the "alone" and "alone or in combination" population.

Tabulations from the decennial census are published in several forms which can be accessed by the public. The website https://data.census.gov provides a graphical interface for general browsing and exploration of Census Bureau data products. This website is especially suitable for working with specific portions of the data over an internet connection, and for users who are first becoming acquainted with the data. A number of other services such as National Historical Geographic Information System (https://www.nhgis.org) also host census data and provide web interfaces to facilitate access to it. For programmatic access to the data, the website https://api.census.gov provides access to the data through an API.

Users of these data may be required to work in restricted environments which do not allow internet access. This is often a consideration in projects at the Census Bureau and with researchers at Research Data Centers (RDCs) who collaborate with the Census Bureau. Although the publicly released census data are not considered sensitive, they are often used in conjunction with data that are protected by laws such as Title 13 (https://www.census.gov/about/policies/privacy/data_stewardship.html) which require a secure computing environment. In such settings, it can be more convenient to work with the complete summary files which have been packaged for download and can be used offline without a connection to the internet. This form of the summary files is provided through the Census Bureau website; e.g. https://www2.census.gov/census_2010/05-Summary_File_2/ for the 2010 SF2 summary file. The download is packaged in a form that makes the data compact but potentially difficult to use. The Census Bureau typically provides some tools to help users process the summary file contents, such as Microsoft Access databases and SAS scripts, while other tools are developed by the user community. A summary file format is also used for another Census Bureau data product, the American Community Survey (ACS), though it differs somewhat from the decennial census format (https://www.census.gov/programs-surveys/acs/data/summary-file.html).

This paper provides a methodology and R package to allow experienced R users [R Core Team, 2022] to navigate and extract information from the downloadable summary files. We explicitly work with downloaded (offline) data files since a primary motivation for this work is aiding research done in a secure computing environment where internet access and API calls may be discouraged. R is one of the primary languages for statistical computing, graphics, and data analysis. It has attracted a dedicated community of users as an open source platform with a relatively easy to learn but powerful programming language. The development of tools such as the `tidyverse` [Wickham et al., 2019], `ggplot2` [Wickham, 2016], and `sf` [Pebesma, 2018] have increased the appeal of R as a preferred tool for data manipulation, plotting, and geographical information systems, respectively.[1] Familiarity with these packages and with R programming is assumed in this paper.[2] We will focus specifically on the 2010 SF2 summary file. The authoritative source of information for interpreting the 2010 SF2 summary file format and its contents is the technical documentation [U.S. Census Bureau, 2011]. However, use of the technical documentation without software tools can become burdensome because of the need to traverse a number of structures and codes to make use of the contents. The `sfreader` package is introduced for this reason; it provides a collection of helper tables and functions which may be used with the tidyverse (or other data manipulation tools) to facilitate access to the data. This enables larger scale programming tasks which may involve looping over states, tables, or other entities. Beyond the `tidyverse`, the `sfreader` package has minimal prerequisites which may be beneficial in secure computing environments where contributed packages may not be readily available. `sfreader` also does not rely on an internet connection. This paper and the `sfreader` package are intended as a companion to the technical documentation and other materials included in the summary file but not as a substitute.

There are a number of existing packages to access census data from R. The `tidycensus` package [Walker and

[1]The reader should be aware of two distinct uses of the "sf" acronym. It can be taken to mean "summary file", or to refer to the `sf` package whose full title is "Simple Features for R". The intended meaning should be clear from the context.

[2]Cheatsheets for each of the packages mentioned, along with a nmber of others, can be found at https://www.rstudio.com/resources/cheatsheets/.

Herman, 2022] makes calls to the Census data API which assumes access to the internet. The `totalcensus` package [Li, 2021] is more oriented toward offline processing of census data, caching files as requested during package use. Therefore, it may be possible to precache all desired data in a secure environment for offline use. At the time of this writing, `totalcensus` focuses its support of decennial census data on summary file SF1 for years 2000 and 2010, in particular the urban/rural updates. `totalcensus` provides some conveniences such as associating latitude/longitude information to geographies, but in this case it may be desirable to associate shapes to describe entire areal units. For the release of the more recent 2020 PL94 data, a collection of R scripts have been provided with the data by the Census Bureau. These scripts are set up to be edited manually for each new data construction but provide a reference implementation for querying the data. The `PL94171` package [McCartan and Kenny, 2021] provides a suite of functions to process PL94 summary files for years 2000, 2010, and 2020. It is interesting to note the overlap in these projects as well as the practical difficulty in supporting all summary files for all recent releases of decennial census data. One reason for a lack of overall support of all products is that subtle changes in table structure or indentation level can impact the query style required.

The remainder of the paper proceeds as follows. In Section 2, we begin to explore files contained in the 2010 SF2 summary file without additional tools. Section 3 introduces our `sfreader` package to facilitate navigation of the data. Section 4 provides a demonstration which starts with a vague question and gradually tracks down data which can be used to gain insight. Finally, Section 5 concludes the paper.

We conclude this section with some computational notes, especially for readers who wish to follow along. Relevant web resources are given throughout the paper; their locations (URLs) and content are subject to change. The string R> is used in code displays to represent the R prompt. A complete set of the code chunks used in the paper is provided as a supplemental R script to make them easier to run. The `sfreader` R package may be installed from Github using the following R command.

```r
R> devtools::install_github("andrewraim/sfreader")
```

The paper uses a particular set of summary file data. It may be downloaded from within R using the following code.

```r
library(curl)
library(dplyr)
library(sfreader)

sf = SF2_2010()

dest_dir = "my_files"
dat_url = get_data_urls(sf) %>% filter(FIPS == "11") %>% pull(URL)
dir.create(dest_dir)
local_zip_file = basename(dat_url)
curl_download(url = dat_url, destfile = local_zip_file)
unzip(local_zip_file, exdir = dest_dir)

dest_dir = "my_shapefiles"
dat_url = "https://www2.census.gov/geo/tiger/TIGER2010/TRACT/2010/tl_2010_11_tract10.zip"
dir.create(dest_dir)
local_zip_file = basename(dat_url)
curl_download(url = dat_url, destfile = local_zip_file)
unzip(local_zip_file, exdir = dest_dir)
```

The particular SF2 data explored in this paper have been selected for demonstration purposes; similar concepts should extend to other uses of the 2010 SF2 summary file which may be of interest to the reader.

Readers unfamiliar with the pipe operator (`%>%`) should note that it allows composition of two or more function calls to be expressed sequentially rather than as nested calls. The pipe operator is used extensively with the tidyverse so that involved data transformation remain readable, and also used extensively in this paper. Here is a non-tidyverse example—of computing $\sum_{x=1}^{100} x^2$—carried out via nested calls versus the pipe operator.

```
R> pow = function(x, p) x^p
R> sum(pow(seq(1, 100), 2))
[1] 338350
R> seq(1, 100) %>% pow(2) %>% sum()
[1] 338350
```

On the last line, the result of `seq` is passed as the first argument (`x`) of the `pow` function. The result of `pow` is then passed as the argument of `sum`.

2 Summary File Contents

The 2010 SF2 summary file release is currently available to the public from the URL https://www.census.gov/data/datasets/2010/dec/summary-file-2.html. From here, users can obtain the technical documentation in the form of a 356 page PDF [U.S. Census Bureau, 2011]. In addition, instructions can be found at the SF2 website on use of a supplied Microsoft Access database to work with the data. A link to the data files (https://www2.census.gov/census_2010/05-Summary_File_2/) can also be found here. Let us explore the data without the aid of additional tools until it starts becoming burdensome.

The data files consist of 53 folders corresponding to the 50 states of the U.S., folders for District of Columbia and Puerto Rico, and a folder for the United States. Within each folder is a README and data contained in a ZIP file. Examining the ZIP file for District of Columbia, we find the following contents. The file `dc2010.sf2.prd.packinglist.txt` is a packing list outlining the remaining contents of the ZIP file. The file `dcgeo2010.sf2` is a "geo" file which provides geographic, race/ethnicity, and other identifying information for data records. It is encoded using a fixed width format specified in the technical documentation. The remaining contents `dc001012010.sf2`, ..., `dc599112010.sf2` of the ZIP file are 1,430 CSVs containing tabulation data. We find the following contents in the first few lines of `dc001022010.sf2`.

```
R> readLines("my_files/dc001022010.sf2", n = 5)
[1] "SF2ST,DC,001,02,0000001,601723,0,0,0,0,601723"
[2] "SF2ST,DC,001,02,0000002,601723,0,0,0,0,601723"
[3] "SF2ST,DC,001,02,0000003,601723,0,0,0,0,601723"
[4] "SF2ST,DC,001,02,0000004,601723,0,0,0,0,601723"
[5] "SF2ST,DC,001,02,0000005,601723,0,0,0,0,601723"
```

Here the first five columns provide identifying information for the record and the remaining six columns describe tabulation data; however, we will need to do some work to put this into an interpretable form. For the remainder of the document, we will assume that the ZIP file has been downloaded and extracted locally into a folder `my_files`.

Let us now take a step back and see how the tabulation data has been packaged into files. The summary file consists of a number of tables described in Chapter 5 of the technical documentation. Each table is based on a given universe, which is the type of entry or entity being tabulated [U.S. Census Bureau, 2011]. For example, the first few entries of the table are given in Table 1. The tabulations are partitioned into eleven **segments**, labeled `01`, ..., `11`, so that each segment contains one or more tabulations. The data are also organized by **iteration codes** which describe races and ethnicities used in tabulations. Iteration codes are described in Appendix H of the technical documentation; several are displayed in Table 2. If we are not focusing on any particular race and ethnicity, we may select code `001` which represents "Total Population". We can now interpret the names of the CSV files; taking `dc001022010.sf2` as an example:

- `dc`: file is from the District of Columbia dataset,
- `001`: file is for iteration code `001`,
- `02`: file contains data for segment `02`,
- `2010`: file is for year 2010,

Table 1: Several of the tabulations listed in Chapter 5 of the technical documentation.

Table number	Title	# of cells	Universe
PCT1	TOTAL POPULATION	1	Total population
PCT2	URBAN AND RURAL	6	Total population
PCT3	SEX BY AGE	209	Total population
PCT33	FAMILY TYPE BY PRESENCE AND AGE OF RELATED CHILDREN	20	Families
HCT1	URBAN AND RURAL	6	Occupied housing units
HCT2	TENURE	4	Occupied housing units

Table 2: Several of the iterations listed in Appendix H of the technical documentation.

Code	Description
001	Total Population
002	White alone
003	White alone or in combination with one or more other races
004	Black or African American alone
005	Black or African American alone or in combination with one or more other races
006	American Indian and Alaska Native alone (300, A01–Z99)
278	American Indian tribes, specified, alone (A01–M38, T01–Z99)

- `sf2`: file is from summary file SF2.

Now, to interpret the contents of the file, we refer to Chapter 6: Table (Matrix) Section of the technical documentation to find that segment `02` contains a single table PCT2 "Urban and Rural" which has six tabulation variables along with five variables to identify the record; these are listed in Table 3 for convenience. One important feature is that names of tabulation variables contain the table name as a prefix: e.g. variable `PCT0020001` begins with table name `PCT002` and the table number in both cases is zero-padded to have three digits. Another important feature is that the table variables form a hierarchy where the Total count `PCT0020001` is decomposed into `PCT0020002`, `PCT0020005`, and `PCT0020006`. Furthermore, the Urban count `PCT0020002` is decomposed into `PCT0020003` and `PCT0020004`. The hierarchy is notated via indentation level in Table 3 as well as in the technical documentation. From our display of the first few lines of data, we can see that only totals have been given. Furthermore, it is clear that the following field will be constant throughout the file:

- `FILEID = SF2ST`: file type is SF2 for a state,
- `STUSAB = DC`: state is District of Columbia,
- `CHARITER = 001`: iteration code is `001` for total population, and
- `CIFSN = 02`: data is for segment `02`.

The field `LOGRECNO` changes for each record and maps to information in the geo file which identifies the meaning of the tabulation for a given record. This type of exploration without additional aid can become unwieldy quickly and hence it is helpful to introduce our `sfreader` package while continuing our exploration.

3 The sfreader Package

The `sfreader` package provides some tools which will facilitate navigation of the summary files. These tools include a collection of helper tables and utility functions. Let us first load the package and some prerequisites.

```
R> library(dplyr)
R> library(sfreader)
```

Table 3: Columns of the file `dc001022010.sf2`, interpreted using Chapter 6: Table (Matrix) Section of the technical documentation.

Position	Field	Description
1	FILEID	File Identification
2	STUSAB	State/U.S. Abbreviation (USPS)
3	CHARITER	Characteristic Iteration
4	CIFSN	Characteristic Iteration File Sequence Number
5	LOGRECNO	Logical Record Number
6	PCT0020001	Total
7	PCT0020002	. . . Urban
8	PCT0020003	. . . . . . Inside urbanized areas
9	PCT0020004	. . . . . . Inside urban clusters
10	PCT0020005	. . . Rural
11	PCT0020006	. . . Not defined for this file

3.1 Helper Tables

The fixed-width format of the geo file is described in Figure 2.5 of the technical specification. Field positions and widths are needed to interpret a geo file. Furthermore, only certain fields are present depending on the **summary level** of a record; this indicates whether a record pertains to a state, county, or other geographical entity. If a record describes a state, for example, county-level fields will be empty. The `sfreader` package contains a helper table with this fixed-width format.

```
R> print(sf2_2010_geo_format, n = 10)
# A tibble: 101 x 6
      ID FIELD    START_POS FIELD_SIZE DATA_TYPE DESCRIPTION
   <int> <chr>        <int>      <int> <chr>     <chr>
 1     3 FILEID           1          6 c         File Identification
 2     4 STUSAB           7          2 c         State/US-Abbreviation (USPS)
 3     5 SUMLEV           9          3 c         Summary Level
 4     6 GEOCOMP         12          2 c         Geographic Component
 5     7 CHARITER        14          3 c         Characteristic Iteration
 6     8 CIFSN           17          2 c         Characteristic Iteration File ~
 7     9 LOGRECNO        19          7 c         Logical Record Number
 8    10 REGION          26          1 c         Region
 9    11 DIVISION        27          1 c         Division
10    12 STATE           28          2 c         State (FIPS)
# ... with 91 more rows
# i Use `print(n = ...)` to see more rows
```

In addition, a helper function is provided to parse geo files so that `sf2_2010_geo_format` need not be used directly; this function will be discussed later in the section.

The most useful table that a user will directly interact with is `sf2_2010_fields`. We see its usage in the following.

```
R> sf2_2010_fields %>% filter(SEGMENT == '02') %>% select(-SEGMENT, -NAME)
# A tibble: 11 x 5
   POSITION FIELD      PARENT_FIELD TABLE  DESCRIPTION
      <int> <chr>      <chr>        <chr>  <chr>
 1        1 FILEID     <NA>         HEAD   File Identification
 2        2 STUSAB     <NA>         HEAD   State/US-Abbreviation (USPS)
 3        3 CHARITER   <NA>         HEAD   Characteristic iteration
 4        4 CIFSN      <NA>         HEAD   Characteristic Iteration File Sequen~
 5         5 LOGRECNO   <NA>          HEAD   Logical Record Number
 6         6 PCT0020001 <NA>          PCT002 Total
 7         7 PCT0020002 PCT0020001    PCT002 Total: Urban
 8         8 PCT0020003 PCT0020002    PCT002 Total: Urban: Inside urbanized areas
 9         9 PCT0020004 PCT0020002    PCT002 Total: Urban: Inside urban clusters
10        10 PCT0020005 PCT0020001    PCT002 Total: Rural
11        11 PCT0020006 PCT0020001    PCT002 Total: Not defined for this file
```

We have displayed the definitions for variables in segment 02, which were needed to interpret the columns of dc001022010.sf2 CSV file in Section 2. The hierarchy of the table has been captured in two ways: via the PARENT_FIELD column and within the DESCRIPTION. For example, the total PCT0020001 is decomposed into PCT0020002, PCT0020003, and PCT0020006; therefore PCT0020002, PCT0020003, and PCT0020006 specify PCT0020001 as PARENT_FIELD. The PARENT_FIELD of PCT0020001 is NA to indicate that it is at the top of hierarchy. The DESCRIPTION reflects the path in the hierarchy from the first ancestor so that "A: B: C" indicates that field A is a parent of field B, which in turn is the parent of field C. The TABLE column indicates that the last six variables belong to tabulation PCT002, while the first five variables are listed as HEAD to indicate that they are common headers which do not belong to a specific tabulation.

Iteration codes are provided in a helper table sf2_2010_iterations.

```
R> print(sf2_2010_iterations, n = 7)
# A tibble: 331 x 2
  CODE  DESCRIPTION
  <chr> <chr>
1 001   Total Population
2 002   White alone
3 003   White alone or in combination with one or more other races
4 004   Black or African American alone
5 005   Black or African American alone or in combination with one or more othe~
6 006   American Indian and Alaska Native alone (300, A01-Z99)
7 009   American Indian and Alaska Native alone or in combination with one or m~
# ... with 324 more rows
# i Use `print(n = ...)` to see more rows
```

The available tabulations are provided in a helper table sf2_2010_tables.

```
R> print(sf2_2010_tables, n = 10)
# A tibble: 71 x 2
   TABLE  DESCRIPTION
   <chr>  <chr>
 1 PCT001 TOTAL POPULATION [1]
 2 PCT002 URBAN AND RURAL [6]
 3 PCT003 SEX BY AGE [209]
 4 PCT004 MEDIAN AGE BY SEX [3]  (1 expressed decimal)
 5 PCT005 SEX BY AGE FOR THE POPULATION IN HOUSEHOLDS [49]
 6 PCT006 POPULATION IN HOUSEHOLDS BY AGE (ITERATED BY RACE, HISPANIC OR LATINO~
 7 PCT007 AVERAGE HOUSEHOLD SIZE BY AGE [3]  (2 expressed decimals)
 8 PCT008 HOUSEHOLD TYPE [9]
 9 PCT009 HOUSEHOLD SIZE BY HOUSEHOLD TYPE BY PRESENCE OF OWN CHILDREN [19]
10 PCT010 HOUSEHOLDS BY PRESENCE OF PEOPLE UNDER 18 YEARS BY HOUSEHOLD TYPE BY ~
# ... with 61 more rows
# i Use `print(n = ...)` to see more rows
```

Definitions of summary levels are provided in a helper table sf2_2010_sumlev.

```
R> sumlev_state_notgq = sf2_2010_sumlev %>% filter(TYPE == "STATE", GQ == FALSE)
R> print(sumlev_state_notgq, n = 3)
# A tibble: 52 x 5
  CODE  PARENT_CODE DESCRIPTION                           TYPE  GQ
  <chr> <chr>       <chr>                                 <chr> <lgl>
1 040   <NA>        State                                 STATE FALSE
2 050   040         State-County                          STATE FALSE
3 060   050         State-County-County Subdivision STATE FALSE
# ... with 49 more rows
# i Use `print(n = ...)` to see more rows
```

Note that summary levels form a hierarchy which is expressed here by the PARENT_CODE field. For example, State-County (CODE = 050) is considered a child of State (CODE = 040). Different summary levels hierarchies are available for state versus national files and for group quarters versus records which are not group quarters. For instance, the national file contains some summary levels that cross state lines, while the state files include some additional smaller geographies. Summary levels of Puerto Rico are interpreted somewhat differently then other states. We recommend users review the technical documentation to best understand the hierarchy and available summary levels for each file.

Tabulations for given geographies may be subsetted to specific **geographic components**. These are given in the following helper table.

```
R> print(sf2_2010_geocomp, n = 5)
# A tibble: 114 x 2
  CODE  DESCRIPTION
  <chr> <chr>
1 00    Not a geographic component
2 01    Urban
3 04    Urban-in urbanized area
4 05    Urban-in urbanized area of 5,000,000 or more population
5 06    Urban-in urbanized area of 2,500,000 to 4,999,999 population
# ... with 109 more rows
# i Use `print(n = ...)` to see more rows
```

The following helper table provides names, Federal Information Processing Standards (FIPS) codes and abbreviations for states (and the US) which are used in the summary file. The FIPS code for US is left as NA.

```
R> tail(sf2_2010_states, n = 10)
# A tibble: 10 x 3
   NAME          FIPS  ABBREV
   <chr>         <chr> <chr>
 1 Tennessee     47    TN
 2 Texas         48    TX
 3 United_States <NA>  US
 4 Utah          49    UT
 5 Vermont       50    VT
 6 Virginia      51    VA
 7 Washington    53    WA
 8 West_Virginia 54    WV
 9 Wisconsin     55    WI
10 Wyoming       56    WY
```

3.2 Functions

Much of the functionality in the `sfreader` package is provided via S4 methods [e.g., Wickham, 2019, Chapter 15]. This will potentially allow it to be extended to other summary files in the future. For now, let us create an object of type `SF2_2010`.

```
R> sf = SF2_2010()
```

A `get_data_urls` method is provided to get URLs to the ZIP files which contain the data. Since we focus on an off-line workflow this is thought of as preprocessing step to procure necessary files to work with. If needed, these files could then be securely transferred.

```
R> print(get_data_urls(sf), n = 5)
# A tibble: 53 x 2
  FIPS  URL
  <chr> <chr>
1 01    https://www2.census.gov/census_2010/05-Summary_File_2/Alabama/al2010.sf~
2 02    https://www2.census.gov/census_2010/05-Summary_File_2/Alaska/ak2010.sf2~
3 04    https://www2.census.gov/census_2010/05-Summary_File_2/Arizona/az2010.sf~
4 05    https://www2.census.gov/census_2010/05-Summary_File_2/Arkansas/ar2010.s~
5 06    https://www2.census.gov/census_2010/05-Summary_File_2/California/ca2010~
# ... with 48 more rows
# i Use `print(n = ...)` to see more rows
R> get_data_urls(sf) %>% pull(URL) %>% head(n = 1)
[1] "https://www2.census.gov/census_2010/05-Summary_File_2/Alabama/al2010.sf2.zip"
```

An optional `base_url` argument may be provided if the URL happens to move at some point, or to request the data from an alternate URL.

```
R> get_data_urls(sf, base_url = "http://path/to") %>% pull(URL) %>% head(n = 1)
[1] "http://path/to/Alabama/al2010.sf2.zip"
```

A `read_geo` method is provided to read the fixed-width geo file into a `tibble`.

```
R> read_geo(sf, "my_files/dcgeo2010.sf2")
# A tibble: 590 x 101
   FILEID STUSAB SUMLEV GEOCOMP CHARITER CIFSN LOGRECNO REGION DIVISION STATE
   <chr>  <chr>  <chr>  <chr>   <chr>    <chr> <chr>    <chr>  <chr>    <chr>
 1 SF2ST  DC     040    00      001      <NA>  0000001  3      5        11
 2 SF2ST  DC     040    A0      001      <NA>  0000002  3      5        11
 3 SF2ST  DC     040    A1      001      <NA>  0000003  3      5        11
 4 SF2ST  DC     040    C0      001      <NA>  0000004  3      5        11
 5 SF2ST  DC     040    C1      001      <NA>  0000005  3      5        11
 6 SF2ST  DC     050    00      001      <NA>  0000006  3      5        11
 7 SF2ST  DC     060    00      001      <NA>  0000007  3      5        11
 8 SF2ST  DC     070    00      001      <NA>  0000008  3      5        11
 9 SF2ST  DC     080    00      001      <NA>  0000009  3      5        11
10 SF2ST  DC     080    00      001      <NA>  0000010  3      5        11
# ... with 580 more rows, and 91 more variables: COUNTY <chr>, COUNTYCC <chr>,
#   COUNTYSC <chr>, COUSUB <chr>, COUSUBCC <chr>, COUSUBSC <chr>, PLACE <chr>,
#   PLACECC <chr>, PLACESC <chr>, TRACT <chr>, BLKGRP <chr>, BLOCK <chr>,
#   IUC <chr>, CONCIT <chr>, CONCITCC <chr>, CONCITSC <chr>, AIANHH <chr>,
#   AIANHHFP <chr>, AIANHHCC <chr>, AIHHTLI <chr>, AITSCE <chr>, AITS <chr>,
#   AITSCC <chr>, TTRACT <chr>, TBLKGRP <chr>, ANRC <chr>, ANRCCC <chr>,
#   CBSA <chr>, CBSASC <chr>, METDIV <chr>, CSA <chr>, NECTA <chr>, ...
# i Use `print(n = ...)` to see more rows, and `colnames()` to see all variable names
```

Notice the LOGRECNO field, which joins to LOGRECNO in the CSV file we examined in Section 2.

The interpret_data_filenames method splits names of data files into fields and places them in a tibble. This can be helpful when processing the files.

```
R> files = c("my_files/dc001012010.sf2",
+   "my_files/dc001022010.sf2",
+   "my_files/dc599112010.sf2")
R> interpret_data_filenames(sf, files)
# A tibble: 3 x 6
  STATE ITERATION_CODE SEGMENT YEAR  SF_TYPE PATH
  <chr> <chr>          <chr>   <chr> <chr>   <chr>
1 dc    001            01      2010  sf2     my_files/dc001012010.sf2
2 dc    001            02      2010  sf2     my_files/dc001022010.sf2
3 dc    599            11      2010  sf2     my_files/dc599112010.sf2
```

3.3 Summary File Contents, Continued

With the tools in the sfreader package, we can finish our initial exploration of the SF2 files for District of Columbia. We are not interested in any specific iteration code or geographic component for now, so we will use codes CHARITER = 001 and GEOCOMP = 00 respectively to access unfiltered totals. We select tract-level data which is coded SUMLEV = 140.

The above codes may be identified from the SF2 technical documentation or the helper tables provided in Section 3.1. We mention the justification of SUMLEV = 140 from the SF2 technical documentation to avoid confusion between multiple tract levels, seen in the following output.

```
R> sf2_2010_sumlev %>% filter(TYPE == "STATE", GQ == FALSE) %>%
+   filter(grepl("tract", DESCRIPTION, ignore.case = TRUE)) %>%
+   select(-TYPE, -GQ)
# A tibble: 4 x 3
  CODE  PARENT_CODE DESCRIPTION
  <chr> <chr>       <chr>
1 080   070         State-County-County Subdivision-Place/Remainder-Census Tract
2 140   050         State-County-Census Tract
3 144   140         State-County-Census Tract-American Indian Area/Alaska Nativ~
4 158   155         State-Place-County-Census Tract
```

Chapter 4 "Summary Level Sequence Chart" lists the hierarchy of summary level codes. Code 140 is seen to represent tract under the State-County-Census Tract hierarchy. Other codes involving tracts are specific to more specialized geographies such as places or subdivisions. Figure 2.5 "Geographic Header Record—Summary File 2 State" verifies that corresponding records have fields at the tract level but are not specific to place or subdivision.

We now proceed and overlay the variable names onto the CSV data.

```
R> dat_segment = sf2_2010_fields %>% filter(SEGMENT == '02')
R> dat_csv = read_csv("my_files/dc001022010.sf2", col_names = dat_segment$FIELD) %>%
+   select(-FILEID, -STUSAB, -CHARITER, -CIFSN)
```

Now read the relevant records from the geo file.

```
R> dat_geo = read_geo(sf, "my_files/dcgeo2010.sf2") %>%
+   filter(SUMLEV == '140') %>%
+   filter(GEOCOMP == '00')
```

Finally, join the CSV and geo data together.

```
R> dat_csv %>%
+   inner_join(dat_geo, c("LOGRECNO" = "LOGRECNO")) %>%
+   select(TRACT, starts_with("PCT002"), everything())
# A tibble: 178 x 107
   TRACT  PCT002~1 PCT00~2 PCT00~3 PCT00~4 PCT00~5 PCT00~6 LOGRE~7 FILEID STUSAB
   <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <chr>   <chr>  <chr>
 1 000100     4890       0       0       0       0    4890 0000187 SF2ST  DC
 2 000201     3916       0       0       0       0    3916 0000188 SF2ST  DC
 3 000202     5425       0       0       0       0    5425 0000189 SF2ST  DC
 4 000300     6233       0       0       0       0    6233 0000190 SF2ST  DC
 5 000400     1455       0       0       0       0    1455 0000191 SF2ST  DC
 6 000501     3376       0       0       0       0    3376 0000192 SF2ST  DC
 7 000502     3189       0       0       0       0    3189 0000193 SF2ST  DC
 8 000600     4539       0       0       0       0    4539 0000194 SF2ST  DC
 9 000701     4620       0       0       0       0    4620 0000195 SF2ST  DC
10 000702     3364       0       0       0       0    3364 0000196 SF2ST  DC
# ... with 168 more rows, 97 more variables: SUMLEV <chr>, GEOCOMP <chr>,
#   CHARITER <chr>, CIFSN <chr>, REGION <chr>, DIVISION <chr>, STATE <chr>,
#   COUNTY <chr>, COUNTYCC <chr>, COUNTYSC <chr>, COUSUB <chr>, COUSUBCC <chr>,
#   COUSUBSC <chr>, PLACE <chr>, PLACECC <chr>, PLACESC <chr>, BLKGRP <chr>,
#   BLOCK <chr>, IUC <chr>, CONCIT <chr>, CONCITCC <chr>, CONCITSC <chr>,
#   AIANHH <chr>, AIANHHFP <chr>, AIANHHCC <chr>, AIHHTLI <chr>, AITSCE <chr>,
#   AITS <chr>, AITSCC <chr>, TTRACT <chr>, TBLKGRP <chr>, ANRC <chr>, ...
# i Use `print(n = ...)` to see more rows, and `colnames()` to see all variable names
```

Note that there is only one county in the District of Columbia; therefore, the `TRACT` column uniquely identifies records for the 178 tracts within the data. We have ordered the columns with the `select` statement so that the most interesting ones appear in the display. The analyst could also consider dropping the remaining 100 columns which have been omitted. Recalling the variable definitions from Section 3, only the columns `PCT0020001` ("Total") and `PCT0020006` ("Total: Not defined for this file") from the hierarchy appear to be present for this dataset.

4 An Example

This section provides a more in-depth example of how the `sfreader` package can be used with the tidyverse to navigate the contents of the summary file. We will start with a somewhat vague research question: do households in DC with children tend to be rentals? We could consult the technical document for relevant tables and fields, but here we will make use of R to browse the dataset.

4.1 Wrangling the Data

Let us first search the list of tables for field descriptions that contain any words that begin with "rent". To accomplish this, we use the `grepl` function to search for the string "rent" after a space character.

```
R> sf2_2010_fields %>%
+   filter(grepl("\\brent", DESCRIPTION, ignore.case = TRUE)) %>%
+   select(TABLE, DESCRIPTION)
# A tibble: 73 x 2
   TABLE  DESCRIPTION
   <chr>  <chr>
 1 HCT002 Total: Renter occupied
 2 HCT004 Total population in occupied housing units: Renter occupied
 3 HCT005 Total population in occupied housing units: Renter occupied
 4 HCT007 Total: Renter occupied
 5 HCT007 Total: Renter occupied: 1-person household
 6 HCT007 Total: Renter occupied: 2-person household
 7 HCT007 Total: Renter occupied: 3-person household
 8 HCT007 Total: Renter occupied: 4-person household
 9 HCT007 Total: Renter occupied: 5-person household
10 HCT007 Total: Renter occupied: 6-person household
# ... with 63 more rows
# i Use `print(n = ...)` to see more rows
```

There are a number of descriptions that start with `Total: Renter occupied` or `Total: Renter-occupied`. These all appear to be relevant to our question so we will now list the first few fields in each.

```
dat_tables = sf2_2010_fields %>%
    filter(grepl("Total: Renter.*occupied$", DESCRIPTION)) %>%
    pull(TABLE)

for (i in seq_along(dat_tables)) {
    tab = sf2_2010_fields %>%
        filter(TABLE %in% dat_tables[i]) %>%
        select(FIELD, PARENT_FIELD, NAME)
    print(tab, n = 8)
}
# A tibble: 4 x 3
  FIELD      PARENT_FIELD NAME
  <chr>      <chr>        <chr>
1 HCT0020001 <NA>         Total
2 HCT0020002 HCT0020001   Owned with a mortgage or a loan
3 HCT0020003 HCT0020001   Owned free and clear
4 HCT0020004 HCT0020001   Renter occupied
# A tibble: 17 x 3
  FIELD      PARENT_FIELD NAME
  <chr>      <chr>        <chr>
1 HCT0070001 <NA>         Total
2 HCT0070002 HCT0070001   Owner occupied
3 HCT0070003 HCT0070002   1-person household
4 HCT0070004 HCT0070002   2-person household
5 HCT0070005 HCT0070002   3-person household
6 HCT0070006 HCT0070002   4-person household
7 HCT0070007 HCT0070002   5-person household
8 HCT0070008 HCT0070002   6-person household
# ... with 9 more rows
# i Use `print(n = ...)` to see more rows
# A tibble: 21 x 3
  FIELD      PARENT_FIELD NAME
  <chr>      <chr>        <chr>
1 HCT0080001 <NA>         Total
2 HCT0080002 HCT0080001   Owner occupied
3 HCT0080003 HCT0080002   Householder 15 to 24 years
4 HCT0080004 HCT0080002   Householder 25 to 34 years
5 HCT0080005 HCT0080002   Householder 35 to 44 years
6 HCT0080006 HCT0080002   Householder 45 to 54 years
7 HCT0080007 HCT0080002   Householder 55 to 59 years
8 HCT0080008 HCT0080002   Householder 60 to 64 years
# ... with 13 more rows
# i Use `print(n = ...)` to see more rows
# A tibble: 69 x 3
  FIELD      PARENT_FIELD NAME
  <chr>      <chr>        <chr>
1 HCT0090001 <NA>         Total
2 HCT0090002 HCT0090001   Owner occupied
3 HCT0090003 HCT0090002   Family households
4 HCT0090004 HCT0090003   Husband-wife family
5 HCT0090005 HCT0090004   Householder 15 to 34 years
6 HCT0090006 HCT0090004   Householder 35 to 64 years
7 HCT0090007 HCT0090004   Householder 65 years and over
8 HCT0090008 HCT0090003   Other family
# ... with 61 more rows
# i Use `print(n = ...)` to see more rows
# A tibble: 13 x 3
  FIELD      PARENT_FIELD NAME
  <chr>      <chr>        <chr>
1 HCT0100001 <NA>         Total
2 HCT0100002 HCT0100001   Owner-occupied
3 HCT0100003 HCT0100002   With own children under 18 years
4 HCT0100004 HCT0100003   Under 6 years only
5 HCT0100005 HCT0100003   Under 6 years and 6 to 17 years
6 HCT0100006 HCT0100003   6 to 17 years only
7 HCT0100007 HCT0100002   No own children under 18 years
8 HCT0100008 HCT0100001   Renter-occupied
# ... with 5 more rows
# i Use `print(n = ...)` to see more rows
# A tibble: 13 x 3
  FIELD      PARENT_FIELD NAME
  <chr>      <chr>        <chr>
1 HCT0110001 <NA>         Total
2 HCT0110002 HCT0110001   Owner-occupied
3 HCT0110003 HCT0110002   With related children under 18 years
4 HCT0110004 HCT0110003   Under 6 years only
5 HCT0110005 HCT0110003   Under 6 years and 6 to 17 years
6 HCT0110006 HCT0110003   6 to 17 years only
7 HCT0110007 HCT0110002   No related children under 18 years
8 HCT0110008 HCT0110001   Renter-occupied
# ... with 5 more rows
# i Use `print(n = ...)` to see more rows
# A tibble: 13 x 3
  FIELD      PARENT_FIELD NAME
  <chr>      <chr>        <chr>
1 HCT0120001 <NA>         Total
2 HCT0120002 HCT0120001   Owner-occupied
3 HCT0120003 HCT0120002   With children under 18 years
4 HCT0120004 HCT0120003   Under 6 years only
5 HCT0120005 HCT0120003   Under 6 years and 6 to 17 years
6 HCT0120006 HCT0120003   6 to 17 years only
7 HCT0120007 HCT0120002   No children under 18 years
8 HCT0120008 HCT0120001   Renter-occupied
# ... with 5 more rows
# i Use `print(n = ...)` to see more rows
```

Tables HCT002, HCT007, HCT008, and HCT009 describe ownership level, household size, householder age, and tenure with household type, respectively. Tables HCT010, HCT011, and HCT012 are more relevant to our question; they describe own children, related children, and children regardless of related status, respectively. We will proceed with HCT012.

Recall that we need to specify an iteration code(s), geographic component(s), and summary level(s) to determine which records we want. Let us load the complete geo file.

```
R> dat_geo = read_geo(sf, "my_files/dcgeo2010.sf2")
```

We will assume GEOCOMP = 00 which will not subset by a geographic component. For the summary level, the following query returns the number of records at each summary level.

```
R> dat_geo %>%
+   filter(GEOCOMP == '00') %>%
+   group_by(SUMLEV) %>%
+   summarize(count = n()) %>%
+   inner_join(sf2_2010_sumlev %>% filter(TYPE == "STATE", GQ == FALSE),
+       by = c("SUMLEV" = "CODE")) %>%
+   arrange(SUMLEV)
# A tibble: 20 x 6
   SUMLEV count PARENT_CODE DESCRIPTION                                 TYPE  GQ
   <chr>  <int> <chr>       <chr>                                       <chr> <lgl>
 1 040        1 <NA>        State                                       STATE FALSE
 2 050        1 040         State-County                                STATE FALSE
 3 060        1 050         State-County-County Subdivision             STATE FALSE
 4 070        1 060         State-County-County Subdivision-Place/R~ STATE FALSE
 5 080      178 070         State-County-County Subdivision-Place/R~ STATE FALSE
 6 140      178 050         State-County-Census Tract                   STATE FALSE
 7 155        1 160         State-Place-County                          STATE FALSE
 8 158      178 155         State-Place-County-Census Tract             STATE FALSE
 9 160        1 040         State-Place                                 STATE FALSE
10 320        1 040         State-Metropolitan Statistical Area/Mic~ STATE FALSE
11 321        1 320         State-Metropolitan Statistical Area/Mic~ STATE FALSE
12 322        1 320         State-Metropolitan Statistical Area/Mic~ STATE FALSE
13 323        1 320         State-Metropolitan Statistical Area-Met~ STATE FALSE
14 324        1 323         State-Metropolitan Statistical Area-Met~ STATE FALSE
15 340        1 040         State-Combined Statistical Area             STATE FALSE
16 341        1 340         State-Combined Statistical Area-Metropo~ STATE FALSE
17 500        1 040         State-Congressional District                STATE FALSE
18 610        8 040         State-State Legislative District (Upper~ STATE FALSE
19 871       29 040         State-5-Digit ZIP Code Tabulation Area      STATE FALSE
20 970        1 040         State-School District (Unified)/Remaind~ STATE FALSE
```

DC appears to have meaningful data broken down only by tract, ZIP Code, and State Legislative District geographies. Note that tracts are available in several levels of the hierarchy; we will proceed with tracts identified by SUMLEV = 140 which are directly under counties and do not involve the place or county subdivision geographies.

Next we will select some characteristic iterations to study. Note that CHARITER = 001 (“Total Population”) is listed as a placeholder in every entry of the geo file, so this field will not be used to select records.

```
R> dat_geo %>%
+   group_by(CHARITER) %>%
+   summarize(n = n())
# A tibble: 1 x 2
  CHARITER     n
  <chr>    <int>
1 001        590
```

Characteristic iterations with the qualifier "alone", and not "alone or in combination" nor "alone or in any combination", describe individuals who reported only a single race entry on their census form, or multiple entries which fall into the same major race group. We can browse the iteration code helper table to see which of these are available.

```
R> sf2_2010_iterations %>%
+   filter(grepl("\\balone", DESCRIPTION, ignore.case = TRUE)) %>%
+   filter(!grepl("\\bor in combination", DESCRIPTION, ignore.case = TRUE)) %>%
+   filter(!grepl("\\bor in any combination", DESCRIPTION, ignore.case = TRUE))
# A tibble: 114 x 2
   CODE  DESCRIPTION
   <chr> <chr>
 1 002   White alone
 2 004   Black or African American alone
 3 006   American Indian and Alaska Native alone (300, A01-Z99)
 4 012   Asian alone (400-499)
 5 013   Asian Indian alone (400-401)
 6 014   Bangladeshi alone (402)
 7 015   Cambodian alone (405-409)
 8 016   Chinese alone (410-419)
 9 017   Chinese (except Taiwanese) alone (410-411)
10 018   Taiwanese alone (412-419)
# ... with 104 more rows
# i Use `print(n = ...)` to see more rows
```

We note that not all iteration codes will have data available at every level of the geography. For example, there is no data at the tract level for iteration code `050`, which is `Native Hawaiian and other Pacific Islander alone (500-599)`. This lack of published data is due to the disclosure avoidance implemented for small counts. For the 2000 and 2010 census data products, cell counts below 100 people of a given race were not published [Zayatz et al., 2009].

Let us select several of the less granular codes.

```
R> my_iterations = sf2_2010_iterations %>%
+   filter(DESCRIPTION %in% c("White alone",
+       "Black or African American alone",
+       "Asian alone (400-499)"))
R> print(my_iterations)
# A tibble: 3 x 2
  CODE  DESCRIPTION
  <chr> <chr>
1 002   White alone
2 004   Black or African American alone
3 012   Asian alone (400-499)
```

Now we have narrowed down our query enough to begin processing the CSV files. We can use the `get_filename_patterns` function to identify which files belong to the 2010 SF2 data.

```
R> patterns = get_filename_patterns(sf)
R> print(patterns)
# A tibble: 2 x 2
  TYPE  PATTERN
  <chr> <chr>
```

```
1 geo   "(\\w){2}geo(\\d){4}.sf2"
2 data  "(\\w){2}(\\d){3}(\\d){2}(\\d){4}.sf2"
R> pattern = patterns %>% filter(TYPE == "data") %>% pull(PATTERN)
R> data_paths = list.files("my_files", full.names = TRUE, pattern = pattern)
R> head(data_paths)
[1] "my_files/dc001012010.sf2" "my_files/dc001022010.sf2"
[3] "my_files/dc001032010.sf2" "my_files/dc001042010.sf2"
[5] "my_files/dc001052010.sf2" "my_files/dc001062010.sf2"
```

There are 1430 CSV files in the dataset, but only a small subset are needed for our current study. Recall that each CSV file contains information for a specific state, iteration code, and segment. Use `interpret_data_filenames` to get a tibble with the fields.

```
R> dat_files = interpret_data_filenames(sf, data_paths)
R> print(dat_files, n = 5)
# A tibble: 1,430 x 6
  STATE ITERATION_CODE SEGMENT YEAR  SF_TYPE PATH
  <chr> <chr>          <chr>   <chr> <chr>   <chr>
1 dc    001            01      2010  sf2     my_files/dc001012010.sf2
2 dc    001            02      2010  sf2     my_files/dc001022010.sf2
3 dc    001            03      2010  sf2     my_files/dc001032010.sf2
4 dc    001            04      2010  sf2     my_files/dc001042010.sf2
5 dc    001            05      2010  sf2     my_files/dc001052010.sf2
# ... with 1,425 more rows
# i Use `print(n = ...)` to see more rows
```

Let us filter this list using our target segment and iterations.

```
R> segment = sf2_2010_fields %>%
+   filter(TABLE == 'HCT012') %>%
+   pull(SEGMENT) %>%
+   unique()
R> query_files = dat_files %>%
+   inner_join(my_iterations, by = c("ITERATION_CODE" = "CODE")) %>%
+   filter(SEGMENT == segment)
R> print(query_files)
# A tibble: 3 x 7
  STATE ITERATION_CODE SEGMENT YEAR  SF_TYPE PATH                     DESCRIPT~1
  <chr> <chr>          <chr>   <chr> <chr>   <chr>                    <chr>
1 dc    002            11      2010  sf2     my_files/dc002112010.sf2 White alo~
2 dc    004            11      2010  sf2     my_files/dc004112010.sf2 Black or ~
3 dc    012            11      2010  sf2     my_files/dc012112010.sf2 Asian alo~
# ... with abbreviated variable name 1: DESCRIPTION
```

There are three CSV files relevant to our query. Let us bind them together by rows.

```
col_names = sf2_2010_fields %>% filter(SEGMENT == segment) %>% pull(FIELD)
dat_csv = tibble(.rows = 0)  # Start with an empty tibble

for (i in 1:nrow(query_files)) {
    dat_i_csv = read_csv(file = query_files$PATH[i], col_names = col_names)
    dat_csv = dat_csv %>% rbind(dat_i_csv)
}
```

Join the CSV data to the geo data and select the relevant columns.

```
R> dat = dat_geo %>%
+   filter(SUMLEV == '140', GEOCOMP == '00') %>%
+   select(-CHARITER) %>%
+   inner_join(dat_csv, by = c("LOGRECNO" = "LOGRECNO")) %>%
+   select(CHARITER, TRACT, starts_with("HCT012"))
R> print(dat, n = 5)
# A tibble: 372 x 15
  CHARITER TRACT HCT01~1 HCT01~2 HCT01~3 HCT01~4 HCT01~5 HCT01~6 HCT01~7 HCT01~8
  <chr>    <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
1 002      0001~    2476    1360     224      87      36     101    1136    1116
2 004      0001~      56      20       4       1       0       3      16      36
3 012      0001~      91      38       4       1       0       3      34      53
4 002      0002~       2       1       0       0       0       0       1       1
5 004      0002~       0       0       0       0       0       0       0       0
# ... with 367 more rows, 5 more variables: HCT0120009 <dbl>, HCT0120010 <dbl>,
#   HCT0120011 <dbl>, HCT0120012 <dbl>, HCT0120013 <dbl>, and abbreviated
#   variable names 1: HCT0120001, 2: HCT0120002, 3: HCT0120003, 4: HCT0120004,
#   5: HCT0120005, 6: HCT0120006, 7: HCT0120007, 8: HCT0120008
# i Use `print(n = ...)` to see more rows, and `colnames()` to see all variable names
```

Now, with the data in a usable form, we continue our analysis.

4.2 A Two-Way Contingency Table

First let us consider a two-way contingency table comparing several of the HCT012 factors against iteration codes. To do this, we will melt the table into a narrow format using the reshape2 package [Wickham, 2007] and then produce a table using xtabs.

```
R> library(reshape2)
R> dat_melt = melt(dat, id = c("CHARITER", "TRACT"),
+   variable.name = "FIELD", value.name = "COUNT") %>%
+   filter(FIELD %in% c('HCT0120003', 'HCT0120007', 'HCT0120009', 'HCT0120013'))
R> xtabs(COUNT ~ FIELD + CHARITER, data = dat_melt, drop.unused.levels = TRUE)
            CHARITER
FIELD          002   004   012
  HCT0120003 10230 11192   368
  HCT0120007 44383 38111  2299
  HCT0120009  3529 24192   496
  HCT0120013 56714 53596  5269
```

With more work (not shown), we can replace the codes with human-readable labels and improve table formatting to obtain Table 4. Several interesting observations can be seen from this table. First, both White alone and Black or African American alone households which are owner-occupied have roughly the same proportion with children. However, for renter-occupied households, the proportion of Black or African American alone with children is noticeably larger, while the proportion of White alone with children is much smaller. The proportion of Asian alone households with children is smaller for both owner and renter-occupied cases, but a bit larger for owner-occupied. This analysis could be continued, checking if this empirical result varies by geography.

4.3 Plot by Tract

Let us summarize Table 4 for each characteristic iteration and tract using an odds ratio approach [e.g., Agresti, 2012]. Let X_{ij} be indicators which take on value 1 for renter-occupied and 0 for owner-occupied for tract i and iteration code j. Similarly, let Y_{ij} be indicators for "with children". Consider an adjusted

Table 4: Two-way contingency table with using HCT012 and several characteristic iterations.

	Owner-occupied		Renter-occupied	
	No Children	With Children	No Children	With Children
Asian alone	2299	368	5269	496
Black or African American alone	38111	11192	53596	24192
White alone	44383	10230	56714	3529

log-odds ratio

$$\log\left\{1 + \frac{\mathrm{P}(X_{ij} = 1 \mid Y_{ij} = 1)}{\mathrm{P}(X_{ij} = 0 \mid Y_{ij} = 1)} \Big/ \frac{\mathrm{P}(X_{ij} = 1 \mid Y_{ij} = 0)}{\mathrm{P}(X_{ij} = 0 \mid Y_{ij} = 0)}\right\} \tag{1}$$

using empirical probabilities

$$\mathrm{P}(X_{ij} = 1 \mid Y_{ij} = 1) = \frac{\#\text{ renter-occupied and with children}}{\#\text{ with children}},$$
$$\mathrm{P}(X_{ij} = 1 \mid Y_{ij} = 0) = \frac{\#\text{ renter-occupied and without children}}{\#\text{ without children}},$$

and where one is added to the odds-ratio to accomodate zeros. We can compute (1) using tidyverse operations and the `dat` dataset which is in wide format.

```
R> dat_or = dat %>%
+   filter(HCT0120009 + HCT0120013 > 0, HCT0120003 + HCT0120007 > 0) %>%
+   mutate(PROB11 = HCT0120009 / (HCT0120009 + HCT0120013)) %>%
+   mutate(PROB10 = HCT0120003 / (HCT0120003 + HCT0120007)) %>%
+   mutate(OR = PROB11 / (1 - PROB11) * (1 - PROB10) / PROB10) %>%
+   filter(!is.na(OR) & !is.infinite(OR)) %>%
+   mutate(LOG_OR = log1p(OR)) %>%
+   select(CHARITER, TRACT, PROB11, PROB10, OR, LOG_OR)
```

To plot the data on a map of tracts, we obtain shapefiles for 2010 geography from the Census TIGER/Line website (https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html). At the time of this writing, the following sequence of commands will download and stage the set of files for use in operations that follow. The `curl` package [Ooms, 2021] is used to request data from the given URL.

```
R> library(curl)
R> dest_dir = "my_shapefiles"
R> dat_url = "https://www2.census.gov/geo/tiger/TIGER2010/TRACT/2010/tl_2010_11_tract10.zip"
R> dir.create(dest_dir)
R> local_zip_file = basename(dat_url)
R> curl_download(url = dat_url, destfile = local_zip_file)
R> unzip(local_zip_file, exdir = dest_dir)
```

To load and manipulate the downloaded shapefiles into R, we use the `sf` package [Pebesma, 2018]. We will now make use of the `ggplot2` package [Wickham, 2016] to plot the shape data.

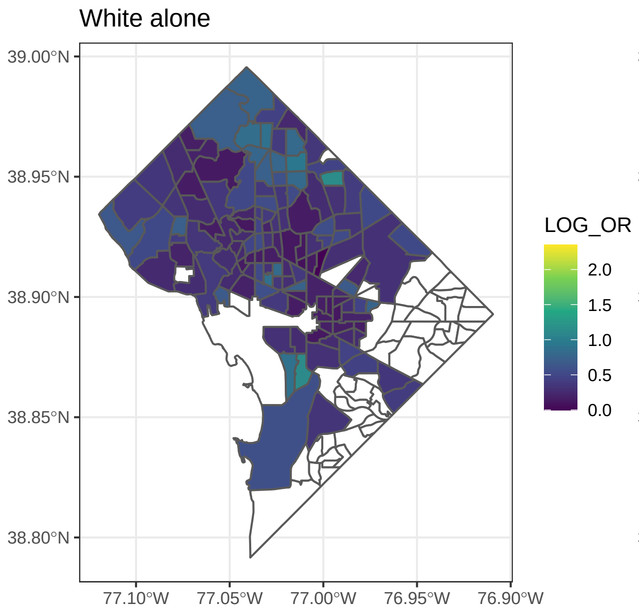
```
R> library(sf)
R> library(ggplot2)
R> dat_shp = st_read("my_shapefiles/tl_2010_11_tract10.shp", quiet = TRUE)
```

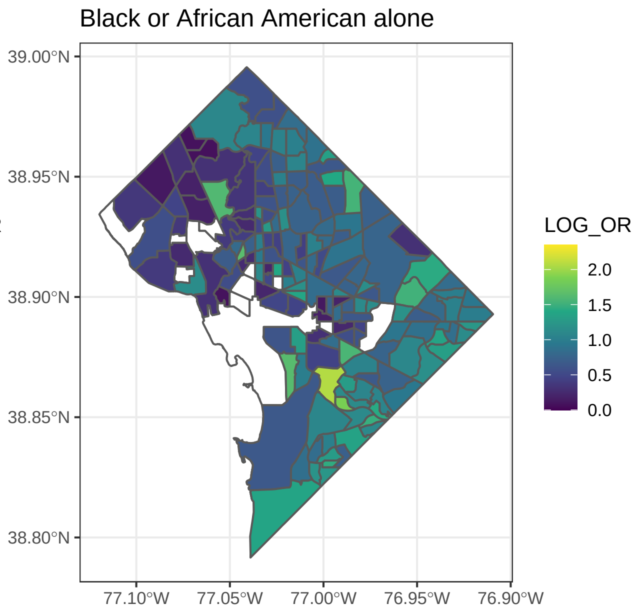
We now join the shape data to the odds-ratio data. We make use of `left_join` operations to ensure that records with no data are present; we would like these to appear as empty areas in the plot. We create one dataset per characteristic iteration.

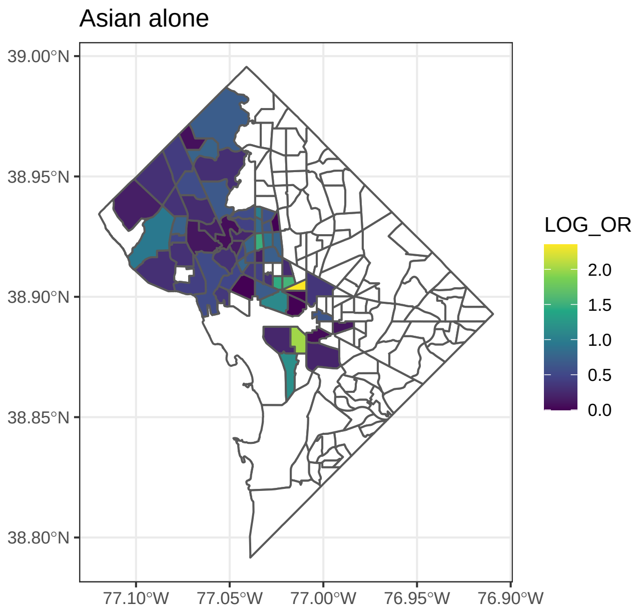
```
dat1_plot = dat_shp %>%
    left_join(dat_or %>% filter(CHARITER == '002'),
        by = c("TRACTCE10" = "TRACT"))
dat2_plot = dat_shp %>%
    left_join(dat_or %>% filter(CHARITER == '004'),
        by = c("TRACTCE10" = "TRACT"))
dat3_plot = dat_shp %>%
    left_join(dat_or %>% filter(CHARITER == '012'),
        by = c("TRACTCE10" = "TRACT"))
```

Finally, we plot each of the joined datasets. We ensure that they are plotted on the same color scale to aid with comparison.

```
my_limits = range(dat_or$LOG_OR)

ggplot(dat1_plot) +
    geom_sf(aes(fill = LOG_OR)) +
    ggtitle("White alone") +
    scale_fill_continuous(type = "viridis", na.value = "white", limits = my_limits) +
    theme_bw()

ggplot(dat2_plot) +
    geom_sf(aes(fill = LOG_OR)) +
    ggtitle("Black or African American alone") +
    scale_fill_continuous(type = "viridis", na.value = "white", limits = my_limits) +
    theme_bw()

ggplot(dat3_plot) +
    geom_sf(aes(fill = LOG_OR)) +
    ggtitle("Asian alone") +
    scale_fill_continuous(type = "viridis", na.value = "white", limits = my_limits) +
    theme_bw()
```







We notice several interesting features in the three plots. The `LOG_OR` could not be computed on all tracts, especially for the Asian alone iteration. Recall that a larger value of `LOG_OR` reflects that the odds of a household being renter occupied are increased when there are children present. The value of `LOG_OR` tends to be low for White alone, but is higher in some specific tracts such as those in northern DC. The value for Black or African American alone is lower in the tracts in west DC. The Asian alone iteration has one particular tract in the center of DC with a tiny area and a very large value. The `LOG_OR` value is generally low in west DC for all three iterations.

Our preliminary analysis has suggested some patterns in the data relating race, ownership versus rentership, and the presence of children in a household. These could be pursued further using additional data sources such as housing costs and employment, along with an understanding of the interplay between these socio-economic factors.

5 Conclusions

The package and methods shown may be useful to researchers and other data users who must work in a secure computing environment or who otherwise wish to process the data without network connectivity. We demonstrated how the contents of summary file SF2 from the 2010 Decennial Census can be navigated in R with the aid of tools in the `sfreader` package. With a basic understanding of the file and tabulation structure, several helper tables and utility functions facilitate data wrangling and plotting with standard packages such as the `tidyverse` and `ggplot2`. The 2010 SF2 summary file is among many summary files which have been released by the Census Bureau for public use; support for others may be considered in future versions of `sfreader`.

Acknowledgements

We thank Liza Hill (Decennial Information Technology Division, U.S. Census Bureau) and Joseph Kang, Eric Slud, and Tommy Wright (Center for Statistical Research & Methodology, U.S. Census Bureau) for reviewing the manuscript and for their encouragement.

References

Alan Agresti. Categorical Data Analysis. Wiley, 3rd edition, 2012.

Guanglai Li. totalcensus: Extract Decennial Census and American Community Survey Data, 2021. URL https://CRAN.R-project.org/package=totalcensus. R package version 0.6.6.

Cory McCartan and Christopher T. Kenny. PL94171: Tabulate P.L. 94-171 Redistricting Data Summary Files, 2021. URL https://CRAN.R-project.org/package=PL94171. R package version 1.0.1.

Jeroen Ooms. curl: A Modern and Flexible Web Client for R, 2021. URL https://CRAN.R-project.org/package=curl. R package version 4.3.2.

Edzer Pebesma. Simple Features for R: Standardized Support for Spatial Vector Data. The R Journal, 10(1): 439–446, 2018. doi: 10.32614/RJ-2018-009. URL https://doi.org/10.32614/RJ-2018-009.

R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2022. URL https://www.R-project.org/.

U.S. Census Bureau. 2010 Census Summary File 2 - Technical Documentation, 2011. URL https://www2.census.gov/programs-surveys/decennial/2010/technical-documentation/complete-tech-docs/summary-file/sf2.pdf.

Kyle Walker and Matt Herman. tidycensus: Load US Census Boundary and Attribute Data as 'tidyverse' and 'sf'-Ready Data Frames, 2022. URL https://walker-data.com/tidycensus/. R package version 1.1.2.9000.

Hadley Wickham. Reshaping data with the reshape package. Journal of Statistical Software, 21(12):1–20, 2007. URL http://www.jstatsoft.org/v21/i12/.

Hadley Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.

Hadley Wickham. Advanced R. Chapman and Hall/CRC, 2nd edition, 2019.

Hadley Wickham, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, Alex Hayes, Lionel Henry, Jim Hester, Max Kuhn, Thomas Lin Pedersen, Evan Miller, Stephan Milton Bache, Kirill Müller, Jeroen Ooms, David Robinson, Dana Paige Seidel, Vitalie Spinu, Kohske Takahashi, Davis Vaughan, Claus Wilke, Kara Woo, and Hiroaki Yutani. Welcome to the tidyverse. Journal of Open Source Software, 4(43):1686, 2019. doi: https://doi.org/10.21105/joss.01686.

Laura Zayatz, Jason Lucero, Paul Massell, and Asoka Ramanayake. Disclosure avoidance for census 2010 and american community survey five-year tabular data products. Technical report, U.S. Census Bureau, November 2009. URL https://www.census.gov/library/working-papers/2009/adrm/rrs2009-10.html.