Code-Along 01

Packages

We'll use the following packages:

- `here()` (relative file paths)
- `tidyverse()` (data wrangling)
- `gssr()` (U.S. General Social Survey data)
- `gssrdoc()` (GSS documentation)

Install `here()` and `tidyverse()`

Let's first install the two packages that are available on CRAN.

Copy and paste the following code into your Console pane. Then hit enter.

```
install.packages("here")
```

Then, do the same to install the tidyverse package.

```
install.packages("tidyverse")
```

Install `gssr()` and `gssrdoc()`

```
# Install 'gssr' from 'ropensci' universe
install.packages('gssr', repos =
  c('https://kjhealy.r-universe.dev', 'https://cloud.r-project.org'))

# Also recommended: install 'gssrdoc' as well
install.packages('gssrdoc', repos =
  c('https://kjhealy.r-universe.dev', 'https://cloud.r-project.org'))
```

Load the packages

```
library(here)
```

```
here() starts at C:/Users/Joanna/Documents/GitHub/Stats for Sociologists
```

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.4
```

```
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(gssr)
```

```
Package loaded. To attach the GSS data, type data(gss_all) at the console.
For the panel data and documentation, type e.g. data(gss_panel08_long) and data(gss_panel_do
For help on a specific GSS variable, type ?varname at the console.
```

```
library(gssrdoc)
```

Environment

```
# software documentation
sessionInfo()
```

```
R version 4.5.1 (2025-06-13 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_Canada.utf8  LC_CTYPE=English_Canada.utf8
[3] LC_MONETARY=English_Canada.utf8 LC_NUMERIC=C
[5] LC_TIME=English_Canada.utf8

time zone: America/Toronto
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] gssrdoc_0.7.0   gssr_0.7        lubridate_1.9.4 forcats_1.0.0
 [5] stringr_1.5.1   dplyr_1.1.4     purrr_1.0.4     readr_2.1.5
 [9] tidyr_1.3.1     tibble_3.3.0    ggplot2_3.5.2   tidyverse_2.0.0
[13] here_1.0.1

loaded via a namespace (and not attached):
 [1] gtable_0.3.6       jsonlite_2.0.0     compiler_4.5.1     tidyselect_1.2.1
 [5] scales_1.4.0       yaml_2.3.10        fastmap_1.2.0      R6_2.6.1
 [9] generics_0.1.4     curl_6.3.0         knitr_1.50         rprojroot_2.1.0
[13] pillar_1.11.0      RColorBrewer_1.1-3 tzdb_0.5.0         rlang_1.1.6
[17] stringi_1.8.7      xfun_0.52          fs_1.6.6           timechange_0.3.0
[21] cli_3.6.5          withr_3.0.2        magrittr_2.0.3     digest_0.6.37
[25] grid_4.5.1         rstudioapi_0.17.1  haven_2.5.5        hms_1.1.3
[29] lifecycle_1.0.4    vctrs_0.6.5        evaluate_1.0.4     glue_1.8.0
[33] farver_2.1.2       rmarkdown_2.29     tools_4.5.1        pkgconfig_2.0.3
[37] htmltools_0.5.8.1
```

Project structure

Let's set up your project structure using the `here()` package.

First, let's establish our project directory.

```
# set the file path to the root of the project
here()
```

```
[1] "C:/Users/Joanna/Documents/GitHub/Stats for Sociologists"
```

Next, we'll create folders within our project using `here()` and `dir.create()`

```
# Create base folders
dir.create(here("data"), recursive = TRUE)
```

```
Warning in dir.create(here("data"), recursive = TRUE):
'C:\Users\Joanna\Documents\GitHub\Stats for Sociologists\data' already exists
```

```
dir.create(here("code-alongs"), recursive = TRUE)
```

```
Warning in dir.create(here("code-alongs"), recursive = TRUE):
'C:\Users\Joanna\Documents\GitHub\Stats for Sociologists\code-alongs' already
exists
```

```
dir.create(here("milestones"), recursive = TRUE)
```

```
Warning in dir.create(here("milestones"), recursive = TRUE):
'C:\Users\Joanna\Documents\GitHub\Stats for Sociologists\milestones' already
exists
```

```
dir.create(here("project"), recursive = TRUE)
```

```
Warning in dir.create(here("project"), recursive = TRUE):
'C:\Users\Joanna\Documents\GitHub\Stats for Sociologists\project' already
exists
```

Now, we'll create sub-folders using `here()` and `dir.create()`

```
# Create project sub-folders
dir.create(here("project", "data"), recursive = TRUE)
```

```
Warning in dir.create(here("project", "data"), recursive = TRUE):
'C:\Users\Joanna\Documents\GitHub\Stats for Sociologists\project\data' already
exists
```

```
dir.create(here("project", "scripts"), recursive = TRUE)
```

```
Warning in dir.create(here("project", "scripts"), recursive = TRUE):
'C:\Users\Joanna\Documents\GitHub\Stats for Sociologists\project\scripts'
already exists
```

```
dir.create(here("project", "outputs"), recursive = TRUE)
```

```
Warning in dir.create(here("project", "outputs"), recursive = TRUE):
'C:\Users\Joanna\Documents\GitHub\Stats for Sociologists\project\outputs'
already exists
```

Check your work by reporting a list of folders and or files in the R-project folders and sub-folder.

```
# Your SOC6302 class folder
list.files(path = here())
```

```
 [1] "_extensions"                 "_quarto-speaker.yml"
 [3] "_quarto.yml"                 "code-alongs"
 [5] "data"                        "docs"
 [7] "lectures"                    "milestones"
 [9] "project"                     "slides"
[11] "SOC6302_readings.qmd"        "SOC6302_syllabus.qmd"
[13] "Stats for Sociologists.Rproj" "tutorials"
```

```
# Your "Project" sub-folder
list.files(path = here("project"))
```

```
[1] "data"    "outputs" "scripts"
```

Save code-along

Save this code-along in your newly created "code-along" sub-folder.

There's no command in the R console to save scripts or Quarto files — you use the editor's File > Save As or Ctrl+S.

Meet your data

We're going to use data from the U.S. General Social Survey (GSS).

Let's load your data.

```
# Load the data (will appear in your Global Environment pane)
data(gss_all)

# Preview the datatable which is automatically named gss_all
gss_all
```

```
# A tibble: 75,699 x 6,867
   year         id wrkstat    hrs1        hrs2        evwork      occ   prestige
   <dbl+lbl> <dbl> <dbl+lbl>  <dbl+lbl>   <dbl+lbl>   <dbl+lbl>   <dbl> <dbl+lb>
 1 1972          1 1 [workin~ NA(i) [iap] NA(i) [iap] NA(i) [iap] 205   50
 2 1972          2 5 [retire~ NA(i) [iap] NA(i) [iap]     1 [yes] 441   45
 3 1972          3 2 [workin~ NA(i) [iap] NA(i) [iap] NA(i) [iap] 270   44
 4 1972          4 1 [workin~ NA(i) [iap] NA(i) [iap] NA(i) [iap]   1   57
 5 1972          5 7 [keepin~ NA(i) [iap] NA(i) [iap]     1 [yes] 385   40
 6 1972          6 1 [workin~ NA(i) [iap] NA(i) [iap] NA(i) [iap] 281   49
 7 1972          7 1 [workin~ NA(i) [iap] NA(i) [iap] NA(i) [iap] 522   41
 8 1972          8 1 [workin~ NA(i) [iap] NA(i) [iap] NA(i) [iap] 314   36
 9 1972          9 2 [workin~ NA(i) [iap] NA(i) [iap] NA(i) [iap] 912   26
10 1972         10 1 [workin~ NA(i) [iap] NA(i) [iap] NA(i) [iap] 984   18
# i 75,689 more rows
# i 6,859 more variables: wrkslf <dbl+lbl>, wrkgovt <dbl+lbl>,
#   commute <dbl+lbl>, industry <dbl+lbl>, occ80 <dbl+lbl>, prestg80 <dbl+lbl>,
#   indus80 <dbl+lbl>, indus07 <dbl+lbl>, occonet <dbl+lbl>, found <dbl+lbl>,
#   occ10 <dbl+lbl>, occindv <dbl+lbl>, occstatus <dbl+lbl>, occtag <dbl+lbl>,
#   prestg10 <dbl+lbl>, prestg105plus <dbl+lbl>, indus10 <dbl+lbl>,
#   indstatus <dbl+lbl>, indtag <dbl+lbl>, marital <dbl+lbl>, ...
```

A "tibble" is another name for "tidy dataset," meaning that the data is organized in structured, clear rows and columns. "(75,699 × 6,867)" means the dataset contains 75,699 rows and 6,867 columns. Commonly, in social sciences, rows are referred to as "observations" and columns as "variables." In our case, there are 75,699 observations (e.g., respondents) and 6,867 variables.

You can also load the GSS data for a specific survey year.

```
# Get the data only for the 2024 survey respondents
gss24 <- gss_get_yr(2024)
```

```
Fetching: https://gss.norc.org/documents/stata/2024_stata.zip
```

```
# look at the first 6 rows of the dataframe
head(gss24)
```

```
# A tibble: 6 x 639
  year        id wrkstat hrs1        hrs2        evwork      marital martype
  <dbl+lb> <dbl> <dbl+l> <dbl+lbl>   <dbl+lbl>   <dbl+lbl>   <dbl+l> <dbl+lbl>
1 2024         1 1 [wor~    43       NA(i) [iap] NA(i) [iap] 5 [nev~ NA(i) [iap]
2 2024         2 5 [ret~ NA(i) [iap] NA(i) [iap]     1 [yes] 5 [nev~ NA(i) [iap]
3 2024         3 5 [ret~ NA(i) [iap] NA(i) [iap]     1 [yes] 1 [mar~     1 [mar~
4 2024         4 2 [wor~    20       NA(i) [iap] NA(i) [iap] 5 [nev~ NA(i) [iap]
5 2024         5 5 [ret~ NA(i) [iap] NA(i) [iap]     1 [yes] 3 [div~ NA(i) [iap]
6 2024         6 4 [une~ NA(i) [iap] NA(i) [iap] NA(i) [iap] 1 [mar~     1 [mar~
# i 631 more variables: divorce <dbl+lbl>, widowed <dbl+lbl>,
#   spwrksta <dbl+lbl>, sphrs1 <dbl+lbl>, sphrs2 <dbl+lbl>, spevwork <dbl+lbl>,
#   cowrksta <dbl+lbl>, coevwork <dbl+lbl>, cohrs1 <dbl+lbl>, cohrs2 <dbl+lbl>,
#   sibs <dbl+lbl>, childs <dbl+lbl>, age <dbl+lbl>, educ <dbl+lbl>,
#   speduc <dbl+lbl>, coeduc <dbl+lbl>, codeg <dbl+lbl>, degree <dbl+lbl>,
#   padeg <dbl+lbl>, madeg <dbl+lbl>, spdeg <dbl+lbl>, sex <dbl+lbl>,
#   race <dbl+lbl>, res16 <dbl+lbl>, reg16 <dbl+lbl>, mobile16 <dbl+lbl>, ...
```

With your mouse, go to the environment panel (upper-right) and click on the "gss24" object. It pops up and you can browse through it.

This is often a good idea to get a first feel for the data, but only if your dataset is relatively small.

The GSS documentation is available online in .pdf form.

The .pdfs will be useful for general overviews.

For specific variable information, it will be helpful to use the documentation you'll load into RStudio.

```
# Load the codebook
data(gss_dict)
```

To see the variables available in the dataset, use the `names()` command. This command is best to use with smaller datasets.

```
names(gss_all)
```

Variables

For information about a specific GSS variable, type `?varname` at the console.

In the output pane, the Help tab will show the variable documentation.

Example: `?meovrwrk`

```
?meovrwrk
```

Notes:

Variable name: meovrwrk
Variable label: Men hurt family when focus on work too much
1994 was the first year of the survey.
695 respondents agreed with the statement.
iap – missing. Values: the numeric and response category key (1 = strongly agree)

We often want to know which years a question was asked.

We can find this out for one or more variables with `gss_which_years()`.

```
gss_which_years(gss_all, meovrwrk)
```

```
# A tibble: 35 x 2
   year      meovrwrk
   <dbl+lbl> <lgl>
 1 1972      FALSE
 2 1973      FALSE
 3 1974      FALSE
 4 1975      FALSE
 5 1976      FALSE
 6 1977      FALSE
 7 1978      FALSE
 8 1980      FALSE
 9 1982      FALSE
10 1983      FALSE
# i 25 more rows
```

If run in the console, to see all rows, wrap the code in the `print()` command:

```
print(gss_which_years(gss_all, meovrwrk), n = 40)
```

You can access the variables (i.e., columns) using the `$` operator, as shown using the table() function

(NOTE: The variable names are case sensitive. In this dataset, all variables are lowercase.)

```
table(gss_all$meovrwrk)
```

```
   1     2     3     4     5
2436 10813  4797  5806   927
```

2436 respondents were coded as 1 on this variable. What does that mean? (Look at the help page.)

Let's look at only the 2024 respondents. Change the code to show just the `gss24` respondents. Then, add text to your code-along that interprets the results for the `2` value.

```
# Replace gss_all with gss24 and then run the code.
table(gss_all$meovrwrk)
```

```
   1     2     3     4     5
2436 10813  4797  5806   927
```

Quarto: Render

Finally, let's `render` your code-along-01 and view the results.