Code-along 04

FirstName LastName

Setup

Packages

Load the standard packages.

```
library(here)
library(tidyverse)
library(haven) # not core tidyverse
library(gssr)
library(gssrdoc)
library(summarytools)
```

GSS Panel Data: Download

https://gss.norc.org/get-the-data/stata

Download GSS 2016-2020 Panel (Release 1a, April 2022)

Save the **unzipped** file in your class `data` folder.

GSS Panel Data: Load

```
# Use here() to construct the file path
gss_panel.dta <- here("data", "GSS_2020_panel_stata_1a/gss2020panel_r1a.dta")

#load the data using `haven::read_dta()`
data <- read_dta(gss_panel.dta)
```

```
# Or, do both at the same time!
# data <- read_dta(here("data", "GSS_2020_panel_stata_1a/gss2020panel_r1a.dta"))
```

GSS 2016-2020 Panel Dataset

```
set.seed(815)  # Ensures you get the same sample every time

data |>
  select(yearid, starts_with("year_"), starts_with("age_")) |>
  slice_sample(n = 10)
```

```
# A tibble: 10 x 7
     yearid year_1a year_1b year_2 age_1a    age_1b    age_2
      <dbl>   <dbl>   <dbl>  <dbl> <dbl+lbl> <dbl+lbl> <dbl+lbl>
 1 20182183      NA    2018     NA NA(i)        52     NA(i)
 2 20180711      NA    2018     NA NA(i)        19     NA(i)
 3 20182189      NA    2018   2020 NA(i)        37        39
 4 20160354    2016      NA     NA    56     NA(i)     NA(i)
 5 20180452      NA    2018   2020 NA(i)        29        31
 6 20181503      NA    2018   2020 NA(i)        58        60
 7 20162744    2016      NA   2020    71     NA(i)        75
 8 20160315    2016      NA   2020    69     NA(i)        73
 9 20160170    2016      NA     NA    75     NA(i)     NA(i)
10 20161888    2016      NA     NA    71     NA(i)     NA(i)
```

Manipulating Dataframes

Selection helpers

Match variables according to a given pattern.

- `starts_with()`: Starts with an exact prefix.
- `ends_with()`: Ends with an exact suffix.
- `contains()`: Contains a literal string.
- ...

```r
my_data <- data |>
  select(yearid, wtssnr_2,
         starts_with("age_"),
         starts_with("family16_"),
         starts_with("socfrend_"),
         starts_with("childs_"))
```

```r
# You can supply multiple prefixes or suffixes.
my_data <- data |>
  select(yearid, wtssnr_2,
         starts_with(c("age_", "family16_", "socfrend", "childs"))
         )

my_data <- as_factor(my_data) # Apply labels to data
```

head() & tail()

Look at the first few column names and **first** few rows.

```r
head(my_data, n = 5)
```

```
# A tibble: 5 x 14
    yearid wtssnr_2 age_1a age_1b age_2 family16_1a       family16_1b family16_2
     <dbl>    <dbl> <fct>  <fct>  <fct> <fct>             <fct>       <fct>
1 20160001    1.44  47     <NA>   51    both own mother ~ <NA>        not avail~
2 20160002    0.722 61     <NA>   65    both own mother ~ <NA>        not avail~
3 20160003   NA     72     <NA>   <NA>  both own mother ~ <NA>        iap
4 20160004    2.89  43     <NA>   47    mother only       <NA>        not avail~
5 20160005   NA     55     <NA>   <NA>  both own mother ~ <NA>        iap
# i 6 more variables: socfrend_1a <fct>, socfrend_1b <fct>, socfrend_2 <fct>,
#   childs_1a <fct>, childs_1b <fct>, childs_2 <fct>
```

Look at the first few column names and **last** few rows.

```r
tail(my_data, n = 5)
```

```
# A tibble: 5 x 14
    yearid wtssnr_2 age_1a age_1b age_2 family16_1a family16_1b       family16_2
     <dbl>    <dbl> <fct>  <fct>  <fct> <fct>       <fct>             <fct>
```

```
1 20182344   NA     <NA>   37     <NA>  <NA>        mother and stepf~ iap
2 20182345    0.995 <NA>   75     77    <NA>        both own mother ~ not avail~
3 20182346    0.995 <NA>   67     70    <NA>        both own mother ~ not avail~
4 20182347   NA     <NA>   72     <NA>  <NA>        both own mother ~ iap
5 20182348   NA     <NA>   79     <NA>  <NA>        both own mother ~ iap
# i 6 more variables: socfrend_1a <fct>, socfrend_1b <fct>, socfrend_2 <fct>,
#   childs_1a <fct>, childs_1b <fct>, childs_2 <fct>
```

Tidy data

This data is tidy!
Each variable in its own column, and each observation in its own row.

```
my_data |>
  pivot_longer(
    cols = c(-yearid, -wtssnr_2),
    names_to = c("variable"),
    values_to = "value") |>
    separate_wider_delim(variable,
                         delim = "_",
                         names = c("variable", "panel")) |>
  pivot_wider(
    names_from = variable,
    values_from = value)
```

```
# A tibble: 15,645 x 7
     yearid wtssnr_2 panel age   family16                    socfrend     childs
      <dbl>    <dbl> <chr> <fct> <fct>                       <fct>        <fct>
 1 20160001    1.44  1a    47    both own mother and father  several tim~ 3
 2 20160001    1.44  1b    <NA>  <NA>                        <NA>         <NA>
 3 20160001    1.44  2     51    not available for this year several tim~ 3
 4 20160002    0.722 1a    61    both own mother and father  several tim~ 0
 5 20160002    0.722 1b    <NA>  <NA>                        <NA>         <NA>
 6 20160002    0.722 2     65    not available for this year about once ~ 0
 7 20160003   NA     1a    72    both own mother and father  <NA>         2
 8 20160003   NA     1b    <NA>  <NA>                        <NA>         <NA>
 9 20160003   NA     2     <NA>  iap                         <NA>         <NA>
10 20160004    2.89  1a    43    mother only                 once or twi~ 4
# i 15,635 more rows
```

pivot_longer()

```
my_data_long <- my_data |>
  pivot_longer(
    cols = 3:14,
    names_to = "variable",
    values_to = "value")

head(my_data_long, n = 5)
```

```
# A tibble: 5 x 4
    yearid wtssnr_2 variable    value
     <dbl>    <dbl> <chr>       <fct>
1 20160001     1.44 age_1a      47
2 20160001     1.44 age_1b      <NA>
3 20160001     1.44 age_2       51
4 20160001     1.44 family16_1a both own mother and father
5 20160001     1.44 family16_1b <NA>
```

separate()

```
my_data_long <- my_data |>
  pivot_longer(
    cols = c(-yearid, -wtssnr_2),
    names_to = "variable",
    values_to = "value") |>
  separate_wider_delim(variable,
                       delim = "_",
                       names = c("variable", "panel"))

head(my_data_long, n = 5)
```

```
# A tibble: 5 x 5
    yearid wtssnr_2 variable panel value
     <dbl>    <dbl> <chr>    <chr> <fct>
1 20160001     1.44 age      1a    47
2 20160001     1.44 age      1b    <NA>
3 20160001     1.44 age      2     51
4 20160001     1.44 family16 1a    both own mother and father
5 20160001     1.44 family16 1b    <NA>
```

pivot_wider()

```
my_data <- my_data |> # overwriting my_data
  pivot_longer(
    cols = c(-yearid, -wtssnr_2),
    names_to = "variable",
    values_to = "value") |>
    separate_wider_delim(variable,
                         delim = "_",
                         names = c("variable", "panel")) |>
  pivot_wider(
    names_from = variable,
    values_from = value)

head(my_data, n = 5)
```

```
# A tibble: 5 x 7
    yearid wtssnr_2 panel age   family16                    socfrend      childs
     <dbl>    <dbl> <chr> <fct> <fct>                       <fct>         <fct>
1 20160001    1.44  1a    47    both own mother and father  several time~ 3
2 20160001    1.44  1b    <NA>  <NA>                        <NA>          <NA>
3 20160001    1.44  2     51    not available for this year several time~ 3
4 20160002    0.722 1a    61    both own mother and father  several time~ 0
5 20160002    0.722 1b    <NA>  <NA>                        <NA>          <NA>
```

Recode the reshaped variable

```
my_data <- my_data |>
  mutate(panel = case_when(
         panel == "1a" ~ 2016,
         panel == "1b" ~ 2018,
         panel == "2" ~ 2020,
         TRUE ~ NA_integer_))

head(my_data, n = 3)
```

```
# A tibble: 3 x 7
    yearid wtssnr_2 panel age   family16                    socfrend      childs
     <dbl>    <dbl> <dbl> <fct> <fct>                       <fct>         <fct>
```

```
1 20160001     1.44  2016 47    both own mother and father  several time~ 3
2 20160001     1.44  2018 <NA>  <NA>                        <NA>          <NA>
3 20160001     1.44  2020 51    not available for this year several time~ 3
```

relocate()

```
my_data <- my_data |>
  relocate(panel)

head(my_data, n = 2)
```

```
# A tibble: 2 x 7
  panel   yearid wtssnr_2 age   family16                   socfrend       childs
  <dbl>    <dbl>    <dbl> <fct> <fct>                      <fct>          <fct>
1  2016 20160001     1.44 47    both own mother and father several times~ 3
2  2018 20160001     1.44 <NA>  <NA>                       <NA>           <NA>
```

```
my_data <- my_data |>
  relocate(panel, .after = yearid)

head(my_data, n = 2)
```

```
# A tibble: 2 x 7
    yearid panel wtssnr_2 age   family16                   socfrend       childs
     <dbl> <dbl>    <dbl> <fct> <fct>                      <fct>          <fct>
1 20160001  2016     1.44 47    both own mother and father several times~ 3
2 20160001  2018     1.44 <NA>  <NA>                       <NA>           <NA>
```

arrange()

```
my_data |>
  arrange(panel) |>
  select(yearid, panel, age, family16)
```

```
# A tibble: 15,645 x 4
     yearid panel age   family16
      <dbl> <dbl> <fct> <fct>
```

```
 1 20160001  2016 47    both own mother and father
 2 20160002  2016 61    both own mother and father
 3 20160003  2016 72    both own mother and father
 4 20160004  2016 43    mother only
 5 20160005  2016 55    both own mother and father
 6 20160006  2016 53    other
 7 20160007  2016 50    both own mother and father
 8 20160008  2016 23    both own mother and father
 9 20160009  2016 45    both own mother and father
10 20160010  2016 71    both own mother and father
# i 15,635 more rows
```

```
my_data |>
  arrange(desc(panel)) |>
  select(yearid, panel, age, family16)
```

```
# A tibble: 15,645 x 4
     yearid panel age   family16
      <dbl> <dbl> <fct> <fct>
 1 20160001  2020 51    not available for this year
 2 20160002  2020 65    not available for this year
 3 20160003  2020 <NA>  iap
 4 20160004  2020 47    not available for this year
 5 20160005  2020 <NA>  iap
 6 20160006  2020 <NA>  iap
 7 20160007  2020 <NA>  iap
 8 20160008  2020 27    not available for this year
 9 20160009  2020 49    not available for this year
10 20160010  2020 <NA>  iap
# i 15,635 more rows
```

Joining Dataframes

Example datasets

dataframe 1

```
  coupleid  name age
1        2  John  42
2        1 Megan  36
3        3   Bin  38
```

dataframe 2

```
  coupleid    name age
1        1     Sue  40
2        3  Ye-jin  39
3        2 Chrissy  35
```

dataframe 3

```
  coupleid marstat numchild country
1        3       1        1 S.Korea
2        1       0        0      US
3        2       1        4      US
```

append data with `bind_rows()`

```
df_all <- bind_rows(df_partner1, df_partner2)

tibble(df_all)
```

```
# A tibble: 6 x 3
  coupleid name      age
     <dbl> <chr>   <dbl>
1        2 John       42
2        1 Megan      36
3        3 Bin        38
4        1 Sue        40
5        3 Ye-jin     39
6        2 Chrissy    35
```

merge data with `left_join()`

```
df_couples <- left_join(df_partner1, df_family, by = "coupleid")

tibble(df_couples)
```

```
# A tibble: 3 x 6
  coupleid name    age marstat numchild country
     <dbl> <chr> <dbl>   <dbl>    <dbl> <chr>
1        2 John     42       1        4 US
2        1 Megan    36       0        0 US
3        3 Bin      38       1        1 S.Korea
```

Think Like a Statistician

Are married people above or below average in internet use or income? Does it vary by survey year?

Answering this research question takes a few steps. But, the first step is to create a dataframe with all the necessary information.

Step 1. Create a new, reshaped dataframe

- select the variables yearid, all marital variables, all wwwhr variables, and all realrinc variables
- pivot the marital, wwwhr, and realrinc variables longer (while keeping yearid)
- separate the new variable that contains the variable names and panel id into two variables
- pivot wider the data so marital, wwwhr, and realrinc are all their own variables (columns)
- recode your panel variable so 1a = 2016, 1b = 2018, and 2 = 2020

Step 2. Create a summary dataframe

- remove rows with missing `wwwhr` or `realrinc` values
- group the data by your `panel` variable
- create a summary dataframe that contains the averages for wwwhr and realrinc

Step 3. Put the two dataframes together

- Use full_join to put your two new dataframes together

```
# TIP: It's often easier to play with your code in an R script first.
# Then, copy and past your working R code into this code-chunk
# and delete the `eval` statement when your code is fully working

# Reshape data
think_data <- data |>
    select() |>
    pivot_longer() |>
    separate_wider_delim() |>
  pivot_wider()
```

```
# recode variables
think_data <- think_data |>
  mutate()

## summarize by panel
think_summary <- think_data |>
  drop_na()  |>
  group_by() |>
  summarise()

think_full <- full_join()

think_full
```

Your Data Take

What's your conclusion to our initial research question?

NOTE: after you've created a ***`think_full`*** *dataframe with the appropriate variables, delete the* ***`echo`*** *and* ***`eval`*** *statements in each of the code blocks below to produce the necessary tables.*