DC for Social Science: Exercises for Day 2

Lachlan Deer

Before We Start

You will need to use the following libraries:

```
library(readr)
library(dplyr)
library(lubridate)
library(tidyr)
library(ggplot2)
```

And you will have needed to download the `SAFI` data.

The code below will download the cleaned SAFI data as a csv file into a subdirectory called `data` if you have not already downloaded the data, otherwise it will print a message saying the data doesn't need to be downloaded.

```
dest_file <- "data/SAFI_clean.csv"
if(!file.exists(dest_file)){
  dir.create('data')
  download.file("https://ndownloader.figshare.com/files/11492171", destfile =  dest_file, mode = "wb")
} else {
message ('data already downloaded')
}
```

Load the data that was downloaded as follows:

```
interviews <- read_csv('data/SAFI_clean.csv', na = "NULL")
```

Solutions: Will be made available at the end of the session here.

Part 1 - Working With Data

Exercise 1

- Create a data frame (`interviews_100`) containing only the data in row 100 of the `interviews` dataset.

solution

```
interviews_100 <- slice(interviews, 100)
```

- Create a new data frame (`interviews_last`) from that last row. (*Hint:* Refer to `?n()` for help)

solution

```
interviews_last <- slice(interviews, n())
```

- Extract the row that is in the middle of the dataset and store the content of this middle row in an object named `interviews_middle`. (*hint:* Use the `median()` function and what you've learned about `n()` to find the middle row!)

solution

```
interviews_middle <- slice(interviews, median(1:n()))
```

- Combine `n()` with `slice()` to reproduce the behavior of head(interviews), keeping just the first through 6th rows of the interviews dataset. Name the resulting dataset `interviews_head`.

solution

```
interviews_head <- slice(interviews, -(7:n()))
```

Exercise 2

- Using pipes, subset the interviews data to include interviews where respondents were members of an irrigation association (`memb_assoc`) and retain only the columns `affect_conflicts`, `liv_count`, and `no_meals`.

solution

```
interviews %>%
    filter(memb_assoc == "yes") %>%
    select(affect_conflicts, liv_count, no_meals)
```

```
## # A tibble: 33 x 3
##    affect_conflicts liv_count no_meals
##    <chr>                <dbl>    <dbl>
##  1 once                     3        2
##  2 never                    2        2
##  3 never                    2        3
##  4 once                     3        2
##  5 frequently               1        3
##  6 more_once                5        2
##  7 more_once                3        2
##  8 more_once                2        3
##  9 once                     3        3
## 10 never                    3        3
## # ... with 23 more rows
```

- Create a new dataframe from the interviews data that meets the following criteria: contains only the village column and a new column called total_meals containing a value that is equal to the total number of meals served in the household per day on average (no_membrs times no_meals). Only the rows where total_meals is greater than 20 should be shown in the final dataframe.

solution

```
interviews_total_meals <- interviews %>%
    mutate(total_meals = no_membrs * no_meals) %>%
    filter(total_meals > 20) %>%
    select(village, total_meals)
```

- Use group_by() and summarize() to find the mean, min, and max number of household members for each village. Also add the number of observations (hint: see ?n).

solution

```
interviews %>%
  group_by(village) %>%
  summarize(
      mean_no_membrs = mean(no_membrs),
      min_no_membrs = min(no_membrs),
      max_no_membrs = max(no_membrs),
      n = n()
  )
```

```
## # A tibble: 3 x 5
##   village  mean_no_membrs min_no_membrs max_no_membrs     n
##   <chr>             <dbl>         <dbl>         <dbl> <int>
## 1 Chirodzo           7.08             2            12    39
## 2 God                6.86             3            15    43
## 3 Ruaca              7.57             2            19    49
```

- What was the largest household interviewed in each month?

solution

```
interviews %>%
    mutate(month = month(interview_date),
           year = year(interview_date)) %>%
    group_by(year, month) %>%
    summarize(max_no_membrs = max(no_membrs))
```

```
## # A tibble: 5 x 3
## # Groups:   year [2]
##    year month max_no_membrs
##   <dbl> <dbl>         <dbl>
## 1  2016    11            19
## 2  2016    12            12
## 3  2017     4            17
## 4  2017     5            15
## 5  2017     6            15
```

Exercise 3

- Create a new dataframe (named interviews_months_lack_food) that has one column for each month and records `TRUE` or `FALSE` for whether each interview respondent was lacking food in that month.

solution

```
interviews_months_lack_food <- interviews %>%
  separate_rows(months_lack_food, sep = ";") %>%
  mutate(months_lack_food_logical  = TRUE) %>%
  pivot_wider(names_from = months_lack_food,
              values_from = months_lack_food_logical,
              values_fill = list(months_lack_food_logical = FALSE))
```

- How many months (on average) were respondents without food if they did belong to an irrigation association? What about if they didn't?

solution

```r
interviews_months_lack_food %>%
  mutate(number_months = rowSums(select(., Jan:May))) %>%
  group_by(memb_assoc) %>%
  summarize(mean_months = mean(number_months))
```

```
## # A tibble: 3 x 2
##   memb_assoc mean_months
##   <chr>            <dbl>
## 1 no                2   
## 2 yes               2.30
## 3 <NA>              2.82
```

Part 2 - Plotting

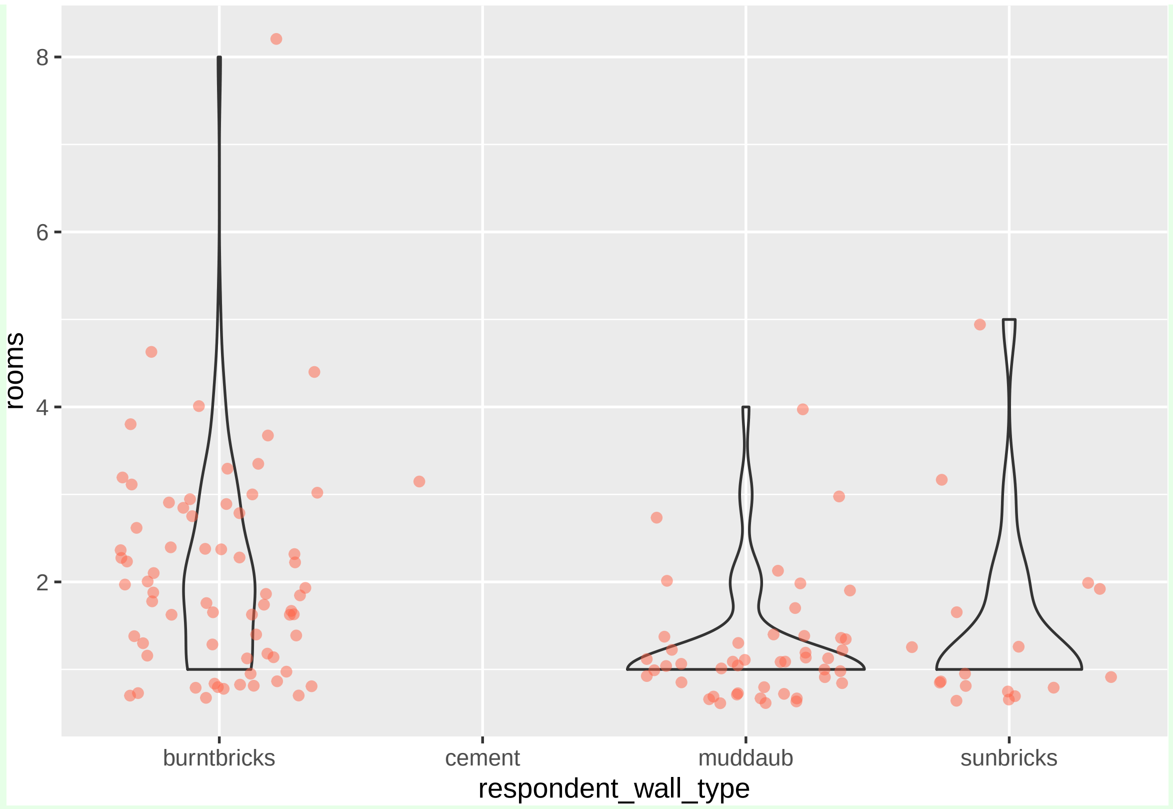
Exercise 1

Boxplots are useful summaries, but hide the shape of the distribution. For example, if the distribution is bimodal, we would not see it in a boxplot.

- An alternative to the boxplot is the violin plot, where the shape (of the density of points) is drawn.. Replace the box plot with a violin plot; use `?geom_violin()` for help as needed.

solution

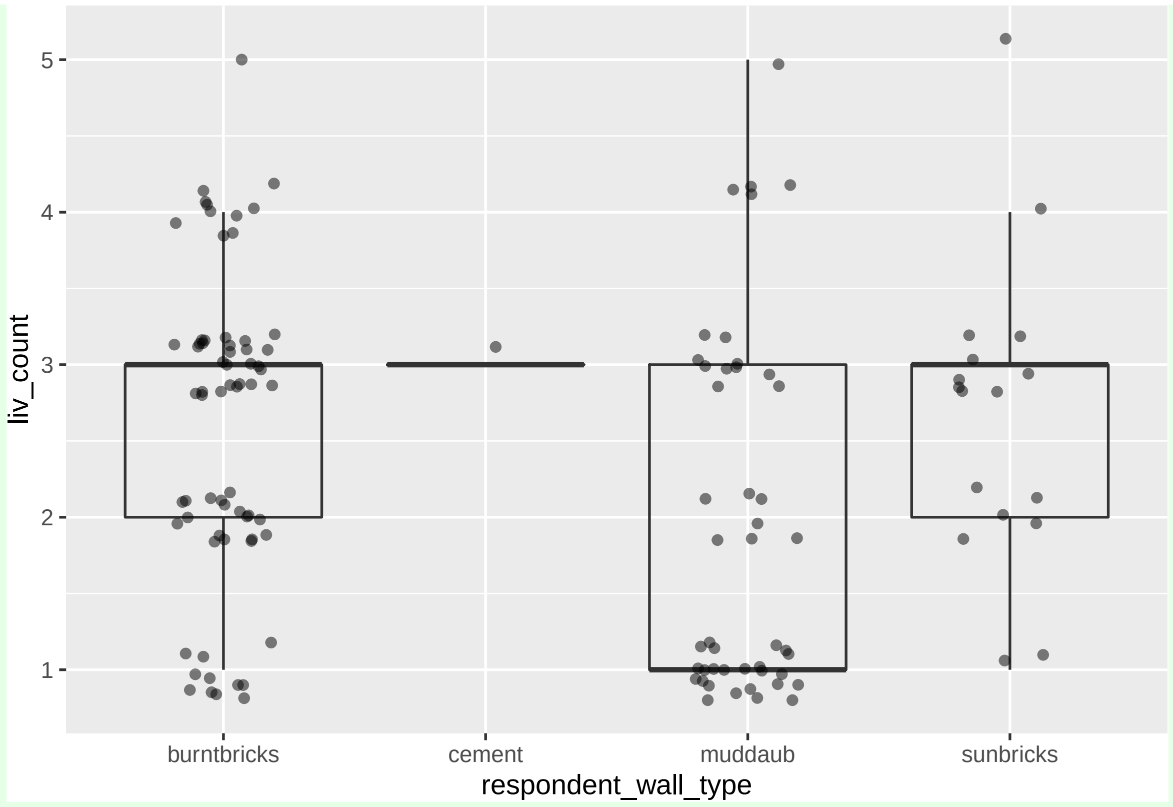
```r
interviews_plotting %>%
  ggplot(aes(x = respondent_wall_type, y = rooms)) +
  geom_violin(alpha = 0) +
  geom_jitter(alpha = 0.5, color = "tomato")
```



- Create a boxplot for `liv_count` for each wall type. Overlay the boxplot layer on a jitter layer to show actual measurements.

solution

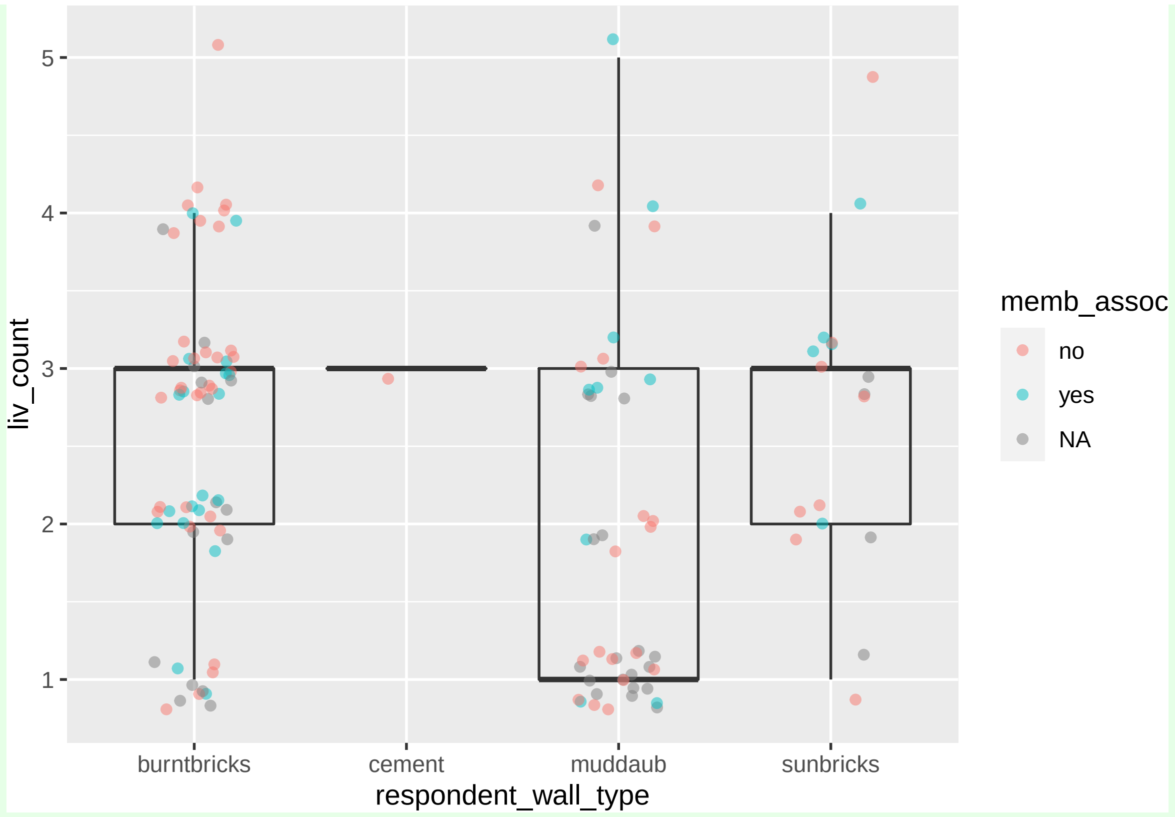
```
interviews_plotting %>%
   ggplot(aes(x = respondent_wall_type, y = liv_count)) +
   geom_boxplot(alpha = 0) +
   geom_jitter(alpha = 0.5, width = 0.2, height = 0.2)
```



- Add colour to the data points on your previous boxplot according to whether the respondent is a member of an irrigation association (memb_assoc).

solution

```r
interviews_plotting %>%
  ggplot(aes(x = respondent_wall_type, y = liv_count)) +
  geom_boxplot(alpha = 0) +
  geom_jitter(aes(color = memb_assoc), alpha = 0.5, width = 0.2, height = 0.2)
```



Exercise 2

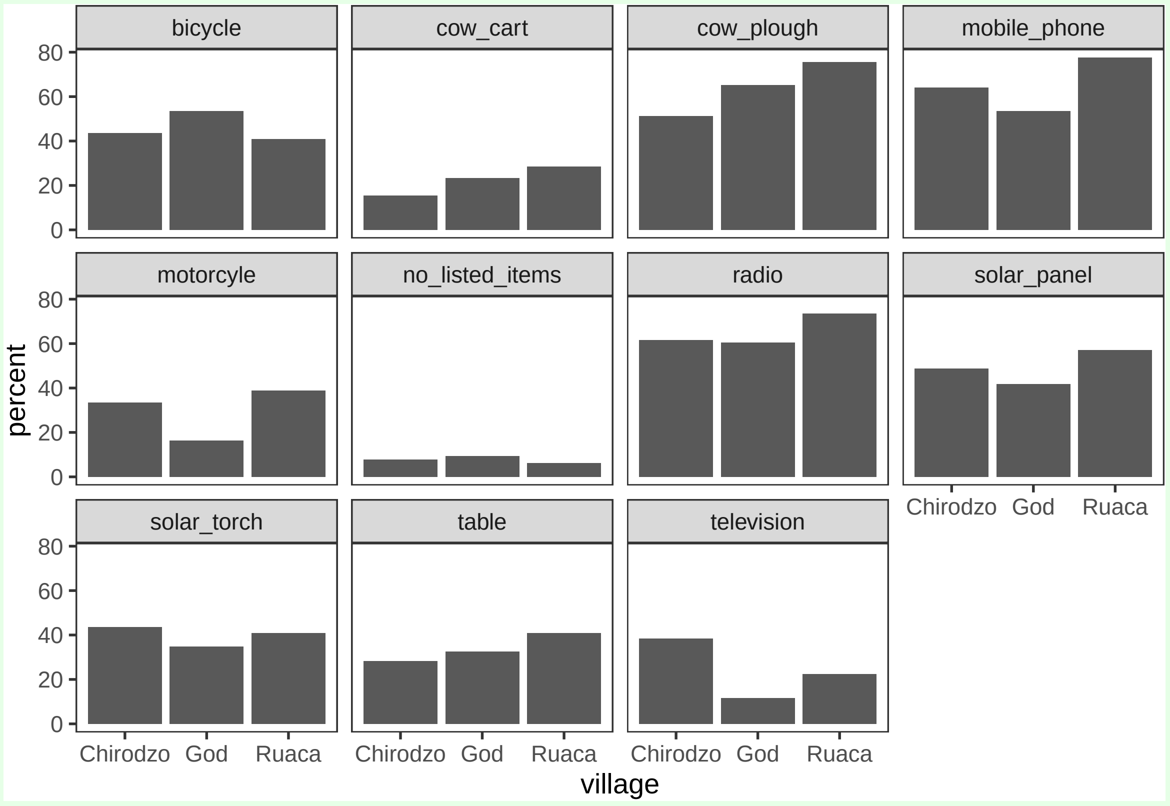
- Create a plot that visualizes the proportion of respondents in each village who owned a particular item?
  - *HINT:* To complete this exercise you will need a count of the number of interviewees in each village. The following lines of code will give you this. You need to join new column of data into the existing dataset to proceed (search `?inner_join` to explore one way to do this).

```
village_pop <- interviews_plotting %>%
        group_by(village) %>%
        count() %>%
        rename(people_in_village = n)
```

solution

```
percent_items <- interviews_plotting %>%
    pivot_longer(cols = bicycle:no_listed_items, names_to = "items",
                 values_to = "items_owned_logical") %>%
    filter(items_owned_logical) %>%
    count(items, village) %>%
    ## add a column with the number of people in each village
    #mutate(people_in_village = case_when(village == "Chirodzo" ~ 39,
    #                                      village == "God" ~ 43,
    #                                      village == "Ruaca" ~ 49)) %>%
    inner_join(village_pop, by = c("village")) %>%
    mutate(percent = (n / people_in_village) * 100)

percent_items %>%
    ggplot(aes(x = village, y = percent)) +
    geom_bar(stat = "identity", position = "dodge") +
    facet_wrap(~ items) +
    theme_bw() +
    theme(panel.grid = element_blank())
```


bicycle
cow_cart
cow_plough
mobile_phone
motorcyle
no_listed_items
radio
solar_panel
solar_torch
table
television
percent
Chirodzo
God
Ruaca
village