Agricultural Statistics: Part 2

In the last module, we used more complex data frame manipulations and calculations to calculate descriptive statistics from the Census of Agriculture. We used `unique` and `length` to calculate the number of commodities a county produces. We used `aggregate` to find the number of farms in each data category. We used `ifelse` to assign labels to data based on their attributes. In this module, we will focus on calculating trends of Census of Agriculture data through time.

Objectives:

1. Use subsets, joins, and calculations across rows to quantify 20-year change in farm size and ownership.
2. Use calculations based on a previous row (with `lag`) to quantify 20-year change in 5-year intervals for agricultural land use and agricultural sales.
3. Use the `dplyr` package for data cleaning.

Getting Started

(1) Open RStudio and your project file. (2) Create a new Quarto file.

(3) Download the following `.csv` files into your `data` folder from the shared Google Drive folder if you have not already done so: num_farms_areafiltered_tenure_1997_2017_ID.csv, cropland_pastureland_total_acres_ID_1997-2017.csv, ag_sales_ID_1997-2017.csv, and ag_expenses_ID_1997-2017.csv.

Trends in farm size and ownership

(4) Create a new code chunk and use `read.csv` to read in the data from num_farms_ areafiltered_tenure_1997_2017_ID.csv. We used this dataset in module 2.1 – it has information about the number of farms in different ownership and size categories. (5) Subset the data to your county. (6) Use two lines of code with the `unique` function to print the unique entries in the columns `Year` and `Domain`. For more clarity, your can click the subset in the Environment tab to display the whole data frame.

(7) Check the data type of the `Value` column. If the data type is `"character"`, use `as.numeric`, as well as `str_remove_all` if necessary, to convert the `Value` column to numbers.

We want to calculate how the number of farms in these categories has changed in the last 20 years. (8) Make **two** subsets: one for 2017 data and one for 1997 data.

(9) Save the total number of farms in each year in two objects called `total_farms_2017` and `total_farms_1997` by retrieving the number in the `Value` column where `Domain` is equal to `TOTAL` in each of the two subsets you created in step (8).

(10) Subtract the number of farms in 1997 from the number of farms in 2017. Record the result in the shared Google sheet. A negative number indicates a loss of farms.

```
total_farms_2017 - total_farms_1997
```

```
[1] -56
```

A raw number of change can be useful, but it's more useful to us if we put it in context. A loss of 56 farms is trivial in a county with thousands of farms, but more concerning in a county with 100 farms. Therefore, we want to calculate the percent change as well as the total change. (11) Divide the total change in farms by the total number of farms in 1997, then multiply by 100 to calculate percent change. Add this value to the shared Google sheet.

```
(total_farms_2017 - total_farms_1997) / total_farms_1997 * 100
```

```
[1] -12.41685
```

It can be helpful to interpret these results in plain language. I would present these results in this way: "Bear Lake county lost 56 of its farms between 1997 and 2017, a reduction of 12%." **(12)** Write a similar statement to interpret the results for your county.

In module 2.1, we applied an equation to an entire column and saved the result in a new column. We can not only apply an equation to a single column, but also create equations with two or more columns.

First, we need to make sure they are lined up correctly. We will line up the number of farms in 1997 and 2017 for each category, and then find the difference for every row. This is the power of coding – we don't have to repeat steps (9)-(10) over and over again!

To do this, we're going to need to do a bit of data cleaning. The best library to do this is called `dplyr`. We can accomplish a lot of things with the base `R` code we've learned so far, but it gets complicated and hard to read very quickly. `dplyr` is generally a better choice for complicated data cleaning, especially if anyone else is going to collaborate with you.

(13) Run `install.packages("dplyr")` in the Console. **(14)** Create a new code chunk and read in the `dplyr` library with `library(dplyr)`.

```
library(dplyr)
```

You may have noticed that our data has a lot of extra columns we're not using. It wasn't a big problem in other modules, but it can make joining columns tedious. We're going to use the `select` function from `dplyr` to grab just the columns we'll need for this calculation. **(15)** Use the code below to select the columns `Domain.Category` and `Value` from your two subsets (you may change the object names).

```
my_county_size_tenure_2017_selected <- select(my_county_size_tenure_2017,
                                              c(Domain.Category, Value))

my_county_size_tenure_1997_selected <- select(my_county_size_tenure_1997,
                                              c(Domain.Category, Value))
```

Notice that the columns names are *not* in quotes here, but we still use the `c()` function to list out multiple names. **(16)** Print the `colnames` for each of these data frames. Make sure that only `Domain.Category` and `Value` remain.

Now, we are going to **join** the two data frames together. There are a lot of different types of joins (if you're interested, see more examples here), but we are going to exclusively use the function `full_join` from `dplyr`.

`full_join` figures out which rows should match based on a key or multiple key columns. We will use `Domain.Category` as our key. We will also specify suffixes for our value columns so we know which year applies to each column. For a visual representation of this process, see Figure 1 on the next page.

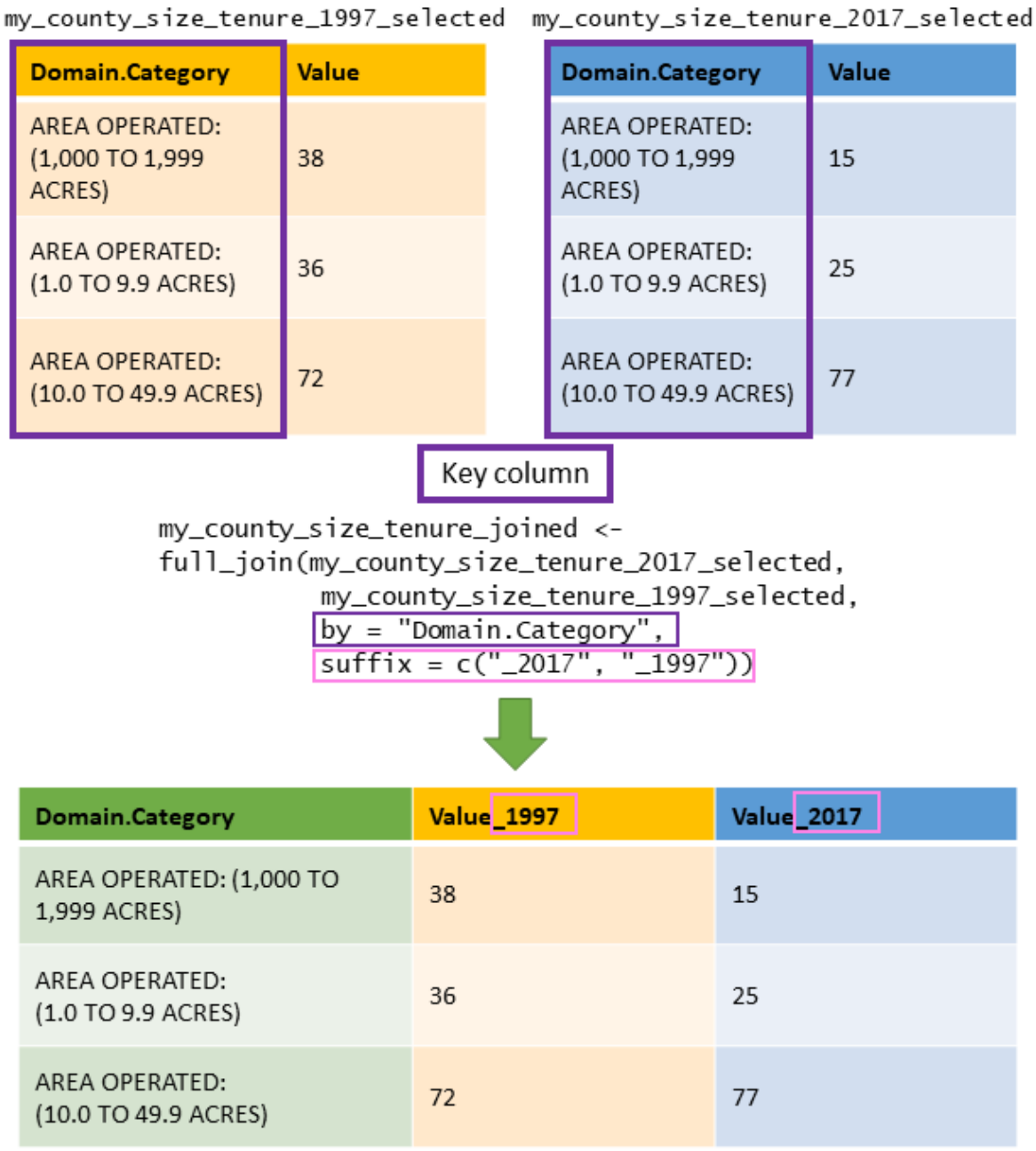


Figure 1: Joining two data frames based on a key column

(17) Use the code below to join the 2017 and the 1997 data into one data frame.

```
my_county_size_tenure_joined <- full_join(my_county_size_tenure_2017_selected,
                                          my_county_size_tenure_1997_selected,
                                          by = "Domain.Category",
                                          suffix = c("_2017", "_1997"))
```

(18) Print the `head` or `tail` of the result and check that it looks right. Are the numbers for 2017 and 1997 similar in each row? You may also want to refer to the two data frames you joined and spot check to see if the numbers ended up in the correct rows.

Now, we can find the different for every row at the same time. **(19)** Create a new column called `Difference`, subtract the `Value_2017` column from the `Value_1997` column and save it in the new column, and print the `head` of the result.

```
my_county_size_tenure_joined$Difference <-
  my_county_size_tenure_joined$Value_2017 -
  my_county_size_tenure_joined$Value_1997

head(my_county_size_tenure_joined)
```

```
                       Domain.Category Value_2017 Value_1997 Difference
1 AREA OPERATED: (1,000 TO 1,999 ACRES)         15         38        -23
2     AREA OPERATED: (1.0 TO 9.9 ACRES)         25         36        -11
3   AREA OPERATED: (10.0 TO 49.9 ACRES)         77         72          5
4     AREA OPERATED: (100 TO 139 ACRES)         38         40         -2
5     AREA OPERATED: (140 TO 179 ACRES)         20         21         -1
6     AREA OPERATED: (180 TO 219 ACRES)         11         22        -11
```

(20) Create a subset of this data where the `Domain.Category` contains the word `TENURE` (hint: use `str_detect`). **(21)** Print the columns `Domain.Category` and `Difference` in your tenure subset, and report the changing number of farms in each category of farm ownership on the shared Google sheet.

(22) Divide the `Difference` column by the `Value_1997` column in your tenure subset and multiply that value by 100. Save the result in a new column called `Percent_difference` and report your results in the shared Google sheet.

Next, let's do the same with the size categories. We need to use the aggregating method we used in module 2.1. **(23)** Create a subset of this data where the `Domain.Category` contains the word `AREA`. **(24)** Create a new column called `label` and fill it with `NA`.

(25) Use `ifelse` to test if `Domain.Category` contains ",000". Every category with this pattern is over 1,000 acres. If yes, then let `label` be "Large farms." If no, let `label` keep its current value.

```
_____$label <- ifelse(test = str_detect(string = _____$_______,
                                        pattern = "____"),
                                  yes = "_________",
                                  no = ______$label)
```

(26) Use `ifelse` to test if `Domain.Category` contains a period. If yes, then let `label` be "Small farms." If no, let `label` keep its current value.

```
_____$label <- ifelse(test = str_detect(__________________,
                                        _________ = "\\."),
                                  _____________,
                                  _____________)
```

(27) Finally, use `ifelse` to test if `label` still equals `NA`. If yes, then let `label` be "Mid-sized farms." If no, let `label` keep its current value.

(28) Print the `Domain.Category` and `label` columns of your area data frame to check the result of this code.

We're going to use the `dplyr` version of `aggregate`, called `summarise`, since we want to keep two columns: `Value_1997` and `Difference`. **(29)** Use the code below to summarise the number of farms in 1997 and the 20-year difference in each category. (The `#` lines are comments to tell you what the line above them does. You don't need to copy them into your report.)

```
my_county_sizes_agg <- summarise(my_county_sizes,
                                 # data frame to be summarised
                                 .by = label,
                                 # group to summarise by
                                 Value_1997_sum = sum(Value_1997, na.rm = TRUE),
                                 Difference_sum = sum(Difference, na.rm = TRUE))
# create new columns which equal the sum of the columns in my_county_sizes
# na.rm equal to true means NAs will be ignored
```

(30) Print the summarised data frame and record the difference of number of farms in each category in the shared Google sheet.

(31) Divide the `Difference_sum` column by the `Value_1997_sum` column in your farm sizes data frame and multiply that value by 100. Save the result in a new column called `Percent_difference` and report your results in the shared Google sheet.

Agricultural Land Use Change

(32) Create a new code chunk and use `read.csv` to read in the data from cropland_pastureland_total_acres_ID_1997-2017.csv. We used this dataset in module 1 – it has information about the amount of land used for cropland, pastureland, and the total land in each county. **(33)** Subset the data to your county. **(34)** Use two lines of code with the `unique` function to print the unique entries in the columns `Year` and `Data.Item`. For more clarity, your can click the subset in the Environment tab to display the whole data frame.

(35) Check the data type of the `Value` column in the land use subset you created in in step 32. If the data type is `"character"`, use `as.numeric`, as well as `str_remove_all` if necessary, to convert the `Value` column to numbers.

Next, we need to separate the total land in each year from the agricultural land in each year. **(36)** Create a subset of this data frame where `Data.Item` contains the string `"NON-AG"` (hint: use `str_detect`). Print the result.

(37) To prepare this data for joining, 'aggregate or `summarise` the `Value` column by `Year`.

To create a data frame of agricultural land for each year, we simply need to flip the selection you just used and subset to all the rows that do *not* contain the string `"NON-AG"`. In `R`, an exclamation point means "not." For example, `!=` means "not equal to."

(38) Create a new code chunk, copy and paste the subset code you used in step (37), change the name of the subset object to indicate that this is a subset of agricultural land, and add a `!` in front of your `str_detect` statement. Print the `unique` values in the `Data.Item` column of your new subset to check that the code worked.

(39) Find the sum of cropland and pastureland acres in each year for your county. You can use `aggregate` or `summarise`. (Hint: find the sum of `Value` by `Year`.)

(40) Use `full_join` to join your agricultural land and total land subsets. Make sure to use suffixes that make sense to you.

```
_____ <- full_join(my_county_agland,
                   my_county_land_incl_non_ag,
                   by = "____",
                   suffix = c("____", "____"))
```

(41) Print the result.

For the next few lines of code to work, we need to arrange the years from earliest (1997) in row 1 to latest (2017) in the last row. Your data may already be in this order. **(42)** If it is not, use the code below to use the `dplyr` function `arrange` to organize the rows of your data by `Year`.

```
my_county_land_join <- arrange(my_county_land_join,
                               Year)
```

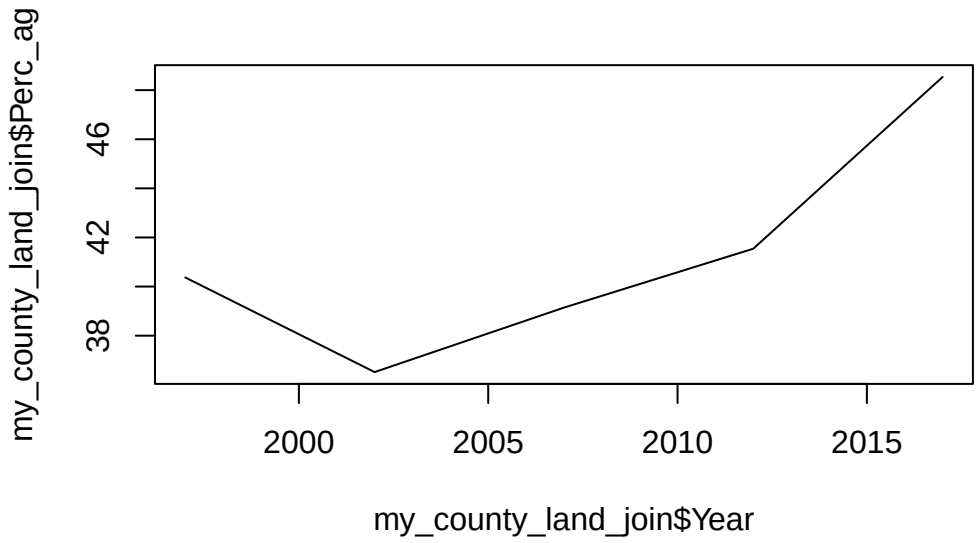
A quirk of this data is that the Census of Agriculture did not start recording the total land in each county until 2002. We can pretty safely assume that the total land in each county was about the same in 1997 as it was in 2002. The code below takes the number in the `Value_total` column, row 2 and copies it into the `Value_total` column, row 1. **(43)** Use the code below to fill in the 1997 total land spot. Make sure to modify this code if your object or column names are different.

```
my_county_land_join$Value_total[1] <- my_county_land_join$Value_total[2]
```

(44) Create a new column, calculate the percent of county land that was used for agriculture in each year, and save the values in the new column.

Optional challenge: (44.1) Check the results of your calculation with a simple graph. Use the code below to track the percentage of agricultural land in your county through time. Make note of any interesting trends you see.

```
plot(x = my_county_land_join$Year,
     y = my_county_land_join$Perc_ag,
     type = "l") # "l" (lowercase L) for line plot
```



The metrics we will use to characterize this change over time are total change over 20 years and recent change from 2012 to 2017. **Optional: (44.2)** Do you see any trends in your plot that you think might be important that are not described by these two metrics? If so, please make note of it for me here so I can improve our research design.

(45) Use the format `data_frame[row_number, "column_name"]` to subtract the percentage of agricultural land in 1997 from the percentage of agricultural land in 2017 to find total change from 1997 to 2017. Record the value in the shared data sheet.

Now, we are going to do the same calculation for each year so we can get the percent change of agricultural land in 5 year increments. The easiest way to refer to the previous year's value is to create a *lag* column. This will line up the values we want to subtract from each other. **(46)** Use the code below to create a new column called `Perc_lag` that contains the values of the column `Perc_ag` shifted down one cell.

```
my_county_land_join$Perc_lag <- lag(my_county_land_join$Perc_ag,
                                    n = 1)

head(my_county_land_join)[, c("Year", "Perc_ag", "Perc_lag")]
```

```
  Year  Perc_ag Perc_lag
1 1997 40.37131       NA
2 2002 36.51727 40.37131
3 2007 39.14223 36.51727
4 2012 41.54188 39.14223
5 2017 48.53603 41.54188
```

(47) Check the results of this code by printing the `head` of your data frame or clicking the data frame in the Environment panel.

(48) Create a new column called `rate` that contains the `Perc_lag` column subtracted from the `Perc_ag` column.

(49) Print the columns `Year` and `rate`. **(50)** Record the rate for 2017 on the shared Google sheet.

(51) Are there any rates that stand out to you that we are not recording in our shared Google sheet? If so, make note of them in your report.

Agricultural Sales

(52) Create a new code chunk and use `read.csv` to read in the data from ag_sales_ID_1997-2017.csv and ag_expenses_ID_1997-2017.csv. These datasets have information about total

agricultural sales and expenses, respectively, for 1997, 2002, 2007, 2012, and 2017. **(53)** Subset each data frame to your county. **(54)** Preview the data by printing the `head` of the columns `Year`, `Data.Item`, `Domain.Category`, and `Value` in each data frame. For more clarity, your can click the subset in the Environment tab to display the whole data frame.

(55) Check the data type of the `Value` columns. If the data type is `"character"`, use `as.numeric`, as well as `str_remove_all` if necessary, to convert the `Value` columns to numbers in both data frames.

(56) Find the sum of sales by year for your county. You can use `aggregate` or `summarise`. **(57)** Do the same for expenses.

(58) Use `full_join` to join your sales and expenses subsets. Make sure to use suffixes that make sense to you. **(59)** Print the result.

(60) If your data frame is not arranged with 1997 in row 1 to 2017 in the last row, use the code in step (42) to organize the rows of your data by `Year`.

(61) Subtract the expenses column from the sales column and save the result in a new column called `profit`.

(Optional: 61.1) Use `plot` to visualize profits over time. Make note of anything that catches your eye about this graph.

(62) Subtract profits in 1997 from profits in 2017 to find total change in profits from 1997 to 2017. Record the value in the shared data sheet.

(63) Create a new column called `profits_lagged` that contains the values of the column `profit` shifted down one cell.

```
_______$______ <- lag(_______$______,
                      n = __)
```

(64) Check the results of this code by printing the `head` of your data frame or clicking the data frame in the Environment panel.

(65) Create a new column called `rate` that contains the `profits_lagged` column subtracted from the `profit` column.

(66) Print the columns `Year` and `rate`. **(67)** Record the rate for 2017 on the shared Google sheet.

(68) Are there any rates that stand out to you that we are not recording in our shared Google sheet? If so, make note of them in your report.

Finishing up

(69) At the end of your report, copy this list and paste it outside of a code chunk to create bullet points. Fill in the data you calculated for your county. Make sure that all of this information is also in the team Google Sheet for comparison with other counties. If you calculated anything extra that not on this list, be sure to add it so you have a record for later.

```
- Change in number of farms, 1997-2017:
- Percent change of farms, 1997-2017:
- Change in number of full-owned farms, 1997-2017:
- Change in number of part-owned farms, 1997-2017:
- Change in number of tenant-operated farms, 1997-2017:
- Percent change of full-owned farms, 1997-2017:
- Percent change of part-owned farms, 1997-2017:
- Percent change of tenant-operated farms, 1997-2017:
- Change in number of small farms, 1997-2017:
- Change in number of mid-sized farms, 1997-2017:
- Change in number of large farms, 1997-2017:
- Percent change of small farms, 1997-2017:
- Percent change of mid-sized farms, 1997-2017:
- Percent change of large farms, 1997-2017:
- Percent change of agricultural land, 1997-2017:
- Recent percent change of agricultural land, 2012-2017:
- Change in agricultural profits, 1997-2017:
- Recent change in agricultural profits, 2012-2017:
```

(70) Go back through your report and add short explanations for what each code chunk does in your own words if you haven't done so already. **(71)** Render your report to a PDF and email it to carolynkoehn@u.boisestate.edu.

Statement of original and referenced work:

The entirety of this module is my original work.