Re-Structuring Data

Restructuring Columns and Rows

As we move into the treatment comparisons part of our course, it is a good time to talk about how datasets are structured. Datasets can have a "wide structure, in which the values for different treatments are listed side by side:

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- 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 become error
```

```
Trt_A = c(1:4)
Trt_B = c(5:8)
Rep = c(1:4)
wide_df = cbind(Rep, Trt_A, Trt_B) %>%
  as.data.frame()

wide_df
```

```
##   Rep Trt_A Trt_B
## 1   1     1     5
## 2   2     2     6
## 3   3     3     7
## 4   4     4     8
```

In general, however, we prefer data be in the long form, so that each observation has its own row in the dataset. In the long form, the data above would be in three columns, the first listing the rep, the second listing the treatment, and the third listing the observed value for that treatment.

To get the data into the correct form, we need to pivot or transpose it: that is, certain columns will become rows, or vice versa. Fortunately, this is easy to do with the "tidyverse" package in R. The "tidyverse" and "tidy" data concept are yet another product from Hadley Wickham, who is an alumnus of Iowa State.

Let's install and run the tidyverse library.

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
library(tidyverse)
```

The tidyverse package introduces us to a new feature in R: pipes. Pipes are connections between lines of code that allow us to work on the same dataset with several lines of code at once. Pipes are signified with the "%>%" string of code.

To make our dataset into the long format, we use the "gather" function to collect the four treatment rows into two rows, one identifying the treatment and the other with the value. "gather" requires three arguments: 1) the name of the column that will identify the data (often called a key), 2) the observed values, and 3) the columns to be reorganized in the key and value columns.

```
long_df = wide_df %>%
  gather(Treatment, Value, Trt_A:Trt_B)

long_df
```

```
##   Rep Treatment Value
## 1   1     Trt_A     1
## 2   2     Trt_A     2
## 3   3     Trt_A     3
## 4   4     Trt_A     4
## 5   1     Trt_B     5
## 6   2     Trt_B     6
## 7   3     Trt_B     7
## 8   4     Trt_B     8
```

In the code above, we created a new data frame, "long_df", from the "wide_df" data frame. We used the "gather" function to create two new columns, "Treatment" and "Value", to contain the treatment name and value associated with each observation. The pipe ("%>%") told the "gather" function to work with the "wide_df" data.

What if we had a long dataset that we wanted to be wide, so that each treatment had its own column? Then we would use the "spread" function. Spread takes two arguments: the column used to identify the values, and the column with the value. Again, the pipe connects the two lines of code.

```
wide_again_df = long_df %>%
  spread(Treatment, Value)

wide_again_df
```

```
##   Rep Trt_A Trt_B
## 1   1     1     5
## 2   2     2     6
## 3   3     3     7
## 4   4     4     8
```

Operations on Data

One of the reasons we may want to restructure data in the wide format is so we can conduct a new statistic from the original values. For example, as we learn this week, we may want to model the distribution of the differences between Trt A and Trt B. Once the data is in the wide format above, we calculcate these differences very quickly.

In R, we use the "mutate" function to build new columns. The argument to "mutate" is usually a mathematical formula. Say we want to cacluate the differences, row by row, between Trt A and Trt B. We would use the following mutate command.

```
trt_differences = wide_again_df %>%
  mutate(Trt_Diff = Trt_A - Trt_B)
```

```
trt_differences
```

```
##   Rep Trt_A Trt_B Trt_Diff
## 1   1     1     5       -4
## 2   2     2     6       -4
## 3   3     3     7       -4
## 4   4     4     8       -4
```

We now have a new column of the differences between Trt_A and Trt_B.

Case Study: Soybean Fungicide

Soybean treated with fungicide applied at R3 was compared with soybean that was untreated. Data are in the soybean_funcicide.csv

```
soybean = read.csv("data/soybean_fungicide.csv")
soybean
```

```
##    Block Treatment Yield
## 1      1         A  71.5
## 2      1         B  75.1
## 3      2         A  73.5
## 4      2         B  77.7
## 5      3         B  77.7
## 6      3         A  72.4
## 7      4         B  78.2
## 8      4         A  75.0
## 9      5         A  76.4
## 10     5         B  75.9
## 11     6         A  76.5
## 12     6         B  77.5
## 13     7         B  80.7
## 14     7         A  74.8
## 15     8         A  75.6
## 16     8         B  78.4
## 17     9         B  80.7
## 18     9         A  73.2
## 19    10         B  76.9
## 20    10         A  74.6
```

This dataset is currently in the long form. To calculate the differenced between Treatments A and B, we need need to convert it to the wide form.

```
soybean_wide = soybean %>%
  spread(Treatment, Yield)
soybean_wide
```

```
##    Block    A    B
## 1      1 71.5 75.1
## 2      2 73.5 77.7
## 3      3 72.4 77.7
## 4      4 75.0 78.2
## 5      5 76.4 75.9
## 6      6 76.5 77.5
## 7      7 74.8 80.7
## 8      8 75.6 78.4
## 9      9 73.2 80.7
## 10    10 74.6 76.9
```

We now have the two treatments in columns A and B. We can then calculate the difference betwee Treatment A and Treatment B using the mutate function..

```
diffs = soybean_wide %>%
  mutate(diff = A-B)
diffs
```

```
##    Block    A    B diff
## 1      1 71.5 75.1 -3.6
## 2      2 73.5 77.7 -4.2
## 3      3 72.4 77.7 -5.3
## 4      4 75.0 78.2 -3.2
## 5      5 76.4 75.9  0.5
## 6      6 76.5 77.5 -1.0
## 7      7 74.8 80.7 -5.9
## 8      8 75.6 78.4 -2.8
## 9      9 73.2 80.7 -7.5
## 10    10 74.6 76.9 -2.3
```

Practice 1

The dataset "darwin.csv" is from a trial by Charles Darwin in which he compared the heights of progeny of corn plants that were self-fertilized and cross-fertilized. Three pairs of plants were grown per pot. Measurements are in inches.

```
darwin = read.csv("data/darwin.csv")

head(darwin)
```

```
##   pot pair  type height
## 1   I    a cross 23.500
## 2   I    a  self 17.375
## 3   I    b cross 12.000
## 4   I    b  self 20.375
## 5   I    c cross 21.000
## 6   I    c  self 20.000
```

The data are currently in the long format. We want to convert it to the wide format, where each treatment has its heights in a separate column. Re-structure the type and height columns by adding the spread function to the code chunk below

```
darwin_wide = darwin
```

The first few rows of the resulting dataset should look like:

I a 23.500 17.375
I b 12.000 20.375
I c 21.000 20.000
II d 22.000 20.000

Practice 2

These data are from a trial that compared apple genotypes.

Calculate the difference between the two apple genotypes (Golden and Redspur) by completing the "mutate" function below.

The first two rows should look like:

R1 121.7750 127.5900 -5.81500
R2 146.7222 126.7625 19.95972

Practice 3

Manganese was applied to soybean at the V5 stage near New Baltimore, Ohio.

```r
manganese = read.csv("data/soybean_manganese.csv")
head(manganese)
```

```
##   Block Treatment Yield
## 1     1         A  78.9
## 2     1         B  81.7
## 3     2         A  80.9
## 4     2         B  84.3
## 5     3         B  84.3
## 6     3         A  79.8
```

Convert the data to wide format so the two treatments are in separate columns

The first two rows of the resulting dataset should look like:

1 78.9 81.7
2 80.9 84.3

Then calculate the difference between treatments A and B by subtracting treatment B from treatment A.

The first couple of rows of the final dataset should look like:

1 78.9 81.7 -2.8
2 80.9 84.3 -3.4

Practice 4

Data are from a wheat trial where fungicide was applied during flag leaf stage.

```r
wheat = read.csv("data/wheat_fungicide.csv")
head(wheat)
```

```
##   Block Treatment Yield
## 1     1         B  95.9
## 2     1         A  93.8
## 3     2         A  92.3
## 4     2         B  97.5
## 5     3         A  98.7
## 6     3         B  98.8
```

Convert the data to wide format. The first few rows should look like:

1 93.8 95.9
2 92.3 97.5
3 98.7 98.8

Calculate the difference between treatments by subtracting Treatment A from Treatment B. The first few rows should look like:

1	93.8	95.9	-2.1
2	92.3	97.5	-5.2
3	98.7	98.8	-0.1