HW5_Sauer_Annie

Annie Sauer

9/20/2018

Problem 3

A good figure should accurately describe the data. Figures should not be intentionally misleading. They should have descriptive yet concise titles and labels and should be easy to understand.

Problem 4

```r
# Create function to compute the proportion of successes in a vector
success <- function(x){
  # x: vector of 1 and 0 (1 represents success)
  return(sum(x)/length(x))
}

# Create matrix to simulate 10 flips of a coin
set.seed(12345)
data <- matrix(rbinom(10, 1, prob = (30:40)/100), nrow = 10, ncol = 10)

# Success by row
apply(data, 1, success)
```

```
##  [1] 1 1 1 1 0 0 0 0 1 1
```

```r
# Success by column
apply(data, 2, success)
```

```
##  [1] 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6
```

```r
# Notice - each column is identical.  This is because we only call the
# rbinom function once.  To fix this, we need to call the rbinom function
# 10 times.

# Create a new function that outputs a vector of length 10 from an input 0 <= p <= 1
binomial <- function(p){
  if (p < 0 | p >1){
    cat('Error: not a valid probability')
    break
  }
  return(rbinom(10, 1, prob = p))
}

# Create vector of desired probabilities
prob <- c((30:40)/100)
new_data <- matrix(ncol = 11, nrow = 10)

# Create desired matrix
for (i in 1:length(prob)){
  new_data[,i] = binomial(prob[i])
}

# Calculate success rate for each column
apply(new_data, 2, success)
```

```
##  [1] 0.2 0.3 0.4 0.3 0.4 0.6 0.3 0.3 0.5 0.6 0.5
```

Problem 5

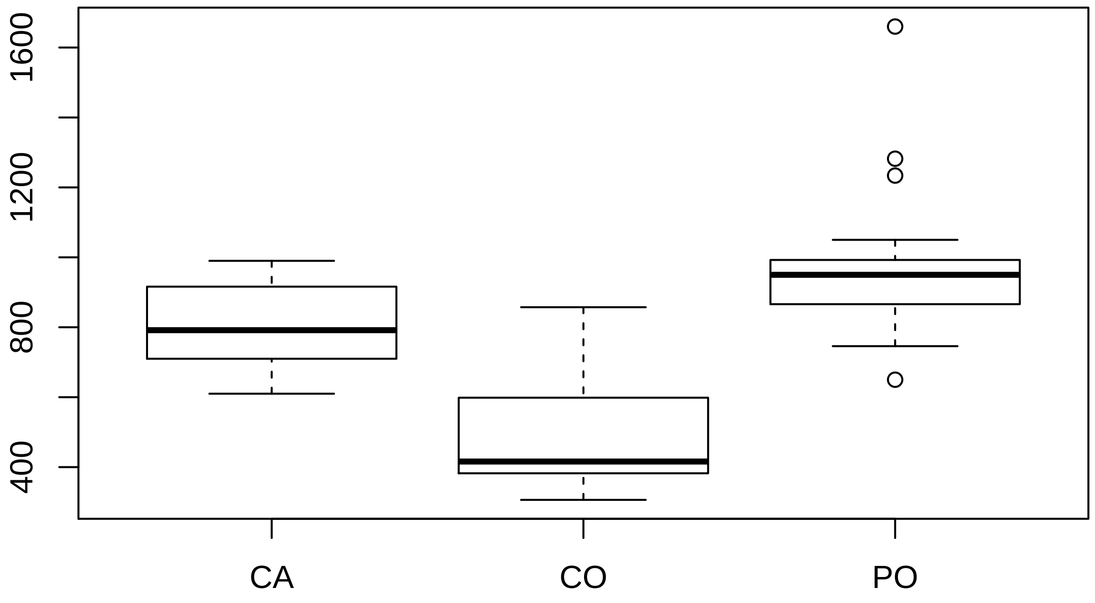
```r
starch_data <- fread('http://www2.isye.gatech.edu/~jeffwu/book/data/starch.dat',
                     sep = ' ')
head(starch_data)
```

```
##    starch strength thickness
## 1:     CA    791.7       7.7
## 2:     CA    610.0       6.3
## 3:     CA    710.0       8.6
## 4:     CA    940.7      11.8
## 5:     CA    990.0      12.4
## 6:     CA    916.2      12.0
```

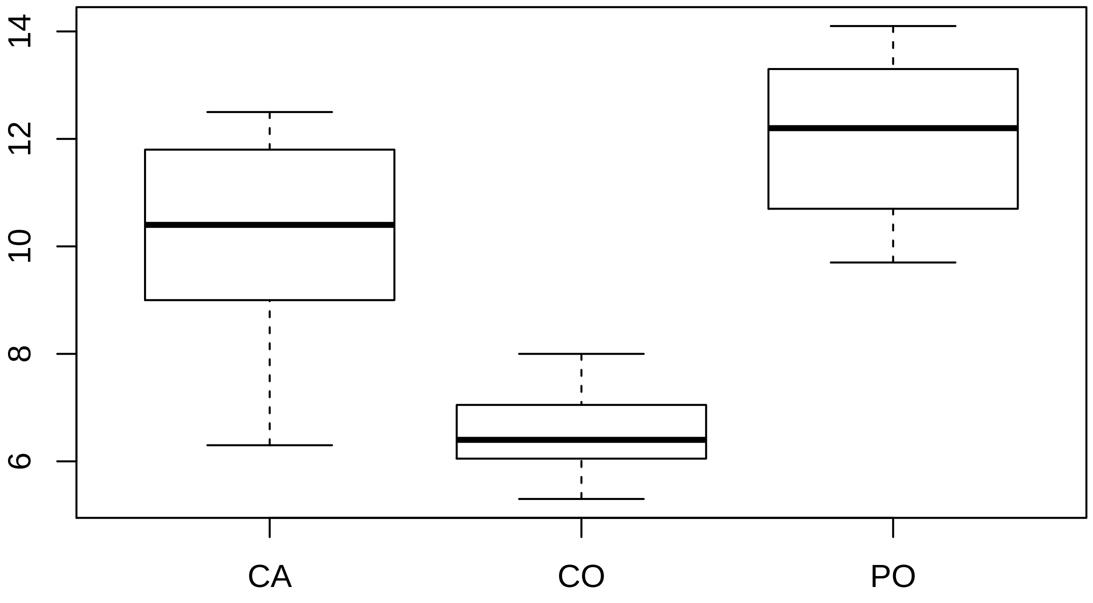
```r
starch_data <- group_by(starch_data, starch)
summarise(starch_data, mean_strength = mean(strength), sd_strength = sd(strength),
          mean_thickness = mean(thickness), sd_thickness = sd(thickness))
```

```
## # A tibble: 3 x 5
##   starch mean_strength sd_strength mean_thickness sd_thickness
##   <chr>          <dbl>       <dbl>          <dbl>        <dbl>
## 1 CA              795.        139.          10.2         1.97 
## 2 CO              483.        158.           6.53        0.741
## 3 PO              976.        238.          12.0         1.51
```

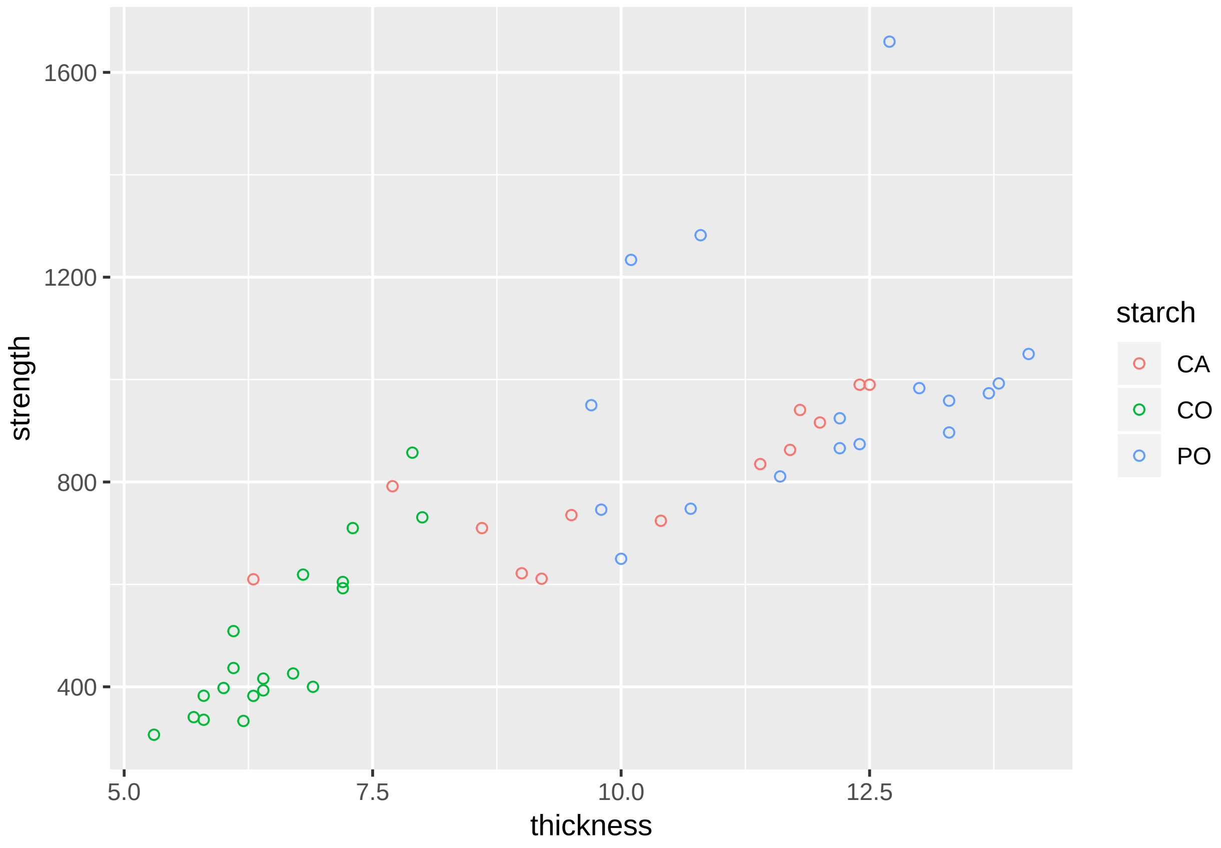
```r
boxplot(starch_data$strength~starch_data$starch)
```



```r
boxplot(starch_data$thickness~starch_data$starch)
```



```r
ggplot(starch_data, aes(x=thickness, y=strength, color=starch)) + geom_point(shape=1)
```



There is a positive relationship between thickness and strength across all starch types. The CO starch generally has the lowest thickness and strength. The CA and PO types appear to overlap in both thickness and strength, but the PO type has the highest values of both variables.

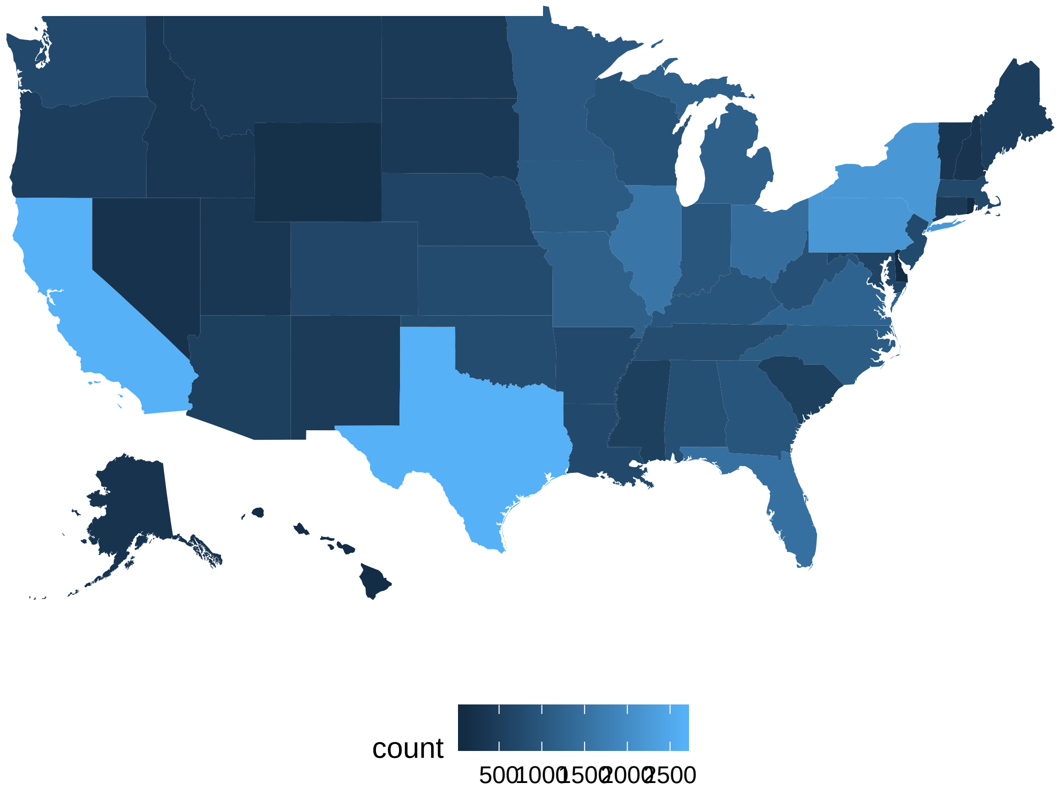
Problem 6

```r
#we are grabbing a SQL set from here
# http://www.farinspace.com/wp-content/uploads/us_cities_and_states.zip

#download the files, looks like it is a .zip
library(downloader)
```

```
##
## Attaching package: 'downloader'

## The following object is masked from 'package:devtools':
##
##     source_url
```

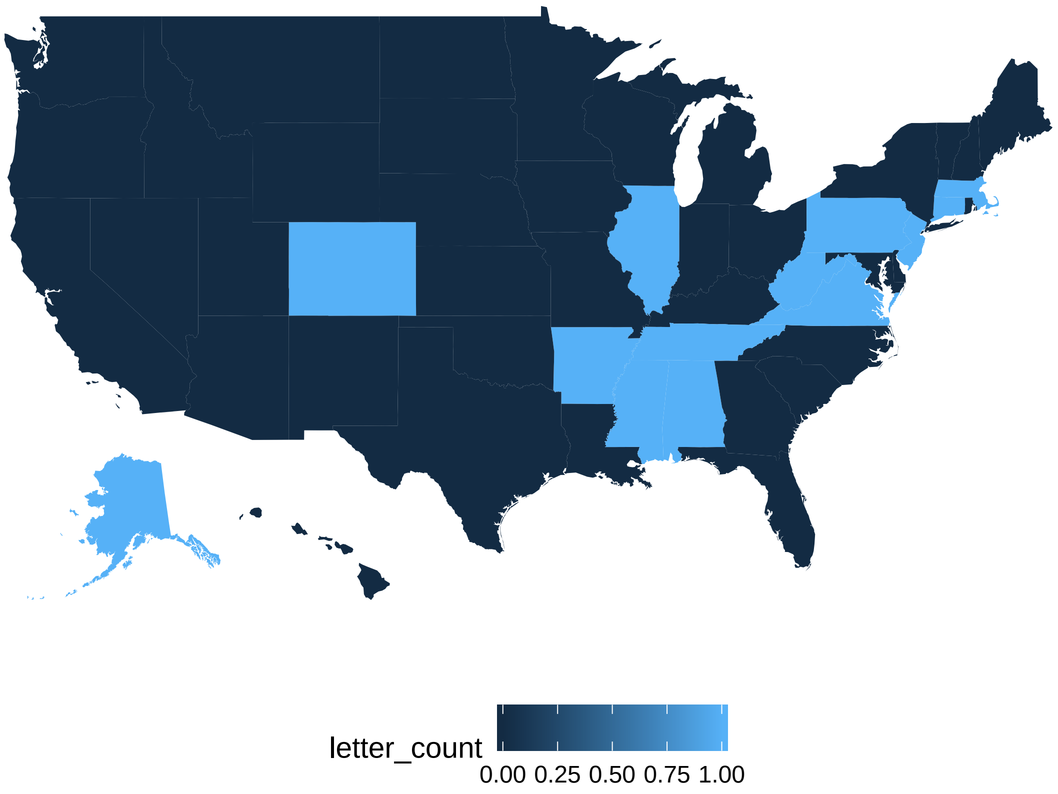
```r
download("http://www.farinspace.com/wp-content/uploads/us_cities_and_states.zip",
             dest="us_cities_states.zip")
unzip("us_cities_states.zip", exdir="./")

#read in data, looks like sql dump, blah
library(data.table)
states <- fread(input = "./us_cities_and_states/states.sql",skip = 23,
                    sep = "'", sep2 = ",", header = F, select = c(2,4))
### YOU do the CITIES
### I suggest the cities_extended.sql may have everything you need
### can you figure out how to limit this to the 50?
cities <- fread(input = "./us_cities_and_states/cities_extended.sql",skip = 23,
                    sep = "'", sep2 = ",", header = F, select = c(2, 4, 6, 8, 10, 12))
# Delete columns that are not in U.S. states
state_code <- states$V4
cities_us <- subset(cities, V4 %in% state_code)

# Create summary table of number of cities by state
by_state <- group_by(cities_us, V4)
total_cities <- summarise(by_state, count = length(V4))
total_cities['state'] <- states$V2

# Create a function that counts the number of occurences of a letter in a string
letter_tally <- function(letter, state_name){
  # Input values must be strings
  return(str_count(state_name, letter))
}

# make state names lowercase
total_cities$state <- tolower(total_cities$state)

# Create a matrix of the number of occurences of each letter in each state's name
letter_count <- data.frame(matrix(NA,nrow=50, ncol=26))
state_name <- tolower(states$V2)
for(i in 1:50){
  for (j in 1:26){
    letter_count[i,j] = letter_tally(letters[j], state_name[i])
  }
}
```

```r
# Create plots
data("fifty_states") # this line is optional due to lazy data loading
crimes <- data.frame(state = tolower(rownames(USArrests)), USArrests)



# Plot of total number of cities
# map_id creates the aesthetic mapping to the state name column in your data
p <- ggplot(total_cities, aes(map_id = state)) +
# map points to the fifty_states shape data
geom_map(aes(fill = count), map = fifty_states) +
expand_limits(x = fifty_states$long, y = fifty_states$lat) +
coord_map() +
scale_x_continuous(breaks = NULL) +
scale_y_continuous(breaks = NULL) +
labs(x = "", y = "") +
theme(legend.position = "bottom",
panel.background = element_blank())

p
```



```r
#ggsave(plot = p, file = "HW5_Problem6_Plot_Settlage.pdf")

# Plot of >3 repeated letters in state name
# Collapse letter count into one vector
three_letters <- c(rep(0, times = 51))
for (i in 1:51){
  three_letters[i] = max(letter_count[i,])
}
```

```r
three_letters_indicator <- c(rep(0, times = 51))
for (i in 1:50){
  if (three_letters[i] >= 3){
    three_letters_indicator[i] = 1
  }
}

total_cities$letter_count <- three_letters_indicator

p <- ggplot(total_cities, aes(map_id = state)) +
# map points to the fifty_states shape data
geom_map(aes(fill = letter_count), map = fifty_states) +
expand_limits(x = fifty_states$long, y = fifty_states$lat) +
coord_map() +
scale_x_continuous(breaks = NULL) +
scale_y_continuous(breaks = NULL) +
labs(x = "", y = "") +
theme(legend.position = "bottom",
panel.background = element_blank())

p
```


letter_count
0.00 0.25 0.50 0.75 1.00