STA 141 Worksheet 5

Richard McCormick

October 12, 2023

Due Date: Thursday, October 19, 2023 before 11:00am.

Instructions

Worksheets must be turned in as a PDF file through Canvas. The worksheet is worth a total of **15 points**, which is 3 percent of your overall grade.

Exercises

Begin by running the following code block to add the packages we need to use to our library. If you haven't installed `Stat2Data` previously then you'll have to install the package first.

Exercise 1

(a) The dataset from this package that we are going to use is called `SandwichAnts`. It summarizes experiments where people left sandwiches with types of bread and different fillings out in the open and counted the number of ants that it attracted. Let's save a copy of the dataset to our environment so that we can use it.

```
data("SandwichAnts")
my.ants <- SandwichAnts
```

Use commands we've learned to look at the different bread types and fillings used in the experiments.

```
unique( my.ants$Bread )
```

```
## [1] WholeWheat MultiGrain Rye        White
## Levels: MultiGrain Rye White WholeWheat
```

```
unique( my.ants$Filling )
```

```
## [1] HamPickles   PeanutButter Vegemite
## Levels: HamPickles PeanutButter Vegemite
```

(b) Produce a frequency table to check that the experiments used each bread type an equal number of times. Do the same for the fillings.

```
table( my.ants$Bread )
```

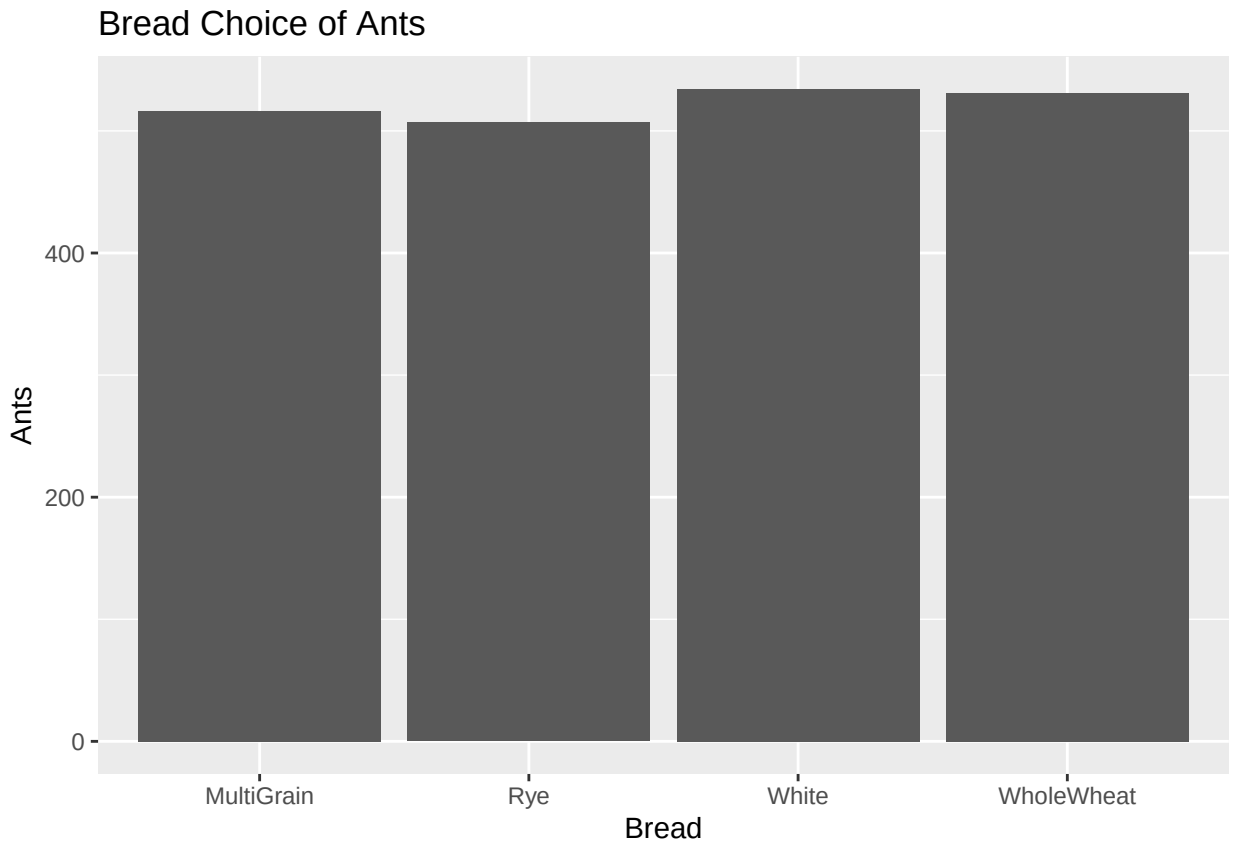
```
##
## MultiGrain        Rye      White WholeWheat
##         12         12         12         12
```

```
table( my.ants$Filling )
```

```
##
##   HamPickles PeanutButter     Vegemite
##           16           16           16
```

(c) Recalling what we did last week, plot a bivariate bar graph of the `Ants` variable against the `Bread` variable.

```
ants_plot <- ggplot( data=my.ants ) +
  geom_bar( stat='identity', aes( x=Bread, y=Ants ) ) +
  labs( title="Bread Choice of Ants" )

ants_plot
```

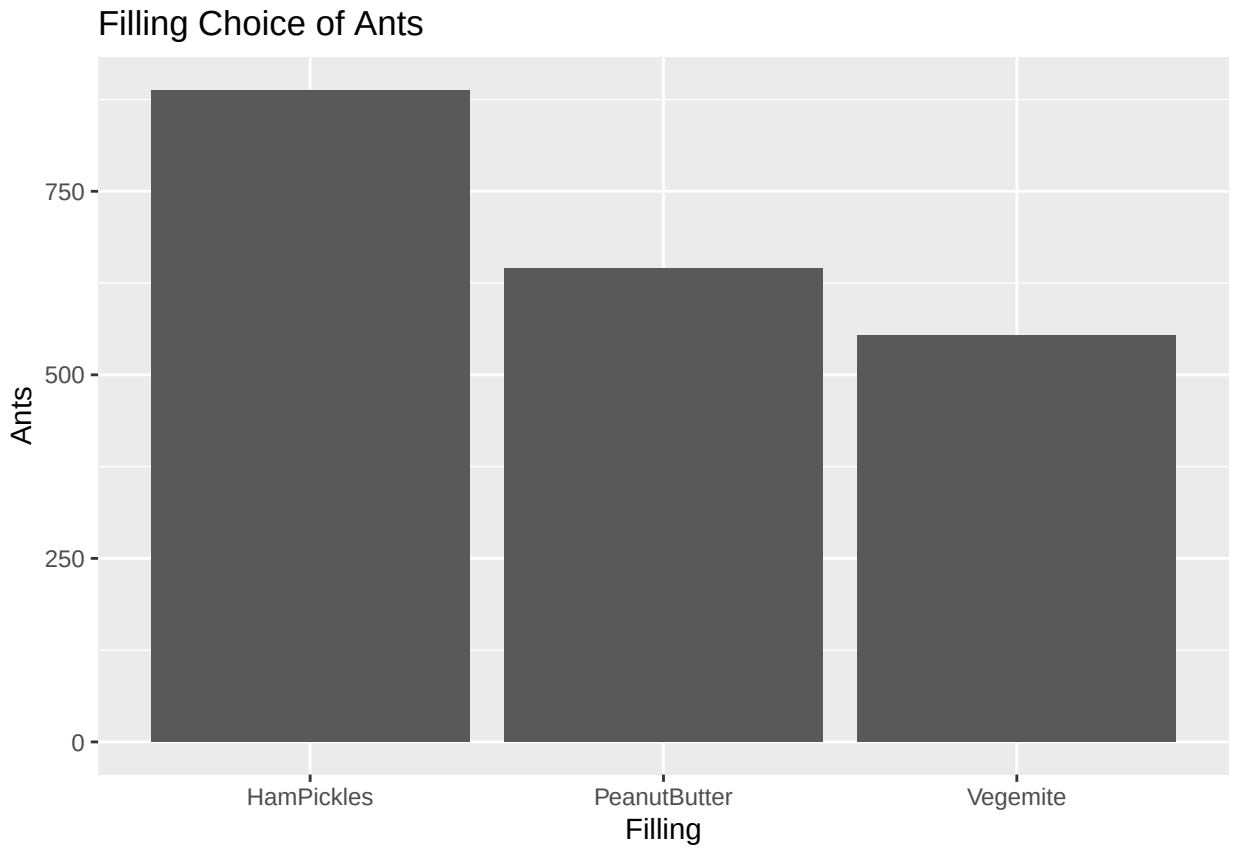


(d) Based on these experiments which bread do ants prefer? And by how much?

It appears that the ants prefer white bread by a small amount.

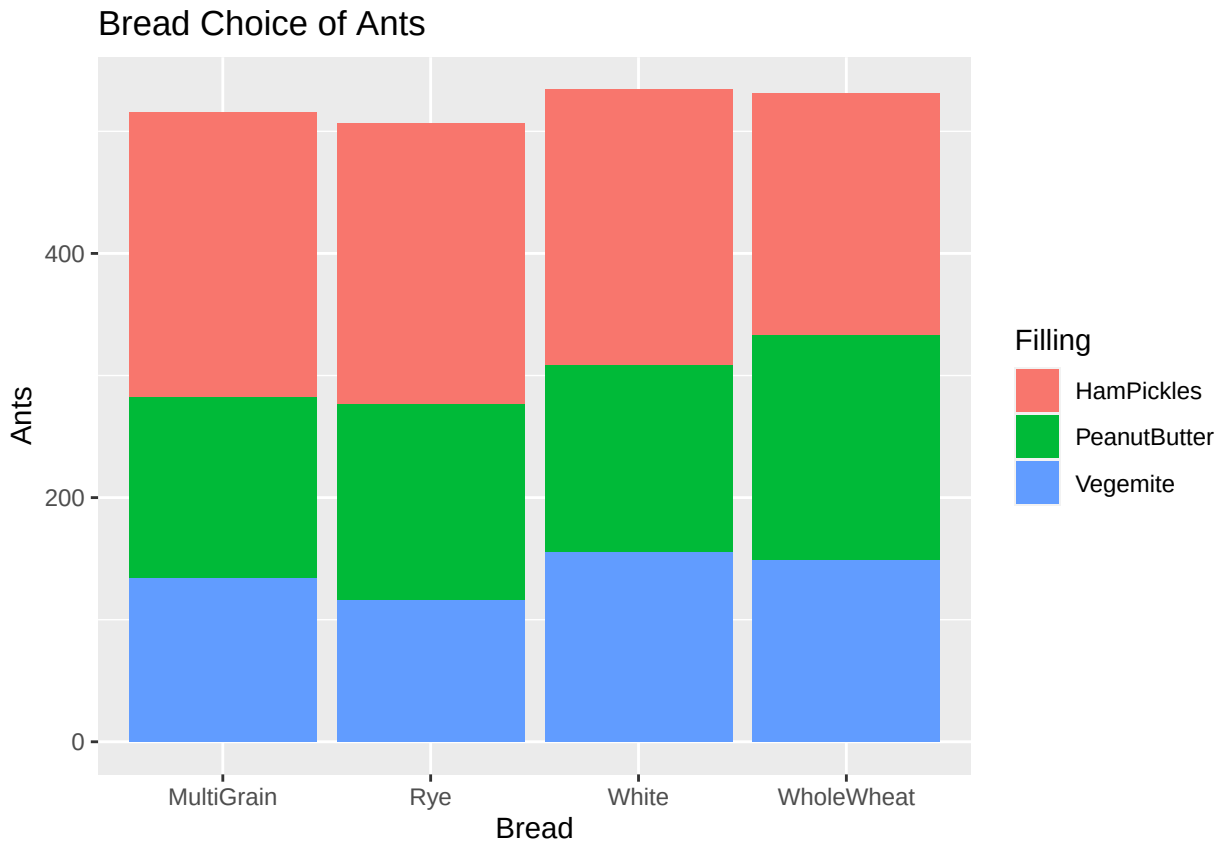
(e) Recalling what we did last week, plot a bivariate bar graph of the `Ants` variable against the `Filling` variable.

```
ants_plot <- ggplot( data=my.ants ) +
  geom_bar( stat='identity', aes( x=Filling, y=Ants ) ) +
  labs( title="Filling Choice of Ants" )

ants_plot
```



(f) Sometimes we find that an overall trend doesn't hold across sub-groups of the data. Produce a stacked bar chart to look at how the number of ants changes with different fillings.

```
ants_plot <- ggplot( data=my.ants ) +
  geom_bar( stat='identity', aes( x=Bread, y=Ants, fill=Filling ) ) +
  labs( title="Bread Choice of Ants" )

ants_plot
```

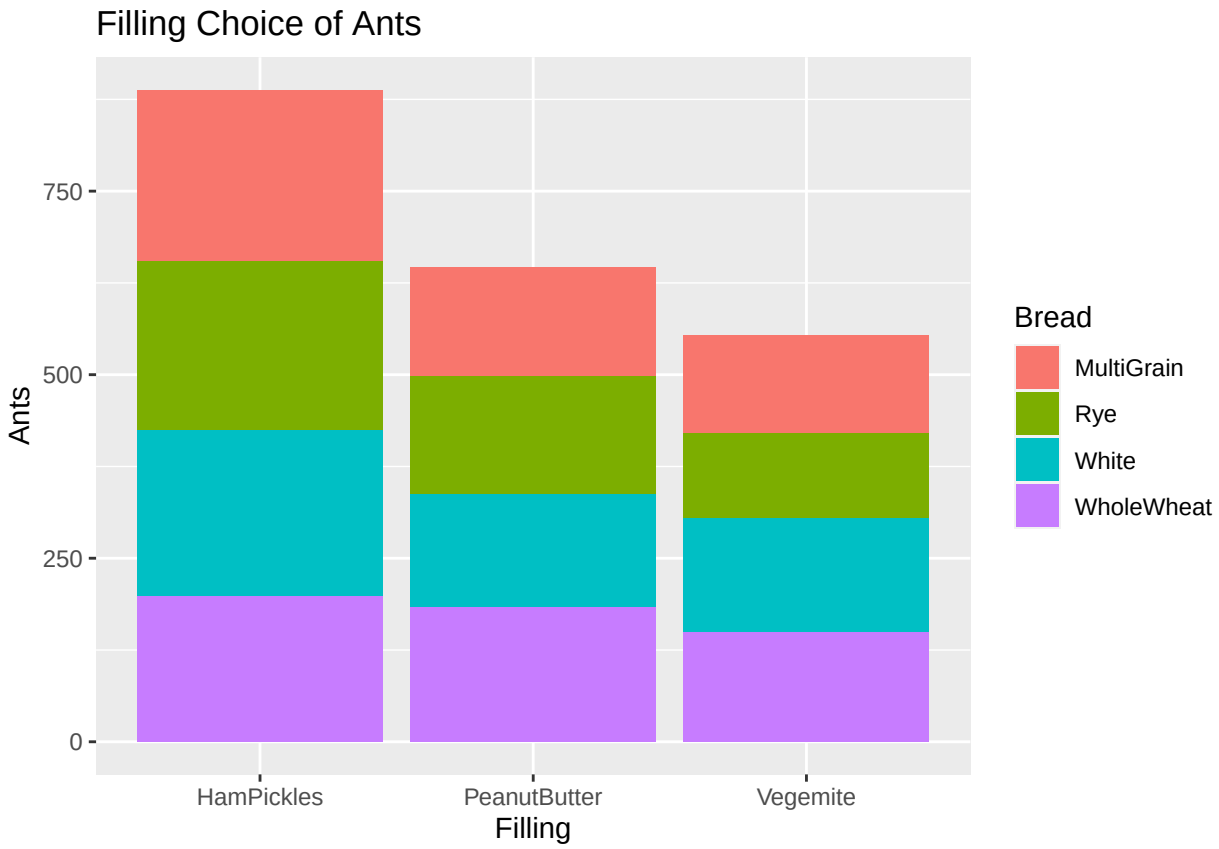


(g) Does it look like ants prefer one filling over the other two?

It looks like the ants prefer the Ham & Pickles filling over other fillings.

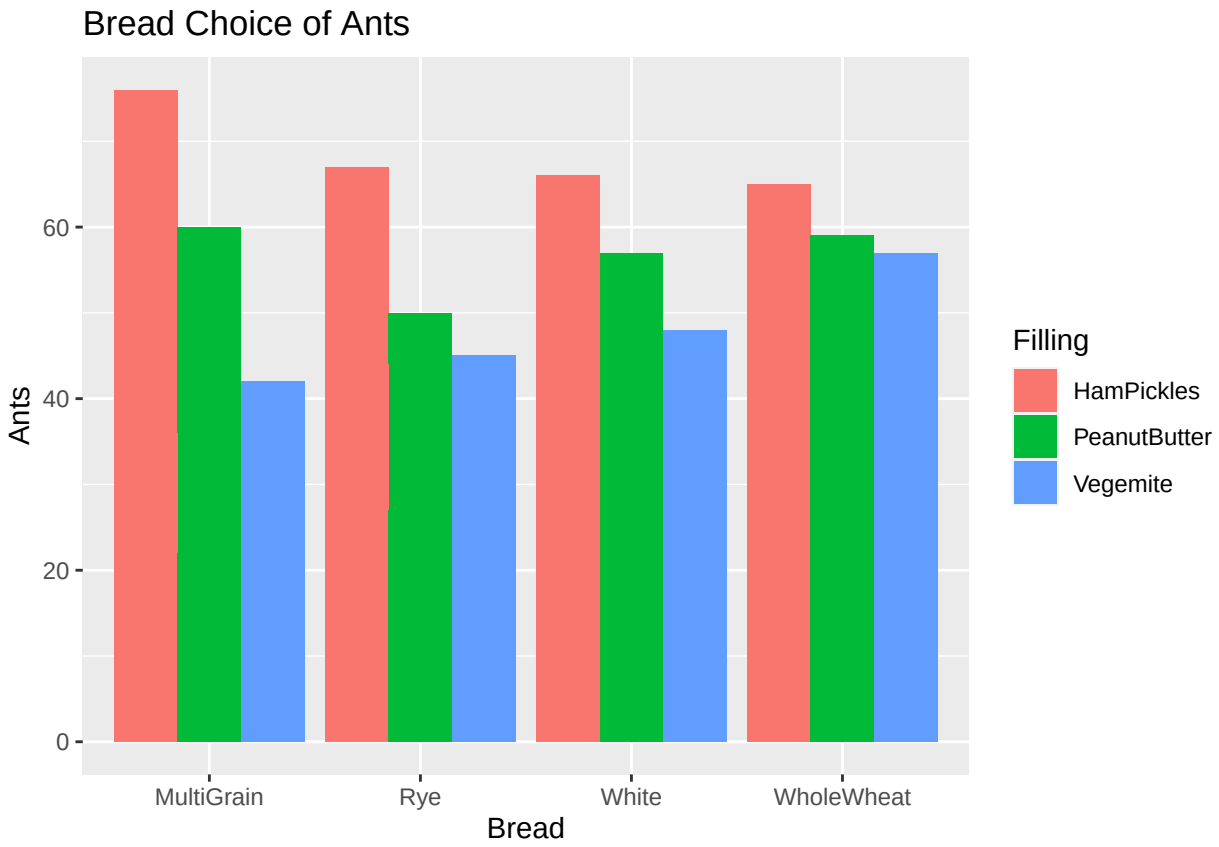
(h) Check your answer to part (f) by switching the mapping of the `Bread` and `Filling` variables. That is, map `Filling` to the `x`-axis and `Bread` to the fill (or color) aesthetic.

```
ants_plot <- ggplot( data=my.ants ) +
  geom_bar( stat='identity', aes( x=Filling, y=Ants, fill=Bread ) ) +
  labs( title="Filling Choice of Ants" )

ants_plot
```



(i) Produce a grouped bar chart using the same variable mappings as the stacked bar chart (f).

```
ants_plot <- ggplot( data=my.ants ) +
  geom_bar( stat='identity', position="dodge",
            aes( x=Bread, y=Ants, fill=Filling ) ) +
  labs( title="Bread Choice of Ants" )

ants_plot
```



(j) Using this plot which is the most popular bread-filling combination and which is the least?

It appears the most popular bread-filling combination is Multigrain Ham & Pickles, while the least popular is Muligrain Vegemite.

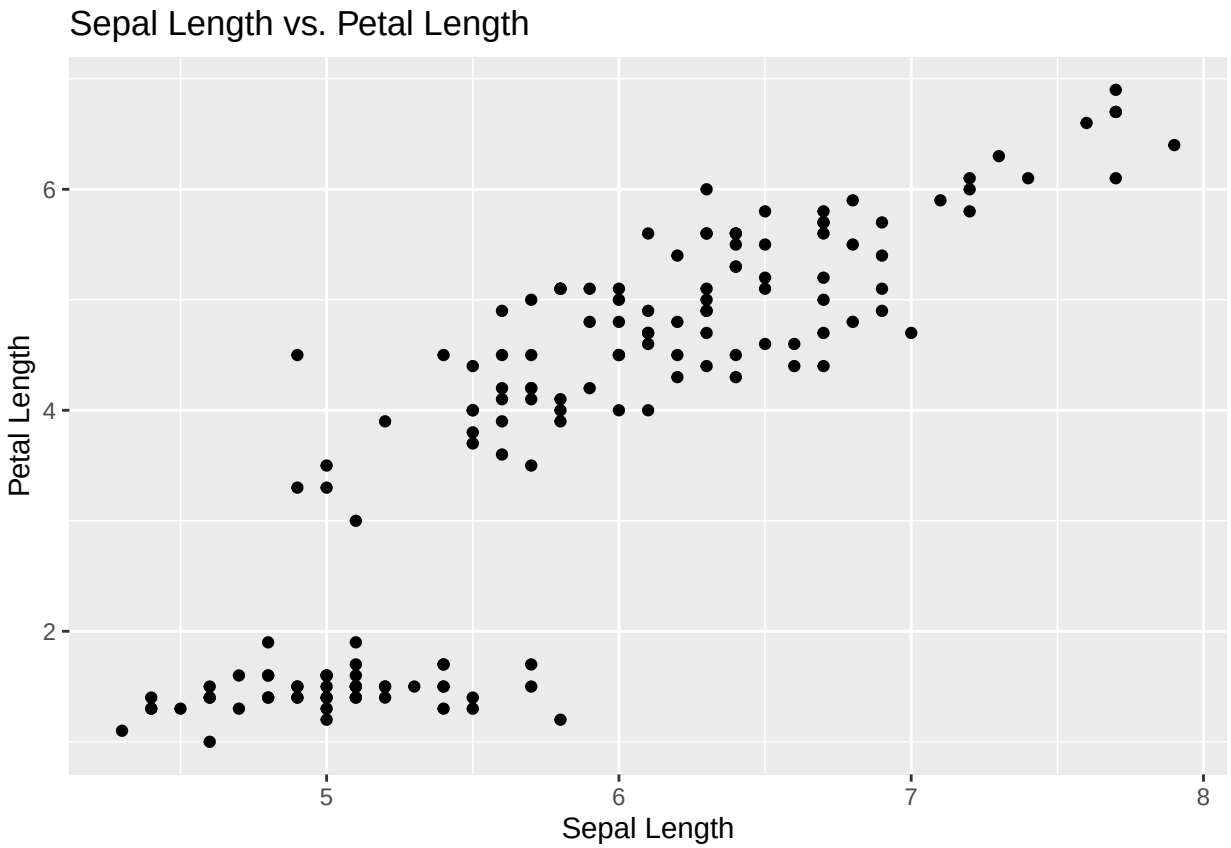
Exercise 2

(a) As we saw in the lecture, one of the most famous datasets in data science is the Iris dataset. It has measurements on 4 variables for 3 different types of flowers. Let's explore multivariate scatterplots using this data. Run the following code block to save a copy of the dataset to a variable called `my.iris`.

```
my.iris <- iris
```

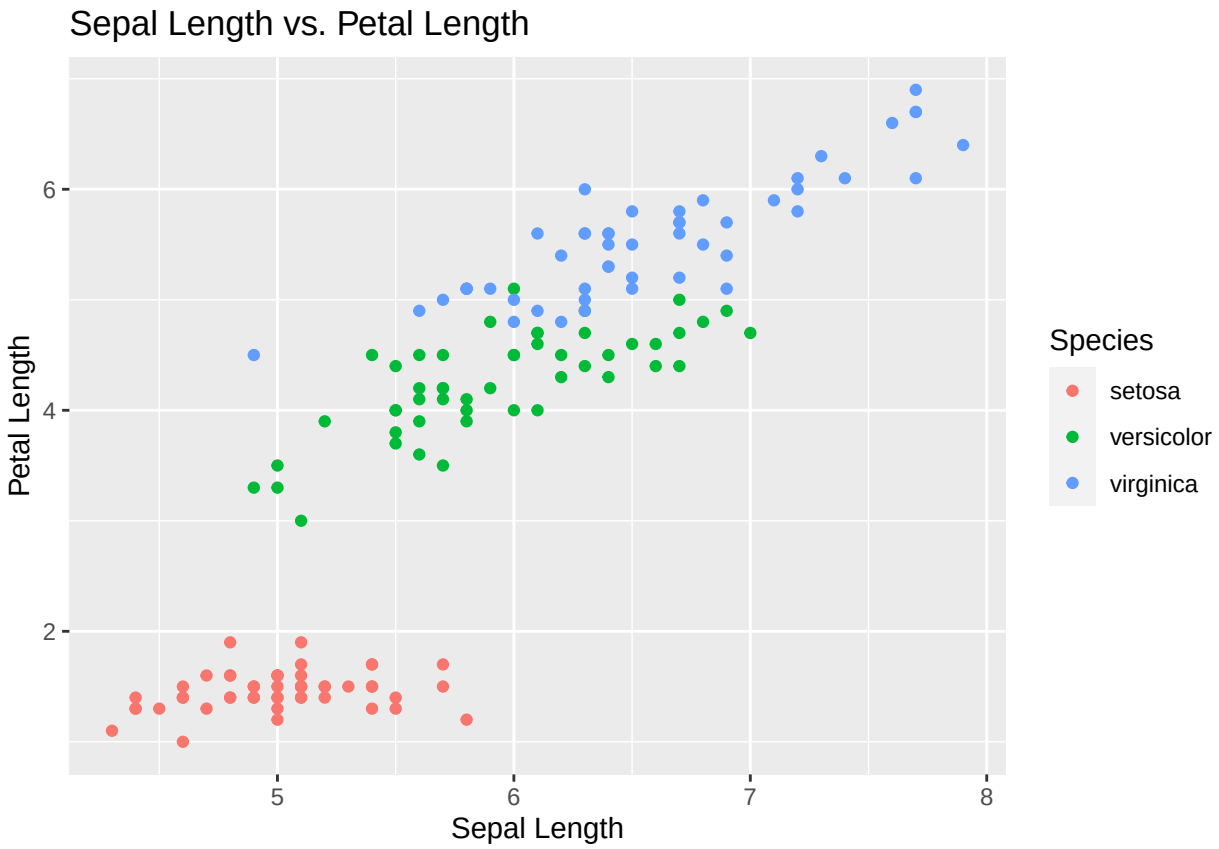
(a) Using our knowledge from Worksheet 4, produce a scatterplot of Petal Length against Sepal Length for the whole dataset.

```
iris_plot <- ggplot( data=my.iris ) +
  geom_point( aes( x=Sepal.Length, y=Petal.Length ) ) +
  labs( title="Sepal Length vs. Petal Length", x="Sepal Length",
        y="Petal Length" )

iris_plot
```



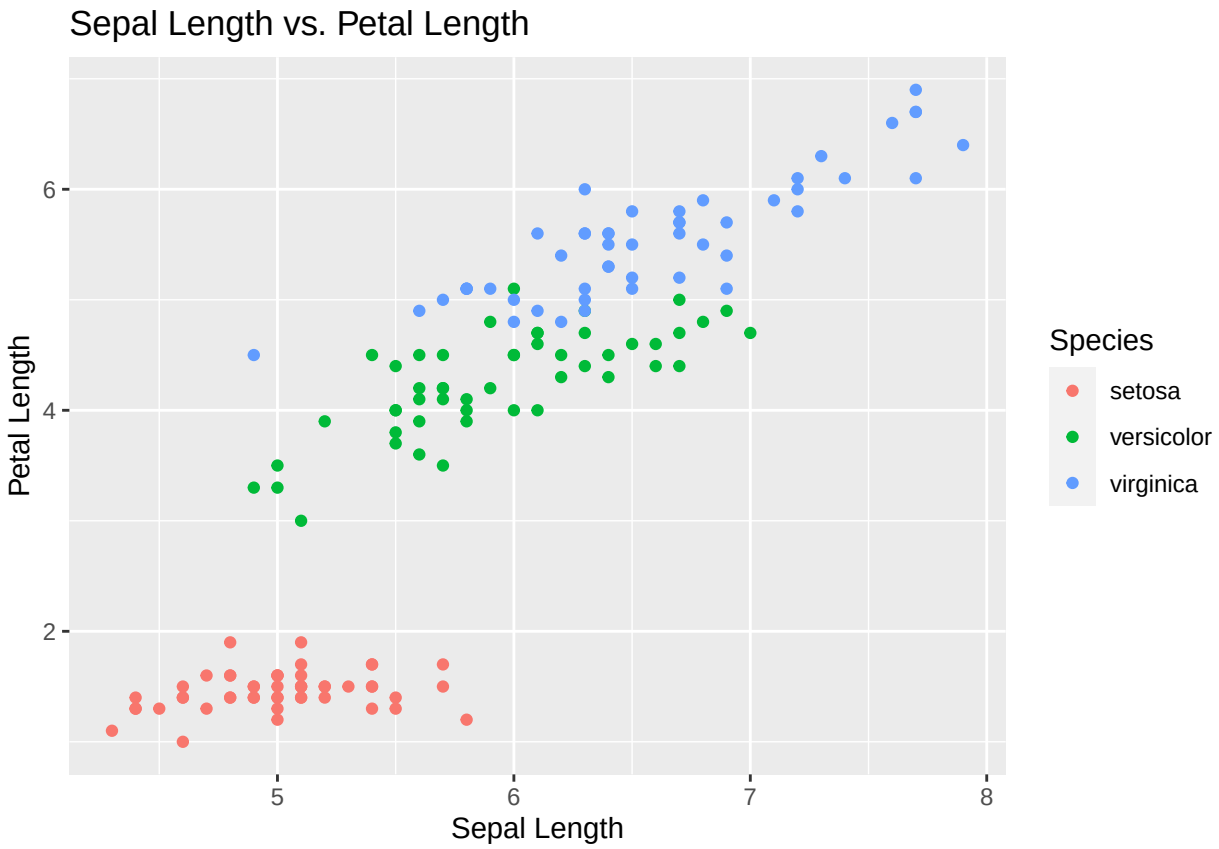
(b) Using your scatterplot from part (a) as a basis, add the species variable by mapping it to color.

```
iris_plot <- ggplot( data=my.iris ) +
  geom_point( aes( x=Sepal.Length, y=Petal.Length, color=Species ) ) +
  labs( title="Sepal Length vs. Petal Length", x="Sepal Length",
        y="Petal Length" )

iris_plot
```



(c) Using your scatterplot from part (a) as a basis, add the species variable by mapping it to color.

```
iris_plot <- ggplot( data=my.iris ) +
  geom_point( aes( x=Sepal.Length, y=Petal.Length, color=Species ) ) +
  labs( title="Sepal Length vs. Petal Length", x="Sepal Length",
        y="Petal Length" )

iris_plot
```



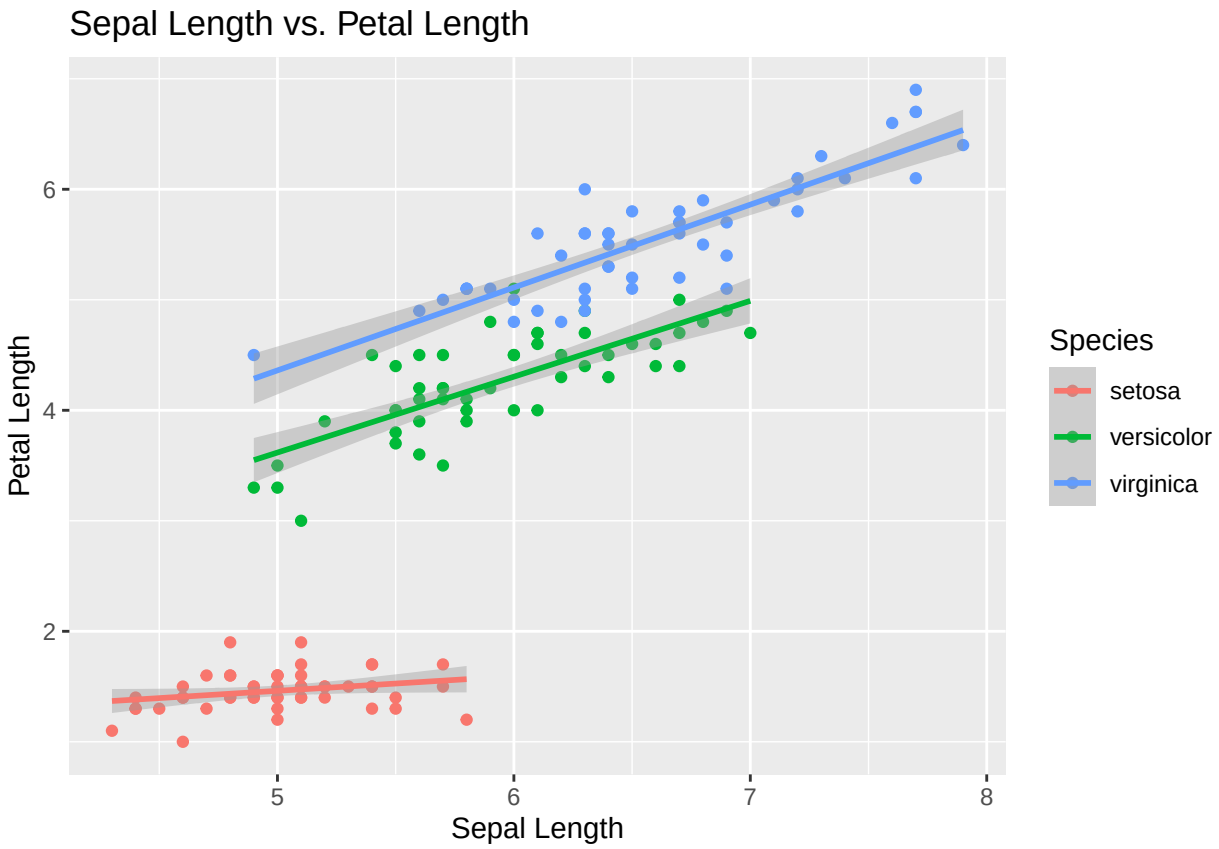
(d) From this, can you see which species has the longest petals and which has the shortest?

From the plot, it looks like the Virginica flower has the longest petals, and the Setosa flower has the shortest petals.

(e) Using your scatterplot from part (c) as a basis, add a line of best fit for each species.

```
iris_plot <- ggplot( data=my.iris ) +
  geom_point( aes( x=Sepal.Length, y=Petal.Length, color=Species ) ) +
  geom_smooth( method="lm", aes( x=Sepal.Length, y=Petal.Length, col=Species ) ) +
  labs( title="Sepal Length vs. Petal Length", x="Sepal Length",
        y="Petal Length" )

iris_plot
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



(f) If I measure a new flower and it has a petal length of 6.5, which species is it most likely to be? If I measure a new flower and it has a sepal length of 4.8, which species is it most likely to be?

A flower with petal length of 6.5 would most likely be a Virginica flower. A flower with a petal length of 4.8 would most likely be a Versicolor flower.

(g) Produce a pairs plot for our data using the `GGally` package.

```
library(GGally)
```

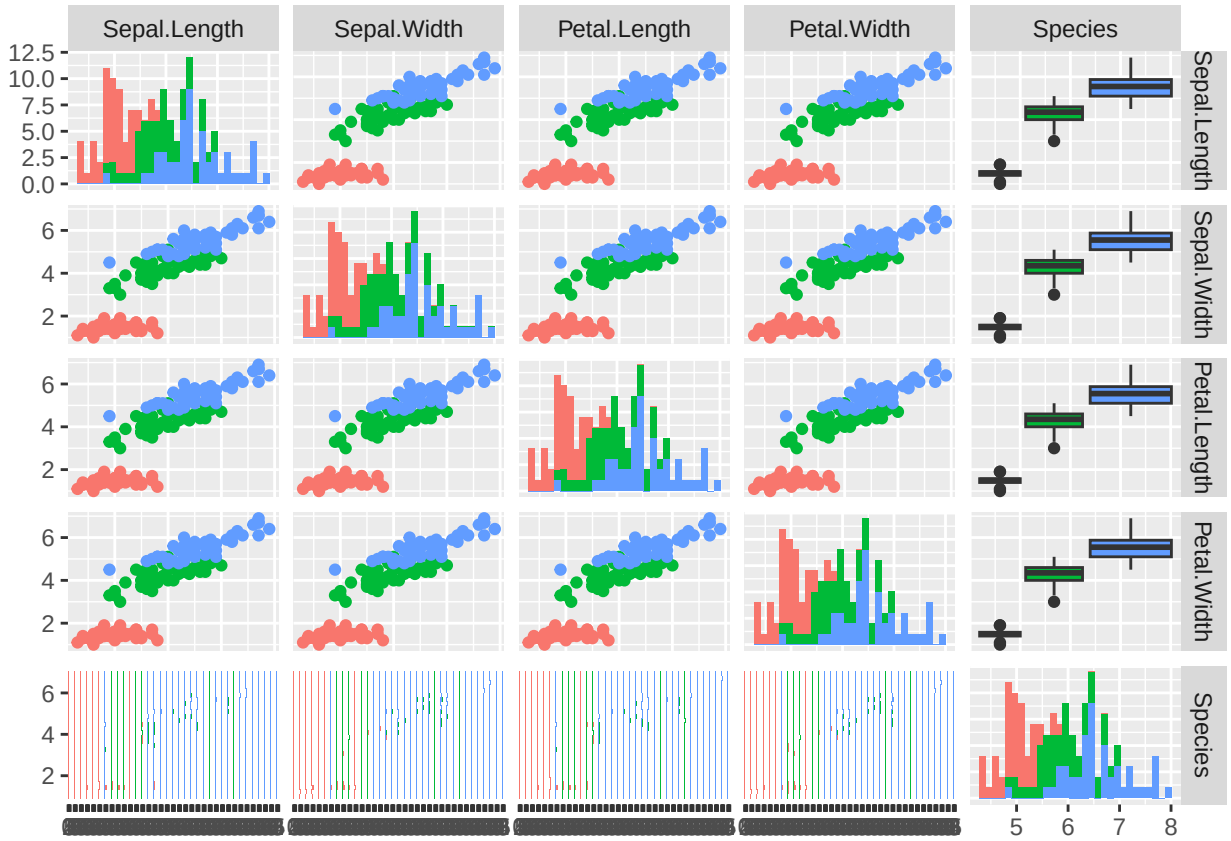
```
## Warning: package 'GGally' was built under R version 4.1.3
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

Look at the help documents (`?ggpairs`) we see if you can work out how to use the ggpairs function to produce a pairs plot with scatterplots on the upper and lower half matrices, and a histogram down the main diagonal.

```
ggpairs( data=my.iris,
         mapping=aes( x=Sepal.Length,
                      y=Petal.Length,
                      color=Species ),
         upper=list( continuous="points" ),
         diag=list( continuous="barDiag" ),
         lower=list( continuous="points" ) )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



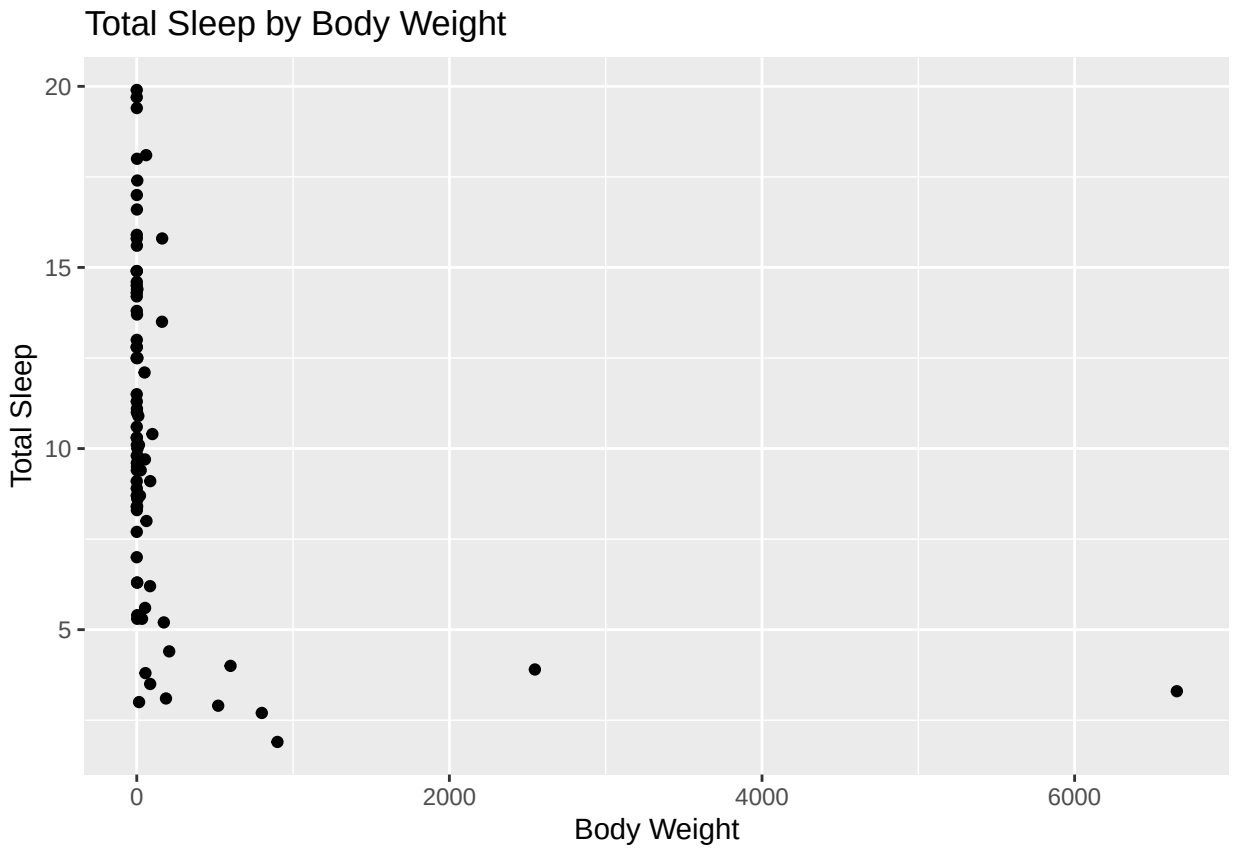
Exercise 3

(a) The final dataset we are going to work with is the `msleep` dataset that describes the sleep characteristics of various mammals. Run the following code block to save a copy of the dataset to a variable called `my.mammals`. The following will also remove some of the missing values from the original dataset so we don't run into any issues. We will learn about the `drop_na` function in the next couple of weeks, but for now you can just run the code block as provided.

```
my.mammals <- msleep
my.mammals <- my.mammals %>% drop_na( sleep_total ) %>% drop_na( bodywt )
```

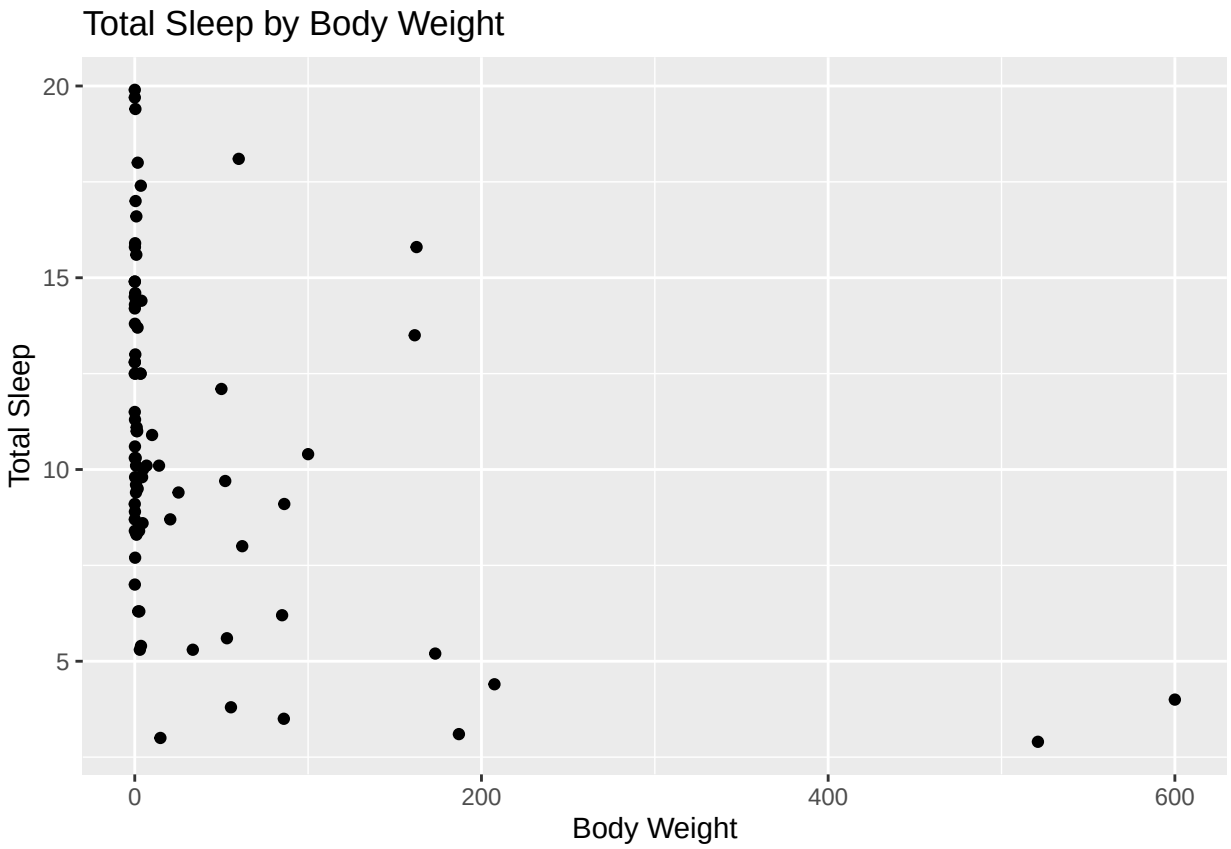
Using this data, produce a scatterplot of sleep total against body weight.

```
mammal_plot <- ggplot( data=my.mammals ) +
  geom_point( aes( x=bodywt, y=sleep_total ) ) +
  labs( title="Total Sleep by Body Weight", x="Body Weight", y="Total Sleep" )

mammal_plot
```



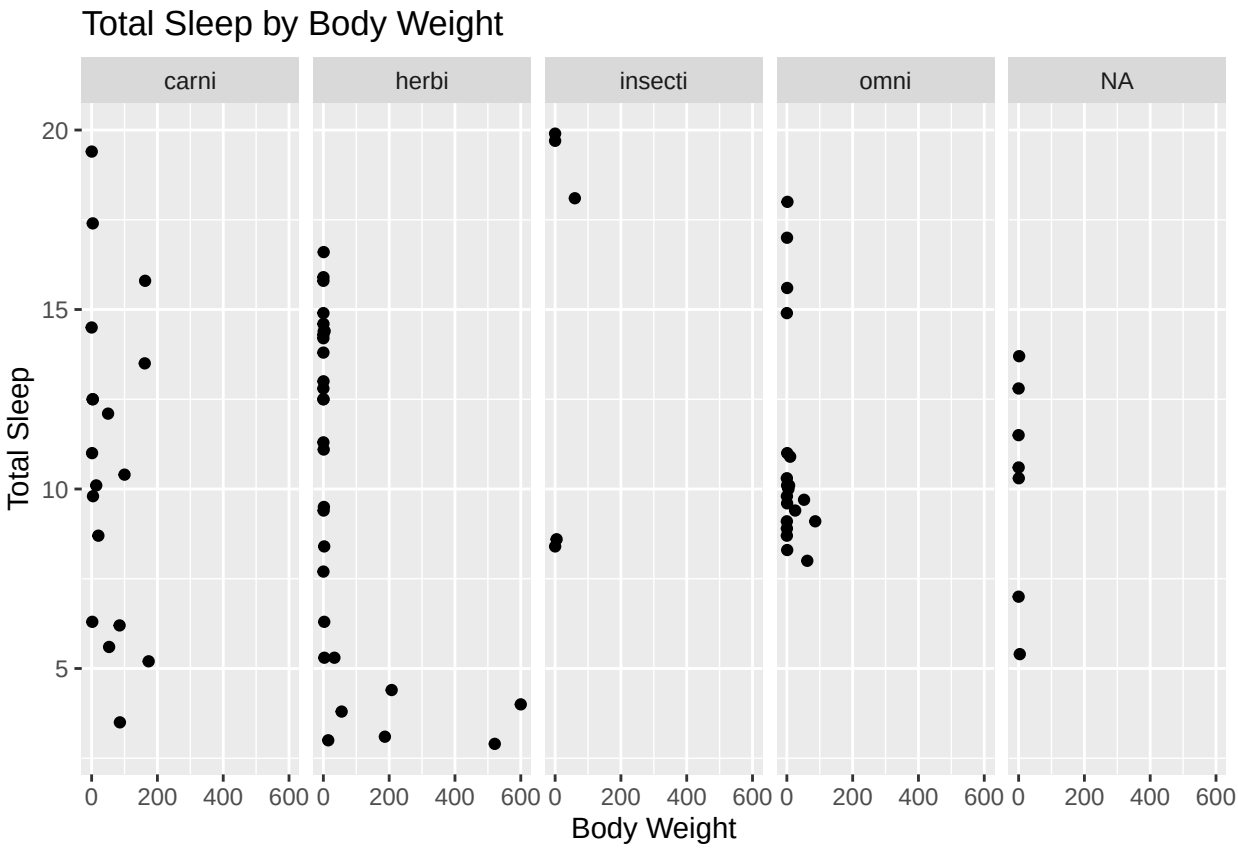
(b) The three largest points seem to be skewing the plot a lot. Remove them and produce the plot again.

```
my.mammals = dplyr::filter( my.mammals, as.numeric(bodywt) < 800 )
mammal_plot <- ggplot( data=my.mammals ) +
  geom_point( aes( x=bodywt, y=sleep_total ) ) +
  labs( title="Total Sleep by Body Weight", x="Body Weight", y="Total Sleep" )

mammal_plot
```



(c) Create a facet grid of scatterplots using the same variables as (a) and (b), but separate the plots by columns mapped to the levels of the `vore` variable.

```
mammal_plot <- ggplot( data=my.mammals ) +
  geom_point( aes( x=bodywt, y=sleep_total ) ) +
  labs( title="Total Sleep by Body Weight", x="Body Weight", y="Total Sleep" ) +
  facet_grid( ~vore )

mammal_plot
```


Total Sleep by Body Weight
carni
herbi
insecti
omni
NA
Total Sleep
20
15
10
5
0
200
400
600
Body Weight