AMAAS: Using the package "ggplot2" for data visualization

Benjamin Utting (bju23@cam.ac.uk)

September 19, 2018

Analytical Methods in Anthropology and Archaeology (AMAAS)

Lent 2018

The following packages are required for running this document: *tidyverse*, *devtools*, *knitr*, *bioanth (available from GitHub)*

Introduction

This R Markdown document is intended to accompany a workshop on using the package "ggplot2" for visualizing data.

ggplot2 is a package developed by Hadley Wickham that is based on the book "The Grammar of Graphics" by Leland Wilkinson. It is centered around the powerful idea that we construct visual representations of data in the same way that we construct sentences. In other words, we can think of plots or graphs as aesthetic and geometric features that combine to convey an appropriate amount of information (not too much or too little) to the viewer.

Here, "aesthetic"" features are defined as attributes such as color, shape, and size whereas "geometric" features are defined as attributes such as points, lines, and bars. ggplot2 users can combine these features together to create both simple and complex data visualizations.

Why use ggplot2?

- ggplot2 is highly versatile
- ggplot2 is more intuitive than the base R plot functions
- The default settings of ggplot2 are much nicer

For this demonstration, we will be using Dr. Benjamin Auerbach's osteometric dataset (available at http://web.utk.edu/~auerbach/DATA.htm). This dataset is automatically loaded with the "bioanth" package that we downloaded from GitHub (above).

- This dataset includes samples of humans throughout the Holocene
- ~1,500 observations from 59 locations
- Dataset includes measurements from the humerus/radius, femur/tibia, and pelvis

```
View(goldman) #looking at data frame
str(goldman) #looking at data types
summary(goldman) #summary statistics for each variable

#For variable name definitions
?goldman

#Where are specimens housed?
summary(goldman$Inst)

#How many of each sex is represented?
summary(goldman$Sex)
```

```
#985 males and 543 females with 10 indeterminates (5 likely male, 5 likely female)
#Important to note that this sample is sex-biased!
```

Today, we will be learning the basics of constructing a plot, as well as several functions, including:

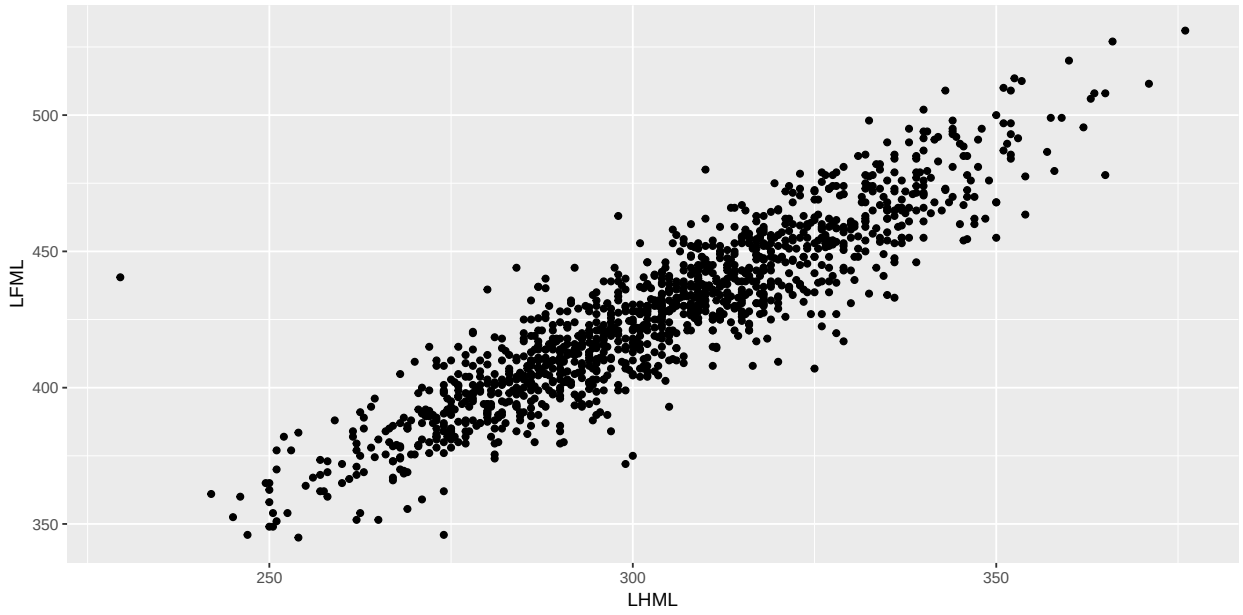
1. "qplot", or quickplot
2. "geom_point" for scatterplots
3. "geom_bar" for barplots
4. "geom_histogram" for histograms
5. "geom_density" for density plots
6. "geom_boxplot" for boxplots

Note that the syntax used for these functions is simply and highly intuitive.

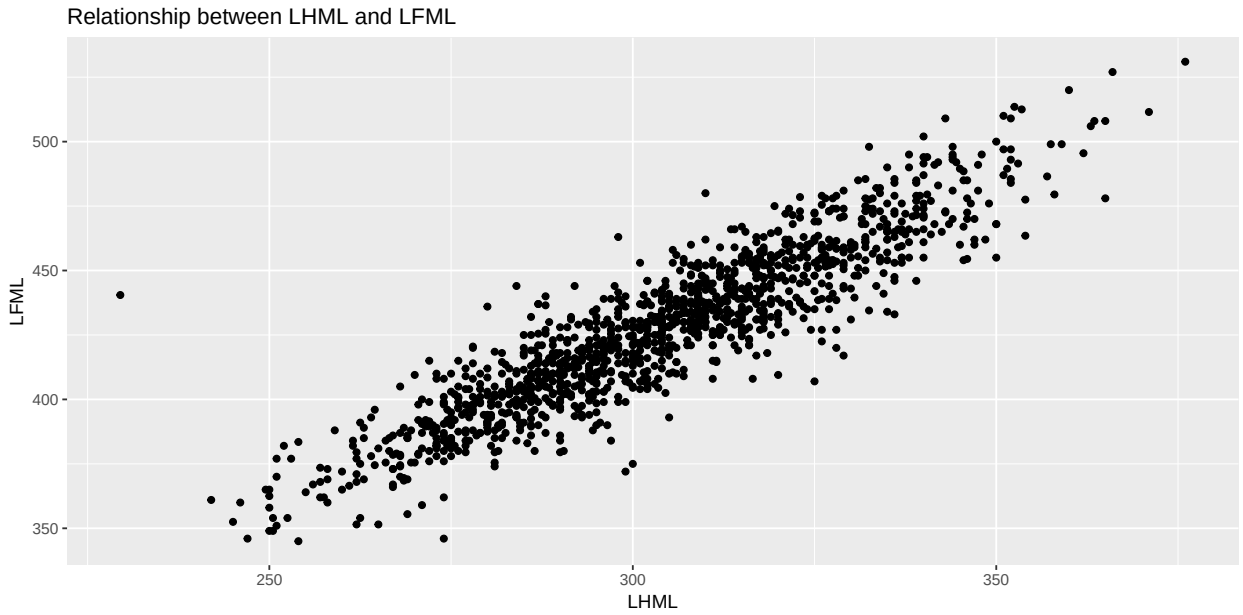
Using 'qplot' and learning the basics

qplot, short for quickplot, is a quick and easy way to visualize data. Here, you can specify one or two variables (aesthetics), the dataframe that the variables come from, and several other parameters for your graph.

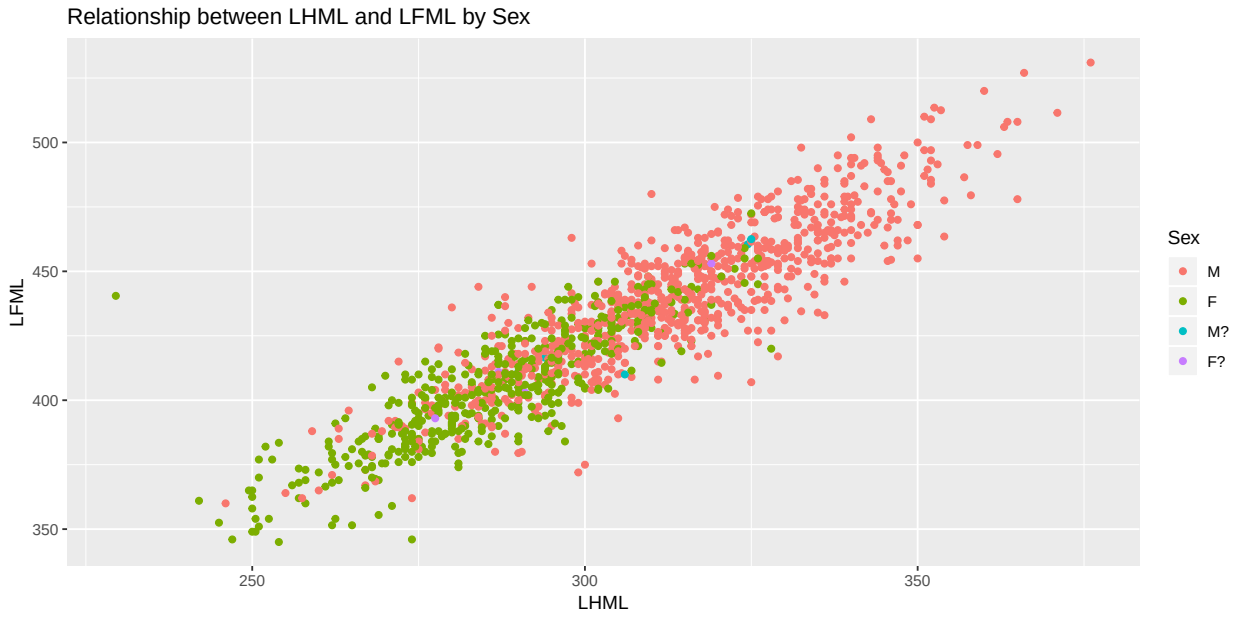
```
#We will begin by plotting the relationship between Left Humerus Maximum Length and Left Femur
#Maximum Length.
qplot(data = goldman,
      x = LHML, y = LFML)
```



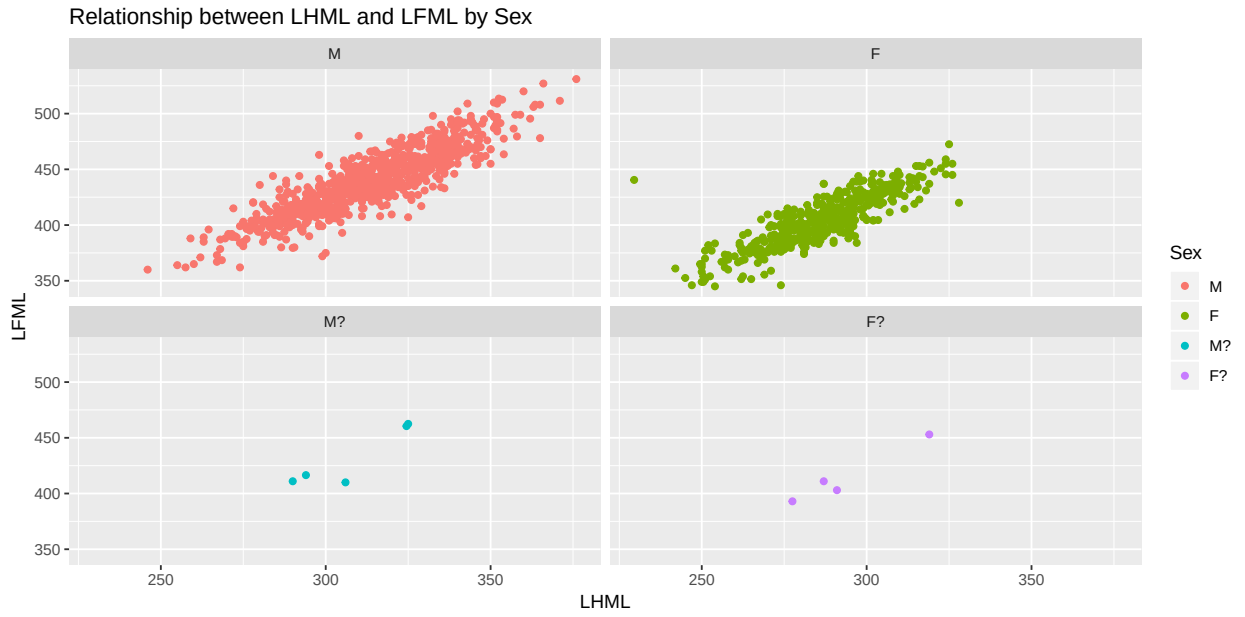

```
#note that qplot(LHML, LFML, data = goldman) will return the same plot

#Adding a title and legend to our plot:
qplot(data = goldman,
      x = LHML, y = LFML) +
      ggtitle("Relationship between LHML and LFML")
```



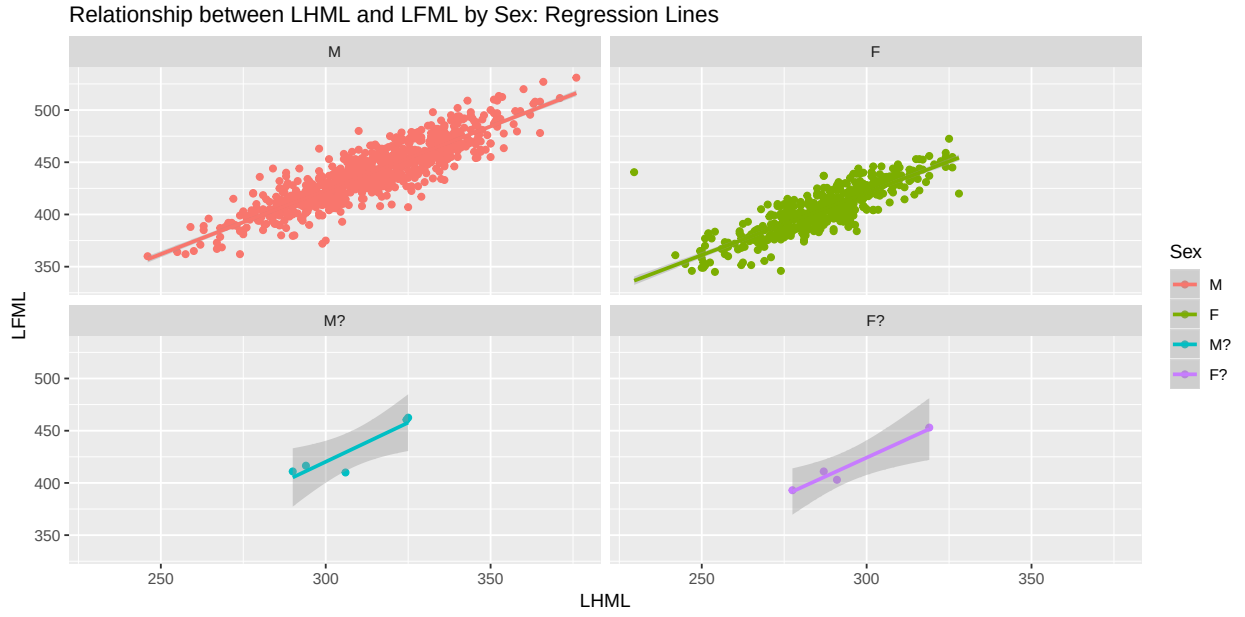

```
#Adding an aesthetic attribute to our plot: color by sex
qplot(data = goldman,
      x = LHML, y = LFML,
      color = Sex) +
      ggtitle("Relationship between LHML and LFML by Sex")
```




```
#Note the automatic legend on the right

#"Faceting"" by a categorical variable
qplot(data = goldman,
      x = LHML, y = LFML,
      color =  Sex,
      facets = ~Sex) +
      ggtitle("Relationship between LHML and LFML by Sex")
```




```
#Adding regression lines
qplot(data = goldman,
      x = LHML, y = LFML,
      color =  Sex,
      facets = ~Sex,
      geom = c("point", "smooth"), method = "lm") +
      ggtitle("Relationship between LHML and LFML by Sex: Regression Lines")
```

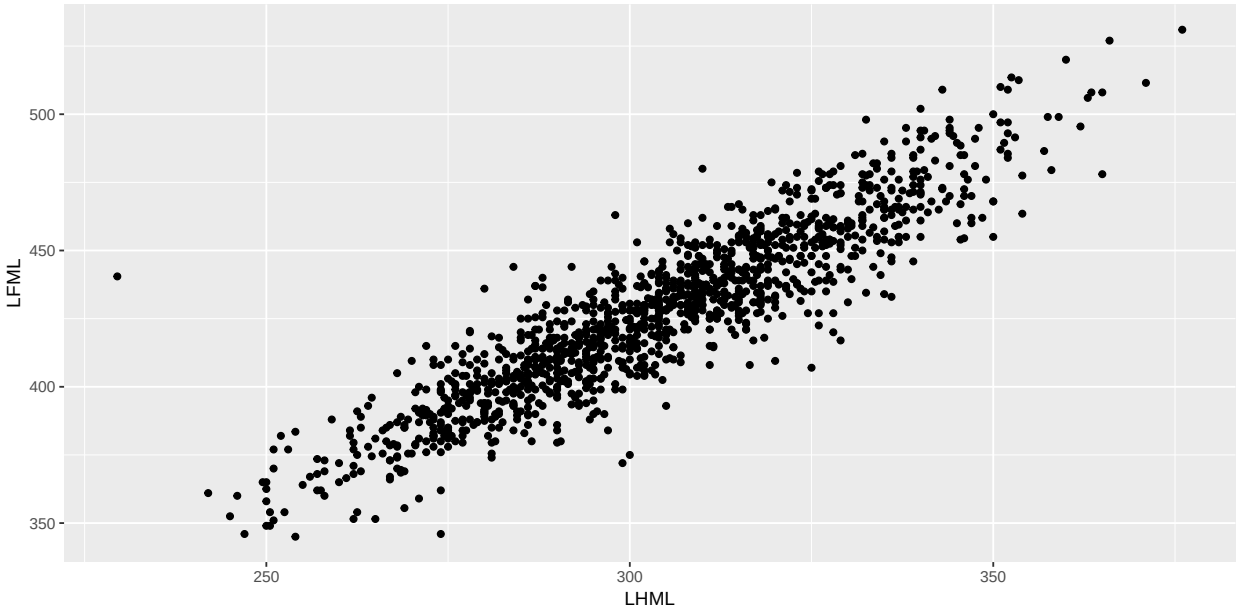


Using the function "ggplot"

In the above section, we used the qplot function to quickly generate a graph. This is fine in many instances, but sometimes, we will need to create more complex visualizations. Here, the function "ggplot" comes into

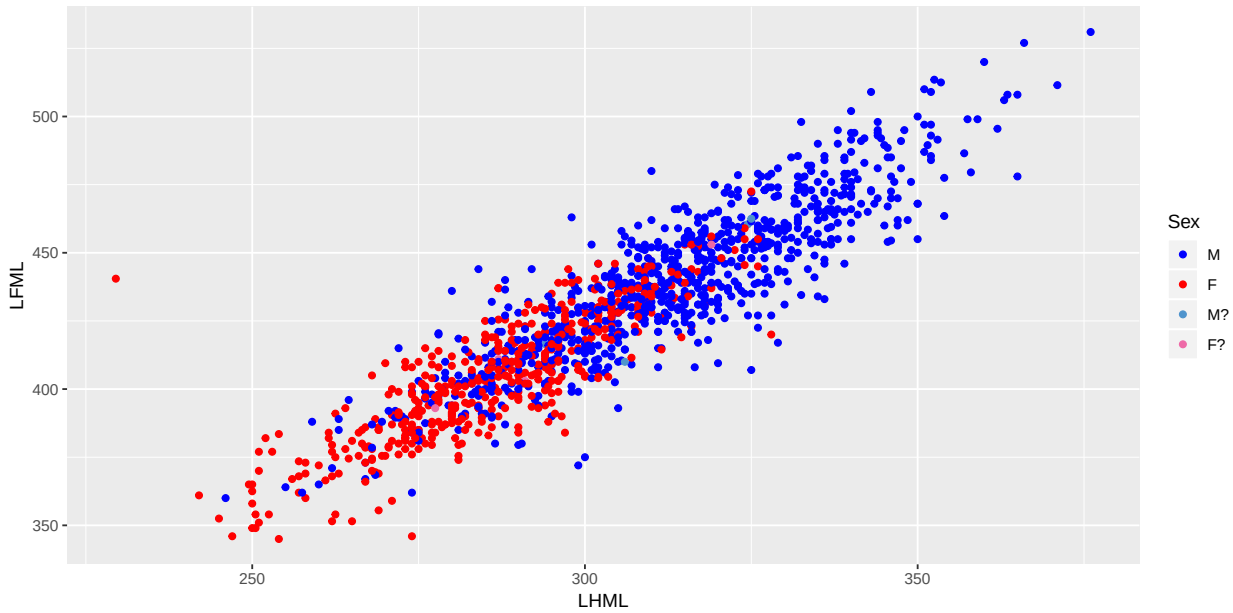
play.

```
#We'll begin with the same plot from above, except we'll use the ggplot() function instead of
  #qplot().
ggplot(goldman, aes(LHML, LFML)) +
  geom_point()
```

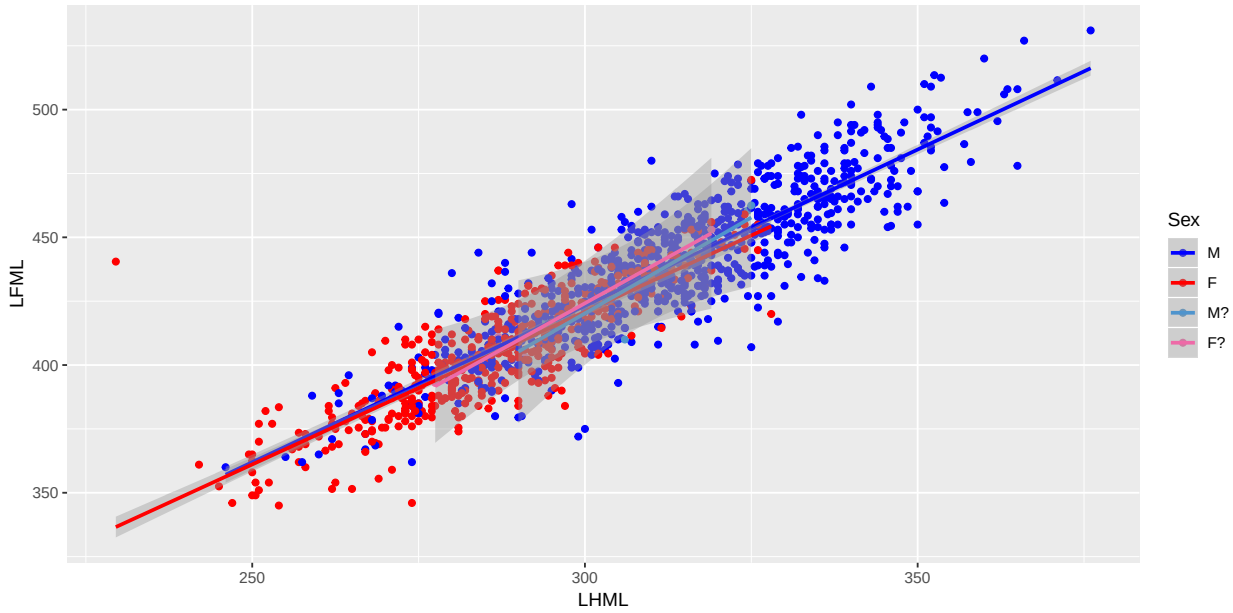



```
#Adding a color to differentiate between M, F, M?, and F?
ggplot(goldman, aes(LHML, LFML, color = Sex)) +
  geom_point()
```


```
#Changing color
ggplot(goldman, aes(LHML, LFML, color = Sex)) +
  geom_point() +
  scale_color_manual("Sex", values = c("M" = "blue", "F" = "red", "M?" = "steelblue3", "F?" = "hotpink2
```

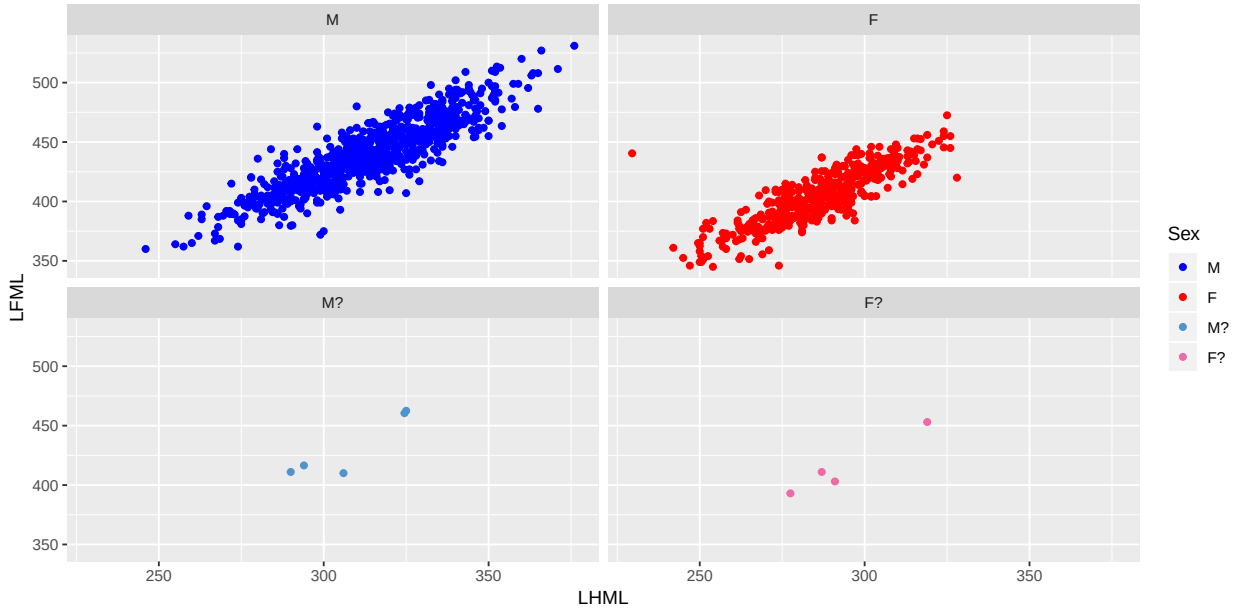



```
#Note that we automatically rearranged the scale as well

#Adding regression lines to single plot
ggplot(goldman, aes(LHML, LFML, color = Sex)) +
  geom_point() +
  scale_color_manual("Sex", values = c("M" = "blue", "F" = "red", "M?" = "steelblue3", "F?" = "hotpink2
  geom_smooth(method = "lm")
```

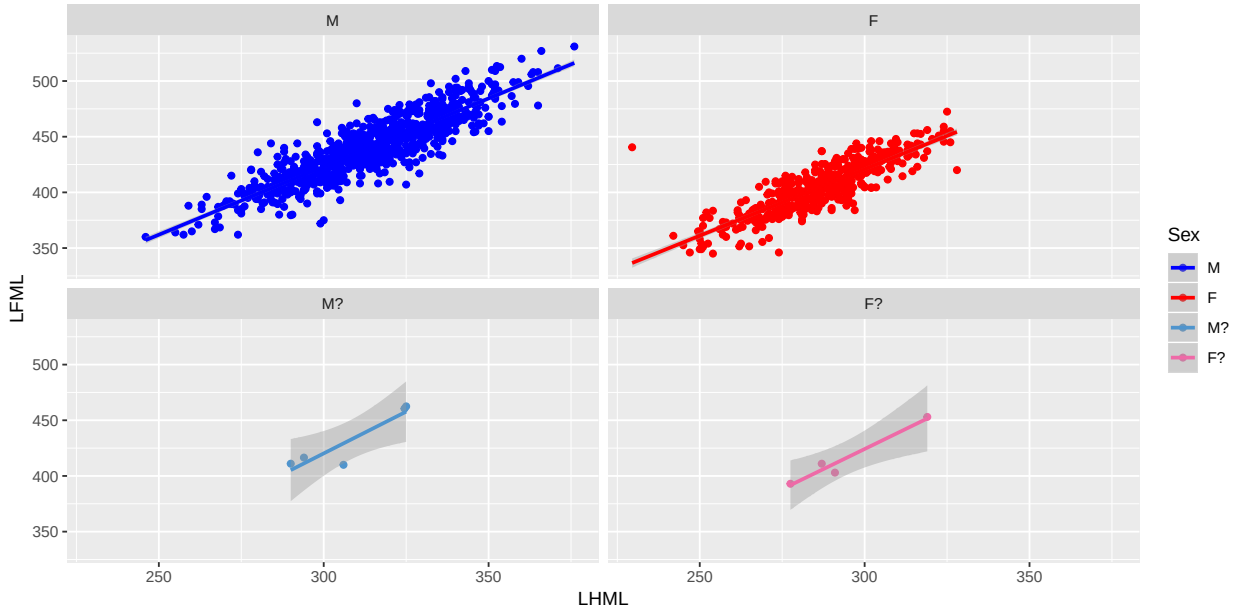



```
#This plot contains too much information and is hard to read - facetting will help us here.

#Adding facets
ggplot(goldman, aes(LHML, LFML, color = Sex)) +
```

```
  geom_point() +
  scale_color_manual("Sex", values = c("M" = "blue", "F" = "red", "M?" = "steelblue3", "F?" = "hotpink2
  facet_wrap(.~Sex) #if we use facet_grid() instead, the graphs will appear in a different order
```

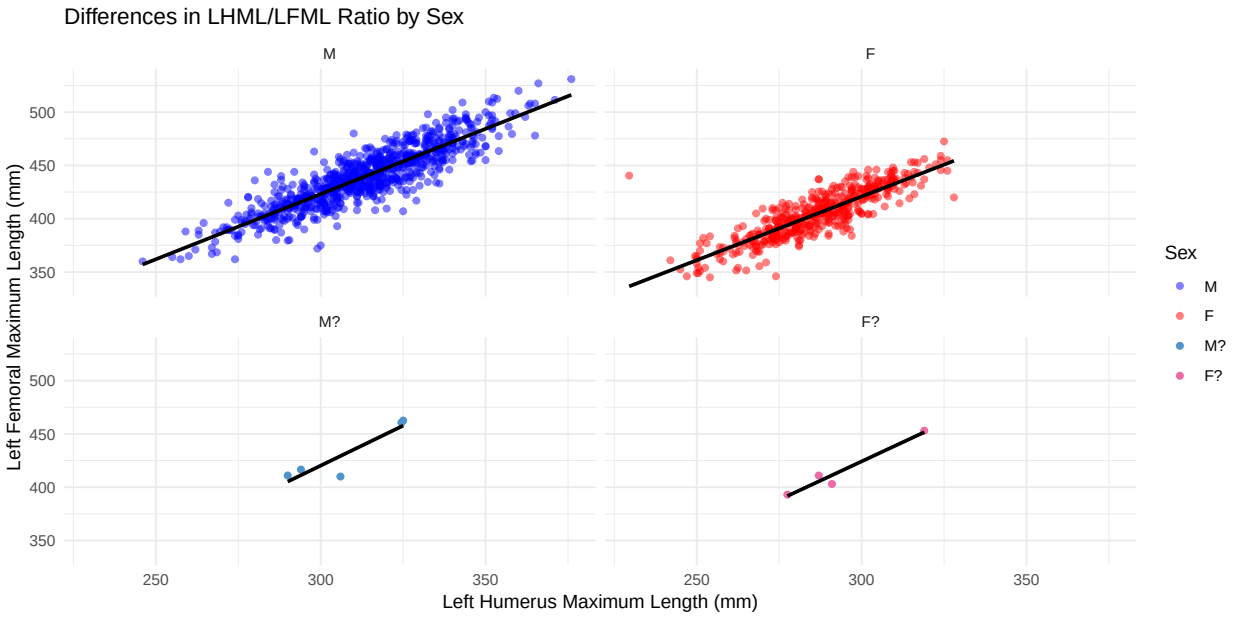



```
#Adding regression lines
ggplot(goldman, aes(LHML, LFML, color = Sex)) +
  geom_point() +
  scale_color_manual("Sex", values = c("M" = "blue", "F" = "red", "M?" = "steelblue3", "F?" = "hotpink2
  facet_wrap(.~Sex) +
  geom_smooth(method = "lm") #the argument "se = FALSE" will remove confidence intervals
```




```
#This plot is a bit clearer. Let's add some finishing touches (title, axis names, theme, changing
#point color strength)
ggplot(goldman, aes(LHML, LFML, color = Sex)) +
```

```
  geom_point() +
  scale_color_manual("Sex", values = c("M" = alpha("blue", 0.5), "F" = alpha("red", 0.5), "M?" = "steell
  facet_wrap(.~Sex) +
  geom_smooth(method = "lm", color = "black", se = FALSE) +
  ggtitle("Differences in LHML/LFML Ratio by Sex") +
  xlab("Left Humerus Maximum Length (mm)") +
  ylab("Left Femoral Maximum Length (mm)") +
  theme_minimal()
```



Learning some syntax for different types of plots

In the previous section, we used "geom_point()" and "geom_smooth" to create our plot. In ggplot2, there are many more geometric arguments. Here is a list of some common geoms:

1. geom_point(): for scatterplots
2. geom_jitter(): adds a small amount of random noise to the plot. Used for large datasets.
3. geom_abline(): a line with slope and intercept
4. geom_vline(): a line with an x-intercept (vertical line)
5. geom_hline(): a line with a y-intercept (horizontal line)
6. geom_bar(): for bar charts (basically interchangeable with 'geom_histogram()')
7. geom_boxplot(): for boxplots
8. geom_density(): for density plots
9. geom_area(): similar to geom_density(), but colors in the area underneath the line
10. geom_smooth(): for adding patterns. Highly flexible.
11. geom_dotplot(): for creating plots where one dot represents one observation. Similar to geom_histogram()

A complete list of geoms can be found online.

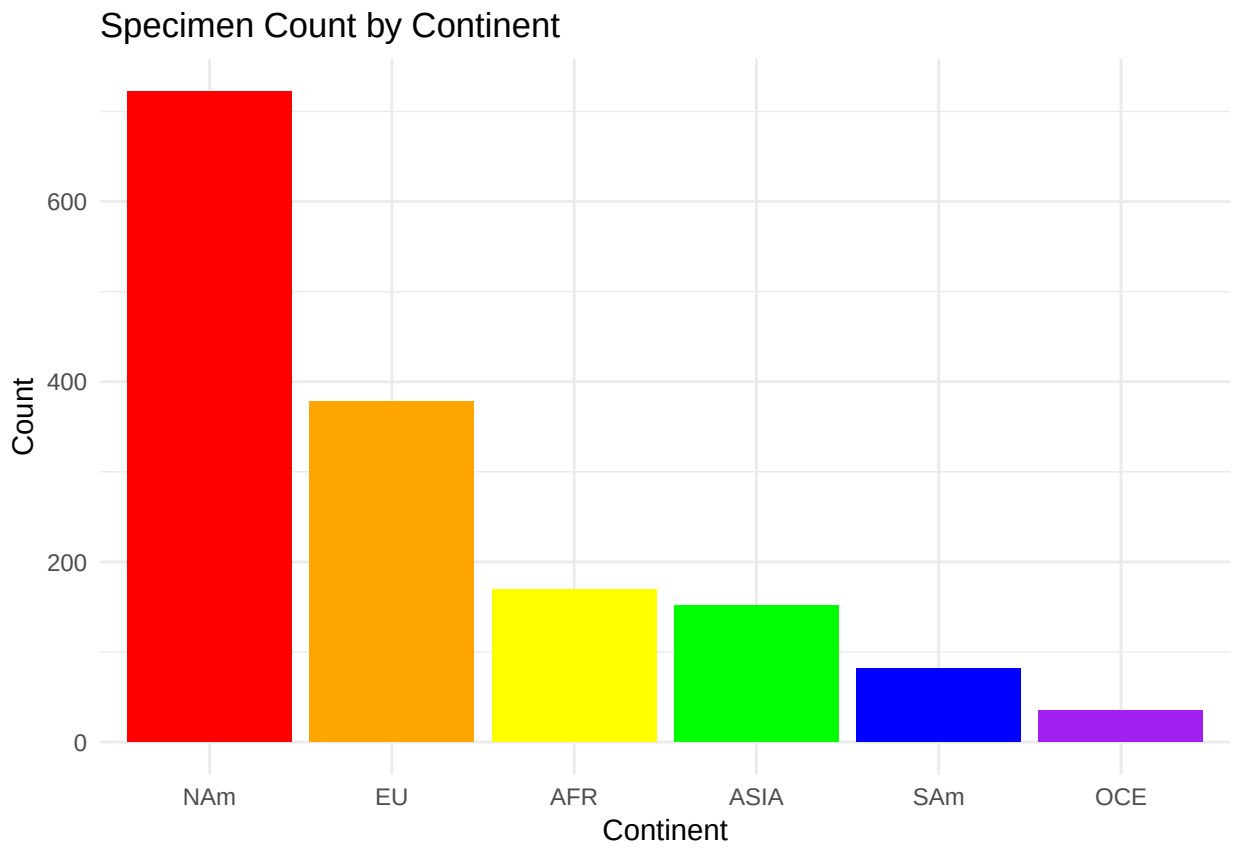
Barplots

Here, we'll create a barplot showing which continent our specimens come from. Note that there isn't a column containing this information in our goldman dataset, so we'll have to create one ourselves. The first step (creating separated_location) is discussed in higher detail in the "dplyr" document.

```
separated_location <- goldman %>%
  separate(Location, into = c("Region", "Subregion", "Country"), sep = ",", fill = "left")

#This function returned "character" class data, so we must transform it into "factor"
  #data in order to explore it
separated_location$Region <- as.factor(separated_location$Region)
separated_location$Subregion <- as.factor(separated_location$Subregion)
separated_location$Country <- as.factor(separated_location$Country)


#Now that we have a new data frame to work with, we need to add a column to tell ggplot which continent

EU <- c(" United Kingdom", "Austria", "Belgium", "France", "Germany", "Greenland", "Italy")
NAm <- c(" United States")
SAm <- c("Argentina", "Chile", "Ecuador", "Peru")
AFR <- c("Canary Islands", "Democratic Republic of the Congo", "Egypt", "Madagascar", "South Africa", "
ASIA <- c("Andaman Islands", "China", "Indonesia", "Japan", "Malaysia", "Philippine Islands", "Russia")
OCE <- c("Australia", "Papua New Guinea", "Solomon Islands", "Tasmania")

separated_location <- separated_location %>%
    mutate(Continent = factor(case_when(Country %in% EU ~ "EU",
                                        Country %in% NAm ~ "NAm",
                                        Country %in% SAm ~ "SAm",
                                        Country %in% AFR ~ "AFR",
                                        Country %in% ASIA ~ "ASIA",
                                        Country %in% OCE ~ "OCE")))
summary(separated_location$Continent)
```

```
##  AFR ASIA   EU  NAm  OCE  SAm
##  169  152  378  722   35   82
```

```
#Please note here that working with data can be highly frustrating and messy. There is a space before "

#First, reorder the levels of "Continent" by frequency (optional)
separated_location$Continent <- factor(separated_location$Continent, levels = c("NAm", "EU", "AFR", "AS

#Now we can create our barplot
ggplot(separated_location, aes(Continent, fill = Continent)) +
  geom_bar() +
  scale_fill_manual(values = c("NAm" = "red", "EU" = "orange", "AFR" = "yellow", "ASIA" = "green", "SAm
  guides(fill = FALSE) +
  xlab("Continent") +
  ylab("Count") +
  ggtitle("Specimen Count by Continent") +
  theme_minimal()
```


Specimen Count by Continent
Count
600
400
200
0
NAm
EU
AFR
ASIA
SAm
OCE
Continent