Exam1

Maria

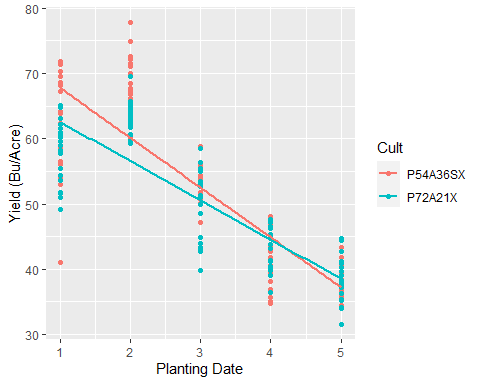
2023-03-07

### Ex. 4

The ‘ggplot’ works in layers. A layer is a combination of data, stat and/or geom. Layering is the process of adding multiple layers (graphical elements) to a graph. You can add layers using the ‘+’ operator in ggplot and it is placed at the end of each line of the code containing a layer. For example, we can add points ‘geom\_point()’ and lines ‘geom\_smooth()’ to a plot by adding them (‘+’) as separate layers. Scales control the mapping between data and aesthetics, for instance the size, type, or color of points in a plot. We can customize the scales using the ‘scale\_ function’ in ggplot. For example, the ‘scale\_color\_gradient()’ function changes the color scale. Themes customize the overall appearance of the plot, such as titles, background color, gridlines, legends, etc. You can customize the theme using the ‘theme\_ function’ in ggplot. For example, ‘theme\_grey()’ results in theme with a grey background and ‘theme\_bw()’ a theme with a white background. A facet breaks a plot into panels. Faceting is the process of dividing the data into subsets, based on the values of a categorical variable, then generates separated plots for each subset. You can create facets using the ‘facet\_ wrap(~variablename)’ in ggplot.

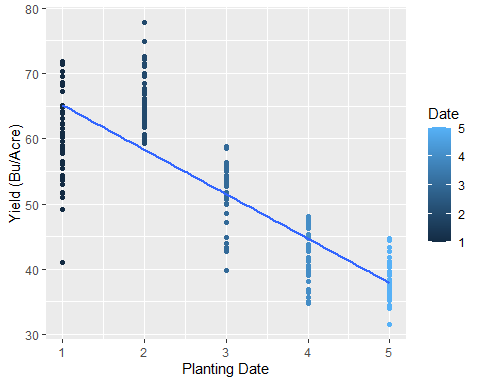
library (ggplot2)  
plantpop <- read.csv("PlantPop.csv")  
  
##Layering##  
  
ggplot(plantpop, aes (x=Date, y=Yield, color=Cult)) +  
 geom\_point() +  
 geom\_smooth (method = lm, se = FALSE) +  
 ylab ("Yield (Bu/Acre)") +  
 xlab ("Planting Date")

## `geom\_smooth()` using formula = 'y ~ x'

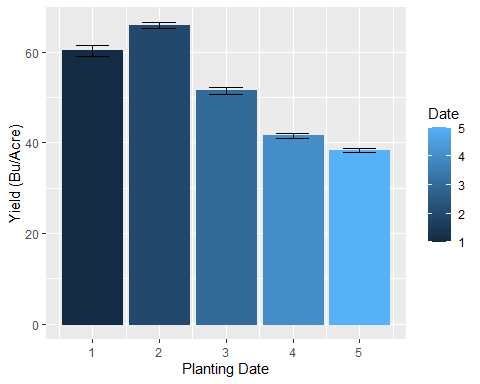


##Scales##  
  
ggplot(plantpop, aes (x=Date, y=Yield)) +  
 geom\_point(aes(color=Date)) +  
 geom\_smooth (method = lm, se = FALSE) +  
 ylab ("Yield (Bu/Acre)") +  
 xlab ("Planting Date")

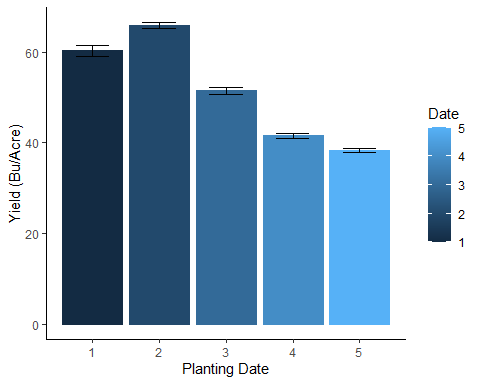
## `geom\_smooth()` using formula = 'y ~ x'



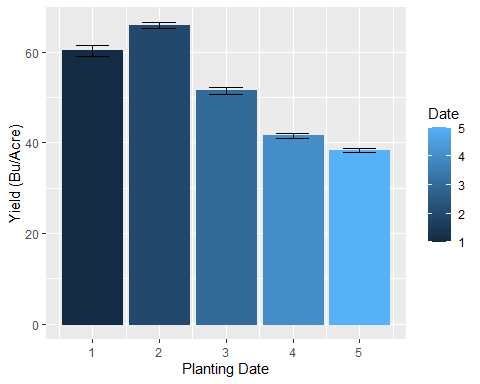
ggplot (plantpop, aes (x=Date, y=Yield)) +   
 stat\_summary(fun=mean, geom="bar", position = "dodge", aes(color=Date, fill=Date)) +  
 stat\_summary(fun.data = mean\_se, geom = "errorbar", position = position\_dodge(width=0.9), width = 0.5) +  
 ylab ("Yield (Bu/Acre)") +  
 xlab ("Planting Date")



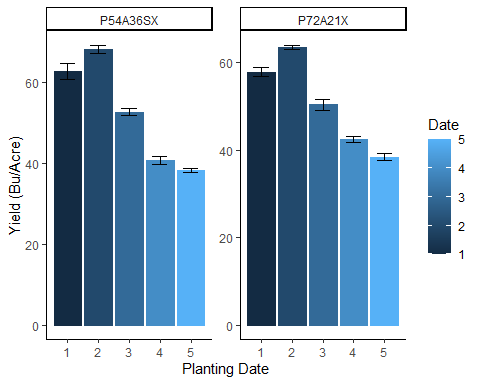
##Themes##  
  
ggplot (plantpop, aes (x=Date, y=Yield)) +   
 stat\_summary(fun=mean, geom="bar", position = "dodge", aes(color=Date, fill=Date)) +  
 stat\_summary(fun.data = mean\_se, geom = "errorbar", position = "dodge", width = 0.5) +  
 ylab ("Yield (Bu/Acre)") +  
 xlab ("Planting Date") +  
 theme\_classic()



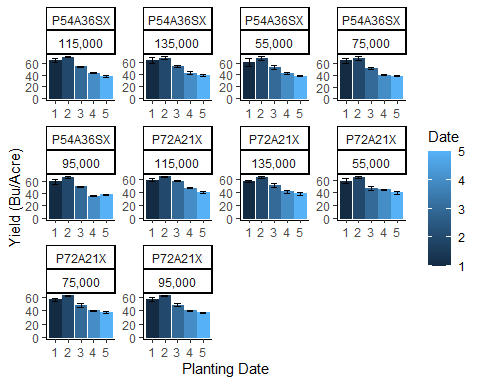
ggplot (plantpop, aes (x=Date, y=Yield)) +   
 stat\_summary(fun=mean, geom="bar", position = "dodge", aes(color=Date, fill=Date)) +  
 stat\_summary(fun.data = mean\_se, geom = "errorbar", position = "dodge", width = 0.5) +  
 ylab ("Yield (Bu/Acre)") +  
 xlab ("Planting Date") +  
 theme\_gray()



##Facets##  
  
ggplot (plantpop, aes (x=Date, y=Yield)) +   
 stat\_summary(fun=mean, geom="bar", position = "dodge", aes(color=Date, fill=Date)) +  
 stat\_summary(fun.data = mean\_se, geom = "errorbar", position = "dodge", width = 0.5) +  
 ylab ("Yield (Bu/Acre)") +  
 xlab ("Planting Date") +  
 theme\_classic() +  
 facet\_wrap(~Cult, scale = "free")



ggplot (plantpop, aes (x=Date, y=Yield)) +   
 stat\_summary(fun=mean, geom="bar", position = "dodge", aes(color=Date, fill=Date)) +  
 stat\_summary(fun.data = mean\_se, geom = "errorbar", position = "dodge", width = 0.5) +  
 ylab ("Yield (Bu/Acre)") +  
 xlab ("Planting Date") +  
 theme\_classic() +  
 facet\_wrap(~Cult\*Density, scale = "free")



### Ex. 5

1. Vector, matrix, and dataframe are all data structure that can be used in R. The vector is the most basic R data objects. It is a one-dimensional display of elements of the same data type (numeric, character, or logical). It can be thought of a list of elements saved together as an object. Vectors are used to create dataframes and matrixes. A matrix, differently from the vector, is a two-dimensional array of elements. It is an atomic object that can contain only one type of data class (numeric, character, or logical). It is created using a vector input and it has fixed number of columns and rows. It can be thought of as a table of elements. A matrix can be treated as a vector if we have a single column or row. A dataframe is very similar to a matrix, it is also a two-dimensional array of elements, but differently it can contain multiple data types together. A dataframe is a list of vectors with equal length where each columns can be a different data types. It can be thought of as a spreadsheet.

subset1 <- ToothGrowth[ToothGrowth$supp == "VC", ]  
subset1

## len supp dose  
## 1 4.2 VC 0.5  
## 2 11.5 VC 0.5  
## 3 7.3 VC 0.5  
## 4 5.8 VC 0.5  
## 5 6.4 VC 0.5  
## 6 10.0 VC 0.5  
## 7 11.2 VC 0.5  
## 8 11.2 VC 0.5  
## 9 5.2 VC 0.5  
## 10 7.0 VC 0.5  
## 11 16.5 VC 1.0  
## 12 16.5 VC 1.0  
## 13 15.2 VC 1.0  
## 14 17.3 VC 1.0  
## 15 22.5 VC 1.0  
## 16 17.3 VC 1.0  
## 17 13.6 VC 1.0  
## 18 14.5 VC 1.0  
## 19 18.8 VC 1.0  
## 20 15.5 VC 1.0  
## 21 23.6 VC 2.0  
## 22 18.5 VC 2.0  
## 23 33.9 VC 2.0  
## 24 25.5 VC 2.0  
## 25 26.4 VC 2.0  
## 26 32.5 VC 2.0  
## 27 26.7 VC 2.0  
## 28 21.5 VC 2.0  
## 29 23.3 VC 2.0  
## 30 29.5 VC 2.0

subset1 <- subset(ToothGrowth, (supp == "VC"))  
subset1

## len supp dose  
## 1 4.2 VC 0.5  
## 2 11.5 VC 0.5  
## 3 7.3 VC 0.5  
## 4 5.8 VC 0.5  
## 5 6.4 VC 0.5  
## 6 10.0 VC 0.5  
## 7 11.2 VC 0.5  
## 8 11.2 VC 0.5  
## 9 5.2 VC 0.5  
## 10 7.0 VC 0.5  
## 11 16.5 VC 1.0  
## 12 16.5 VC 1.0  
## 13 15.2 VC 1.0  
## 14 17.3 VC 1.0  
## 15 22.5 VC 1.0  
## 16 17.3 VC 1.0  
## 17 13.6 VC 1.0  
## 18 14.5 VC 1.0  
## 19 18.8 VC 1.0  
## 20 15.5 VC 1.0  
## 21 23.6 VC 2.0  
## 22 18.5 VC 2.0  
## 23 33.9 VC 2.0  
## 24 25.5 VC 2.0  
## 25 26.4 VC 2.0  
## 26 32.5 VC 2.0  
## 27 26.7 VC 2.0  
## 28 21.5 VC 2.0  
## 29 23.3 VC 2.0  
## 30 29.5 VC 2.0

subset2 <- ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == 0.5, ]  
subset2

## len supp dose  
## 1 4.2 VC 0.5  
## 2 11.5 VC 0.5  
## 3 7.3 VC 0.5  
## 4 5.8 VC 0.5  
## 5 6.4 VC 0.5  
## 6 10.0 VC 0.5  
## 7 11.2 VC 0.5  
## 8 11.2 VC 0.5  
## 9 5.2 VC 0.5  
## 10 7.0 VC 0.5

subset2 <- subset(ToothGrowth, (supp == "VC" & dose == 0.5))  
subset2

## len supp dose  
## 1 4.2 VC 0.5  
## 2 11.5 VC 0.5  
## 3 7.3 VC 0.5  
## 4 5.8 VC 0.5  
## 5 6.4 VC 0.5  
## 6 10.0 VC 0.5  
## 7 11.2 VC 0.5  
## 8 11.2 VC 0.5  
## 9 5.2 VC 0.5  
## 10 7.0 VC 0.5

subset3 <- ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == 0.5, c("len")]  
subset3

## [1] 4.2 11.5 7.3 5.8 6.4 10.0 11.2 11.2 5.2 7.0

subset3 <- subset(ToothGrowth, supp == "VC" & dose == 0.5, select = "len")  
subset3

## len  
## 1 4.2  
## 2 11.5  
## 3 7.3  
## 4 5.8  
## 5 6.4  
## 6 10.0  
## 7 11.2  
## 8 11.2  
## 9 5.2  
## 10 7.0