Coding Notes Data Visualization 1

Mamata K C

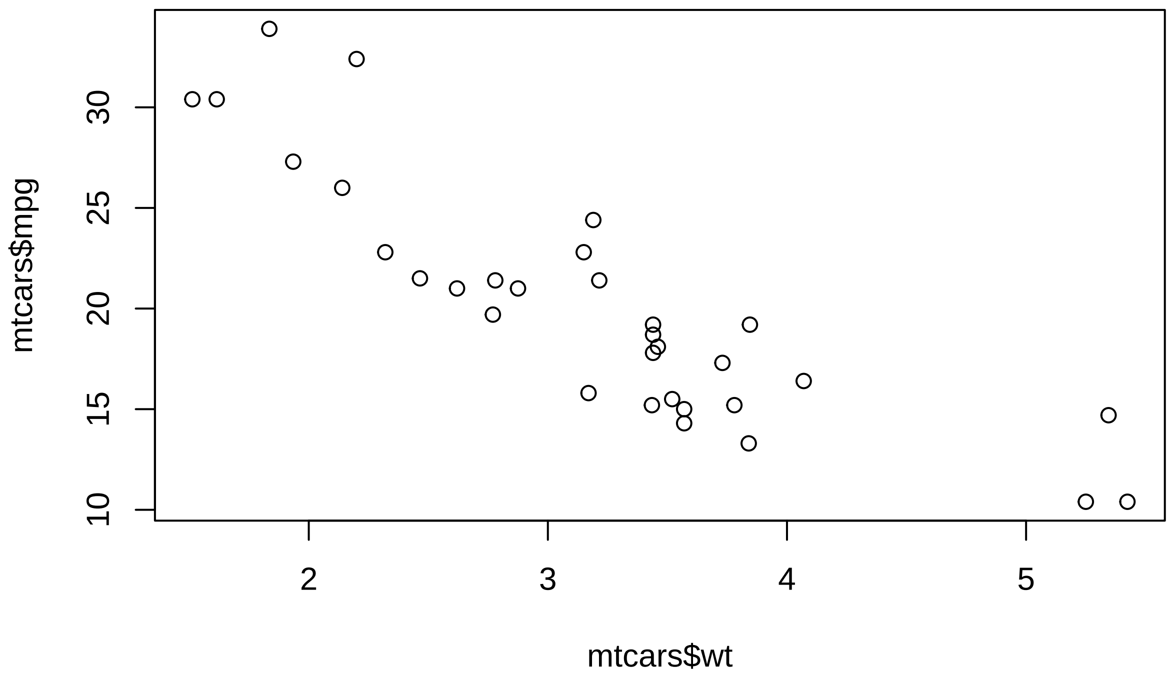
2025-02-13

Basic data visualization in R

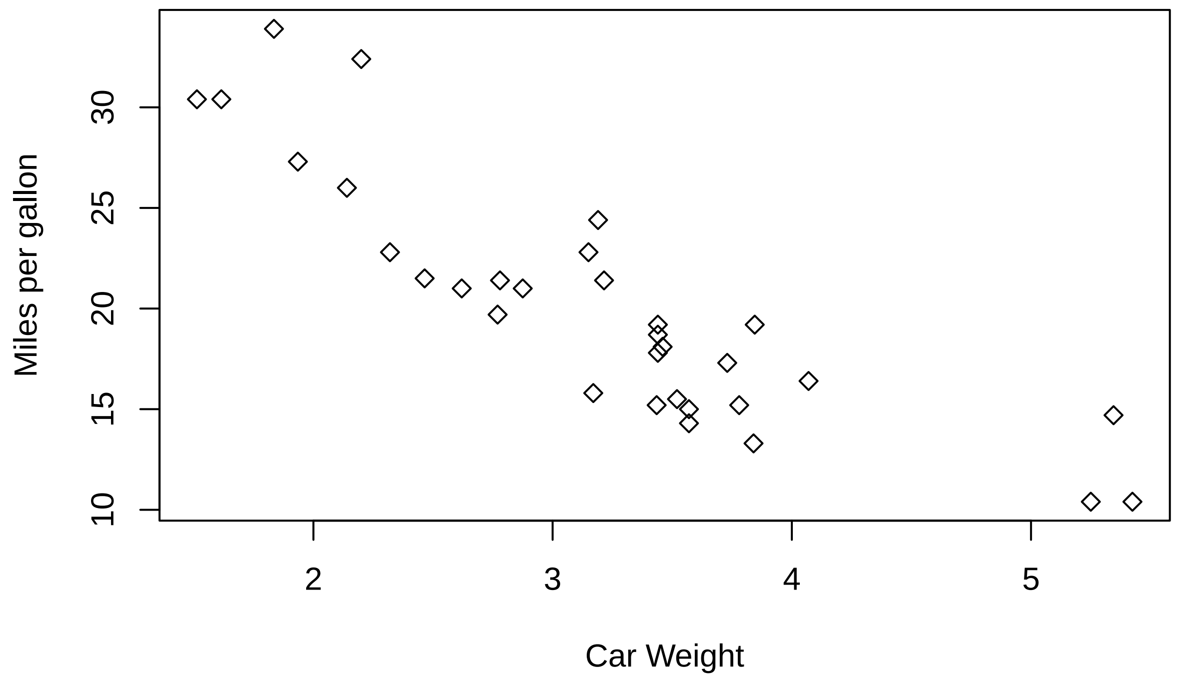
```
#new datasetmtcars
data("mtcars")

#To look at structure of data
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

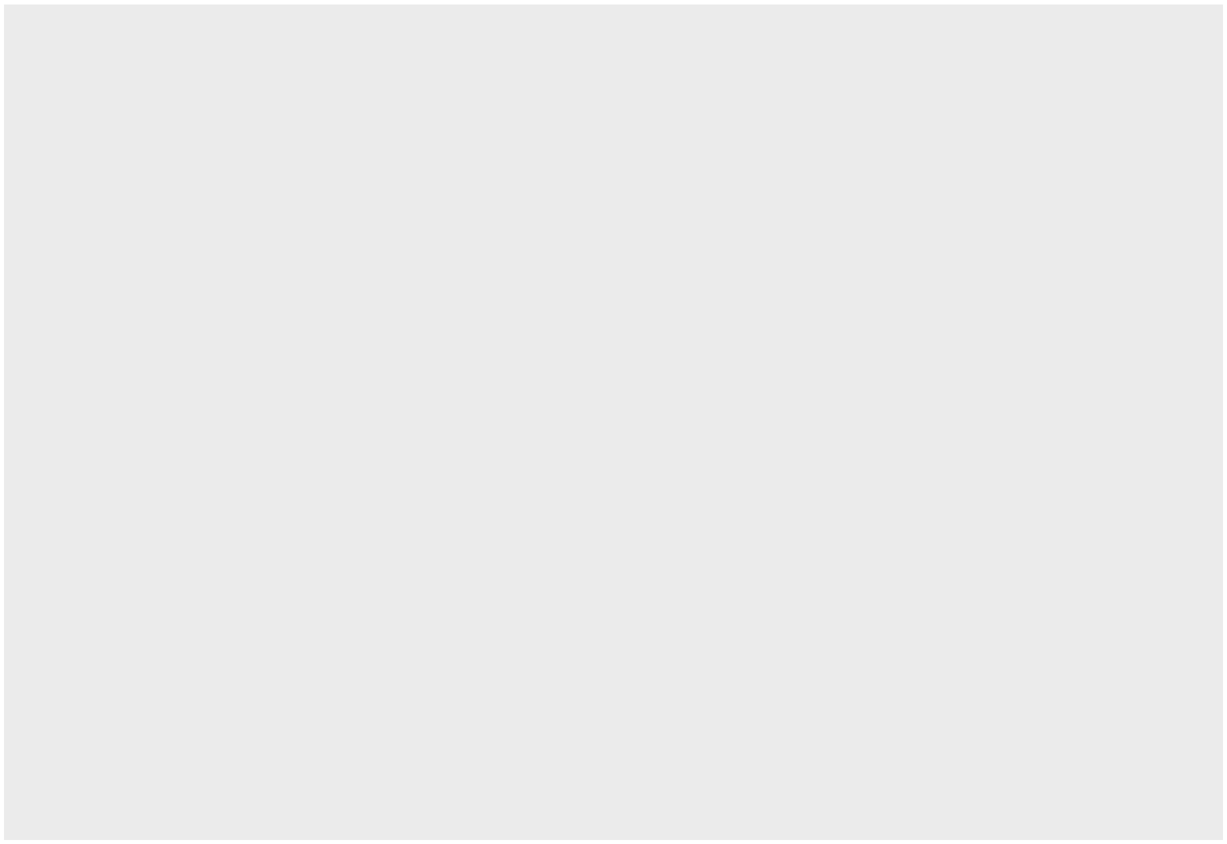
```
#scatterplot
plot(mtcars$wt,mtcars$mpg)
```



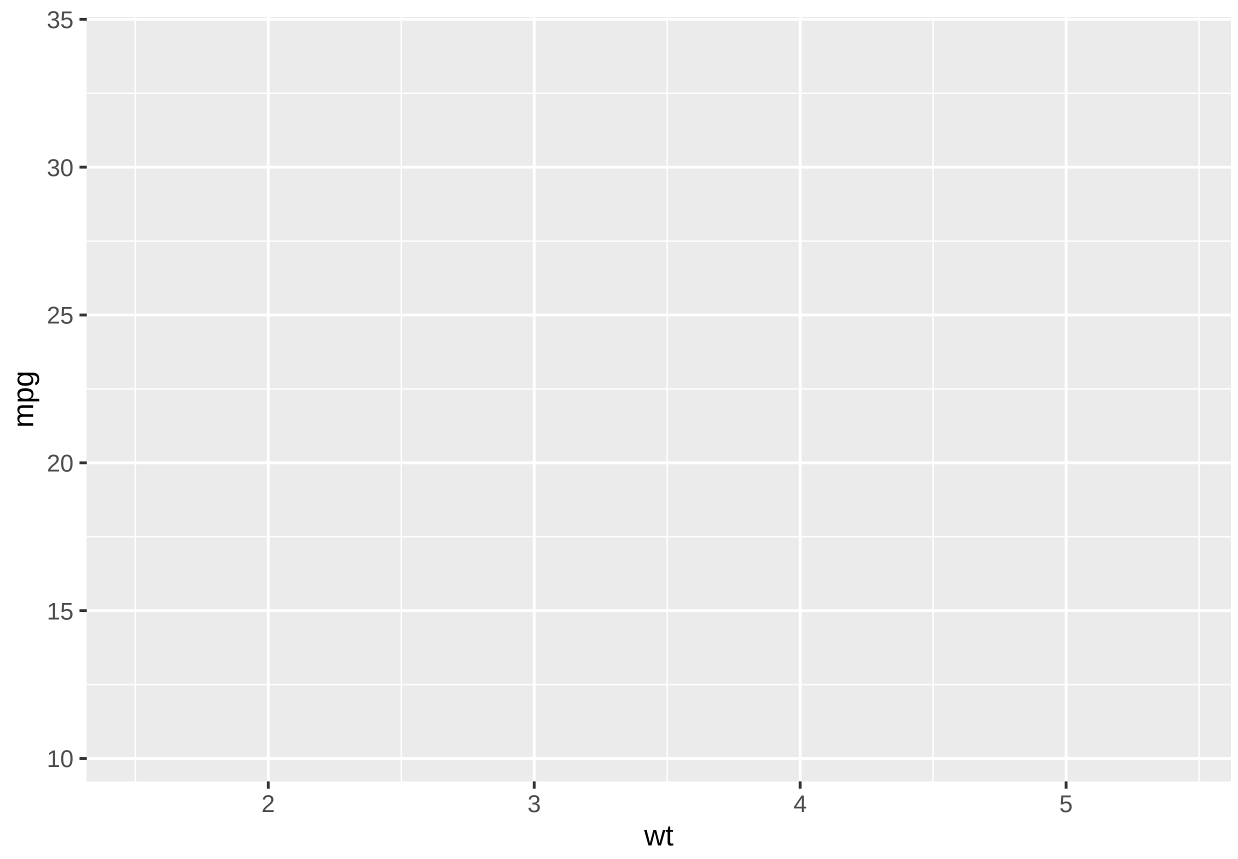
```
#to make plots more fancy, give labels, font size etc
plot(mtcars$wt,mtcars$mpg,
     xlab="Car Weight",
     ylab="Miles per gallon",
     font.lab=10,
     pch=23)
```



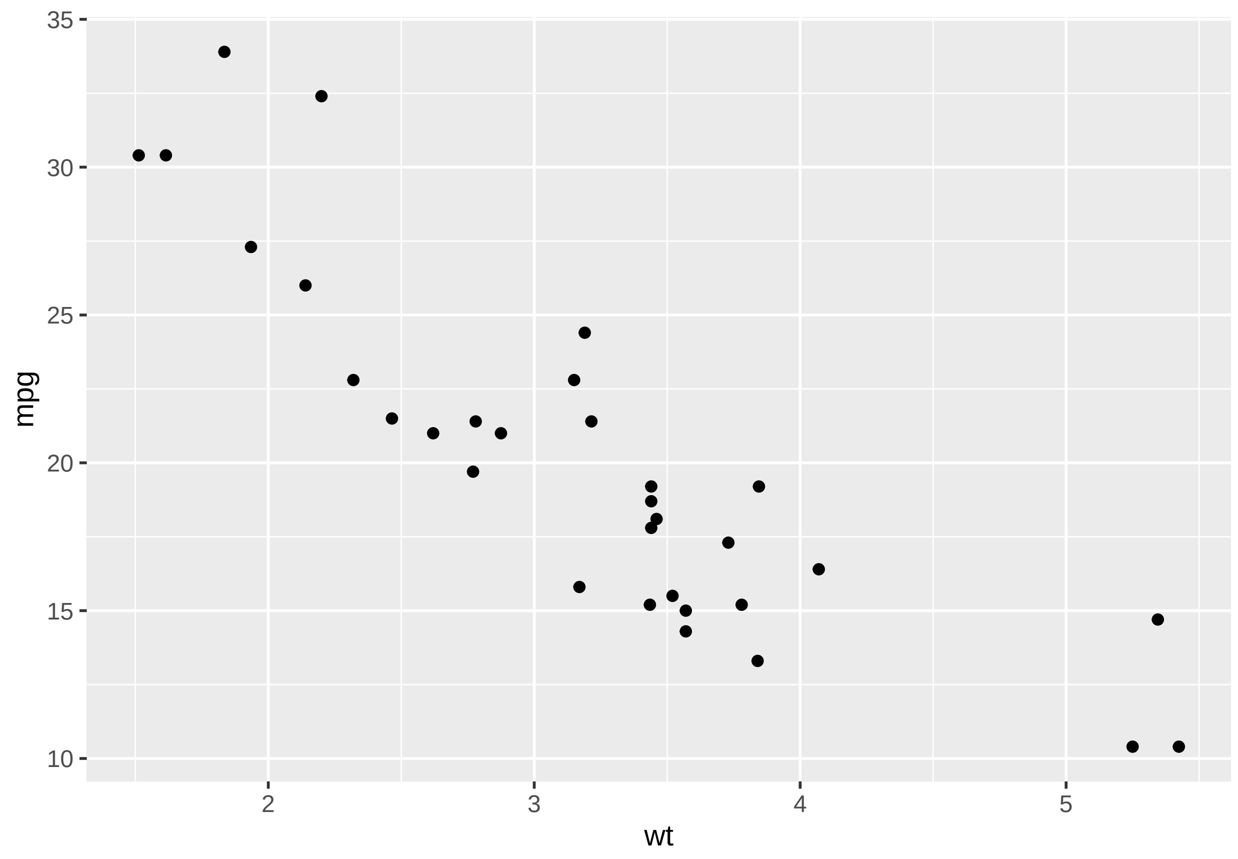
```
####GGPLOT######


#load ggplot package
library(ggplot2)

#ggplot function
ggplot()
```



```r
#generate ggplot (empty plot without any points because ggplot is based on concept of layers so layers
ggplot(mtcars, aes(x = wt, y = mpg))
```

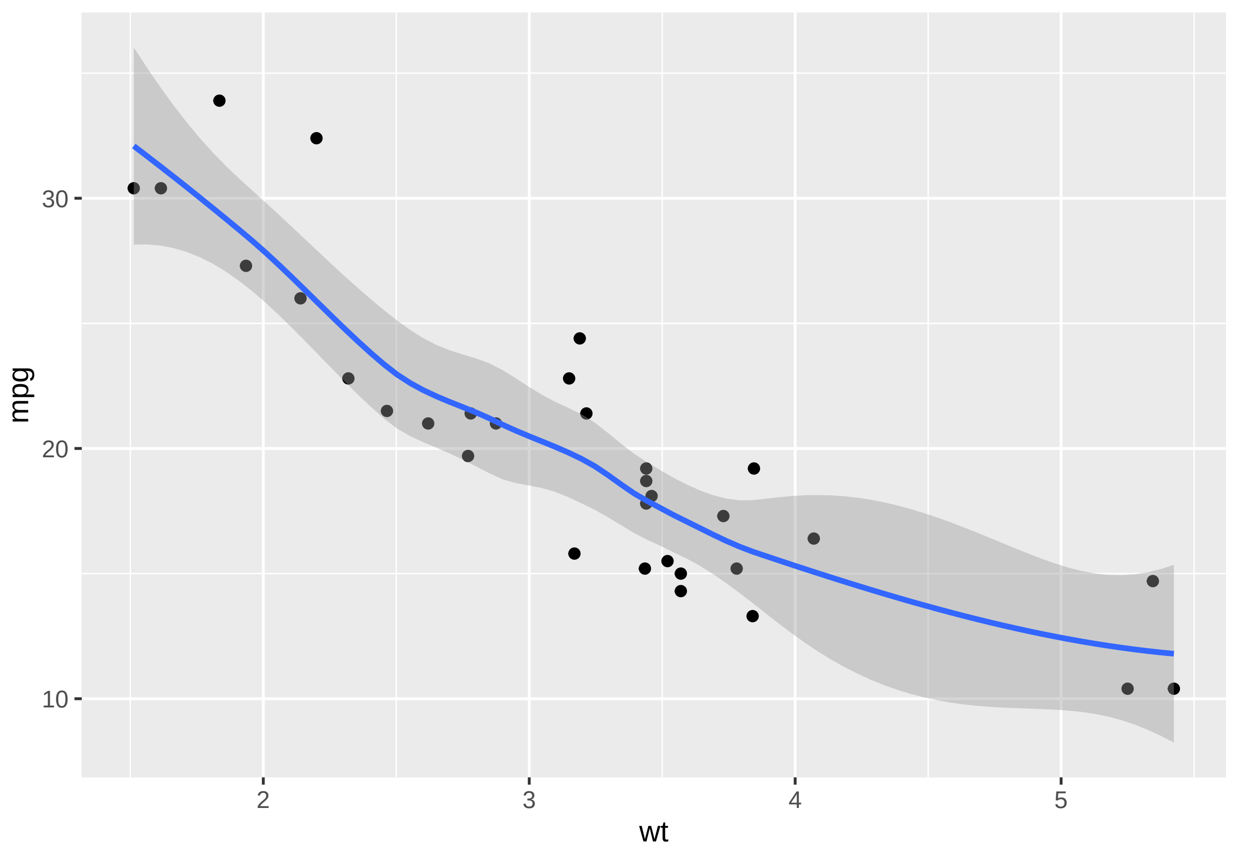


```
##adding layers and data to the plot
# + will add layer to the plot and layer is usually called geom_something such as geom_point, geom_smoo
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point()
```



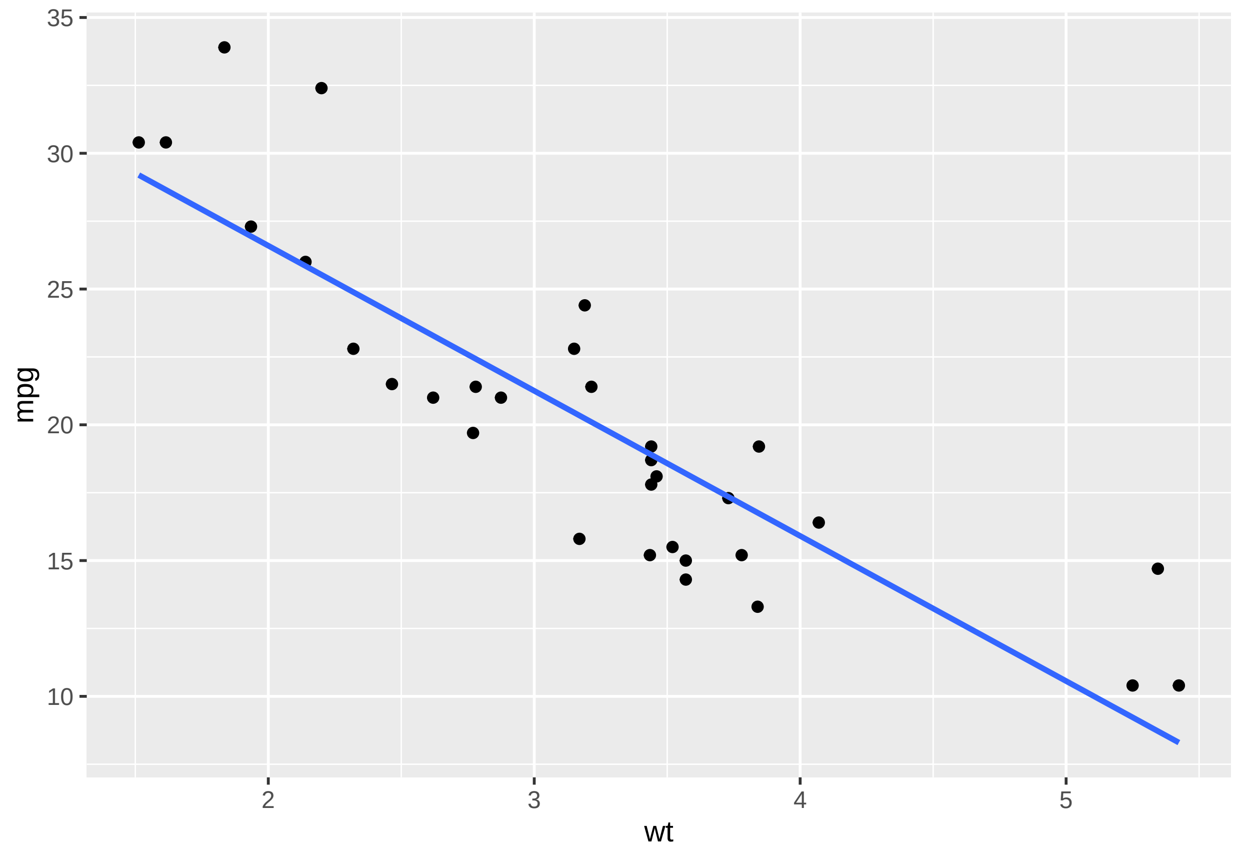
```r
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  geom_smooth()
```

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



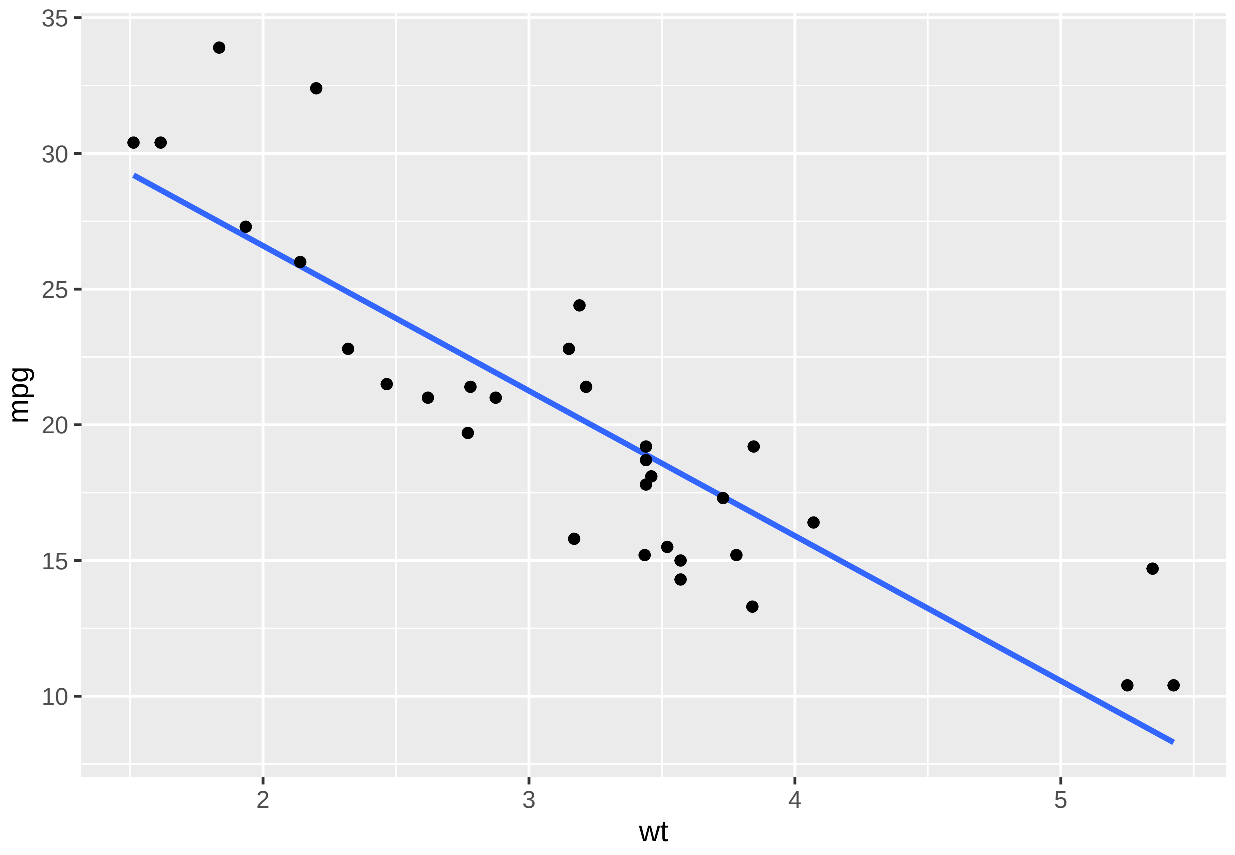
```
#with in layers,it has many options
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  geom_smooth(method = lm, se = FALSE)
```

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



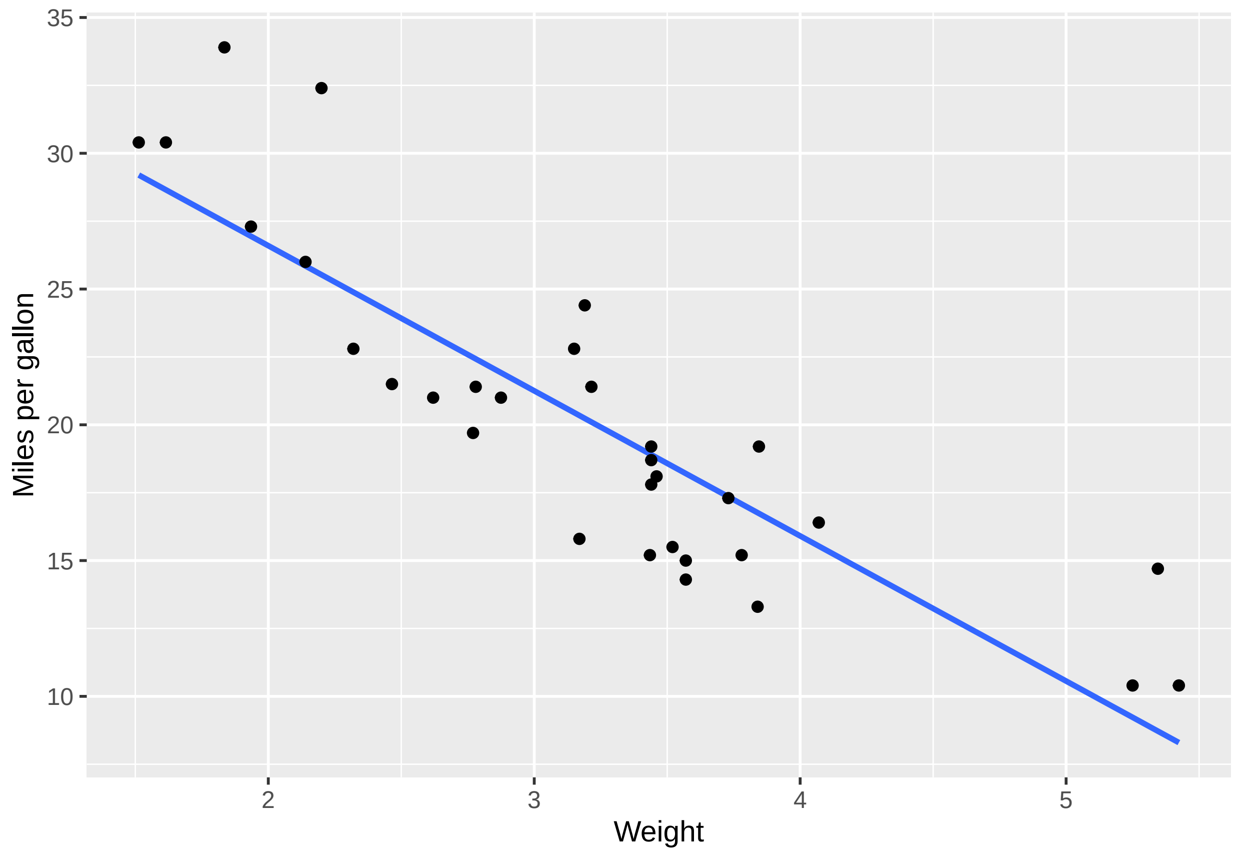
```r
###order of layers can be changed
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point()
```

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



```r
##label x and y axes
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point() +
  xlab("Weight") +
  ylab("Miles per gallon")
```

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

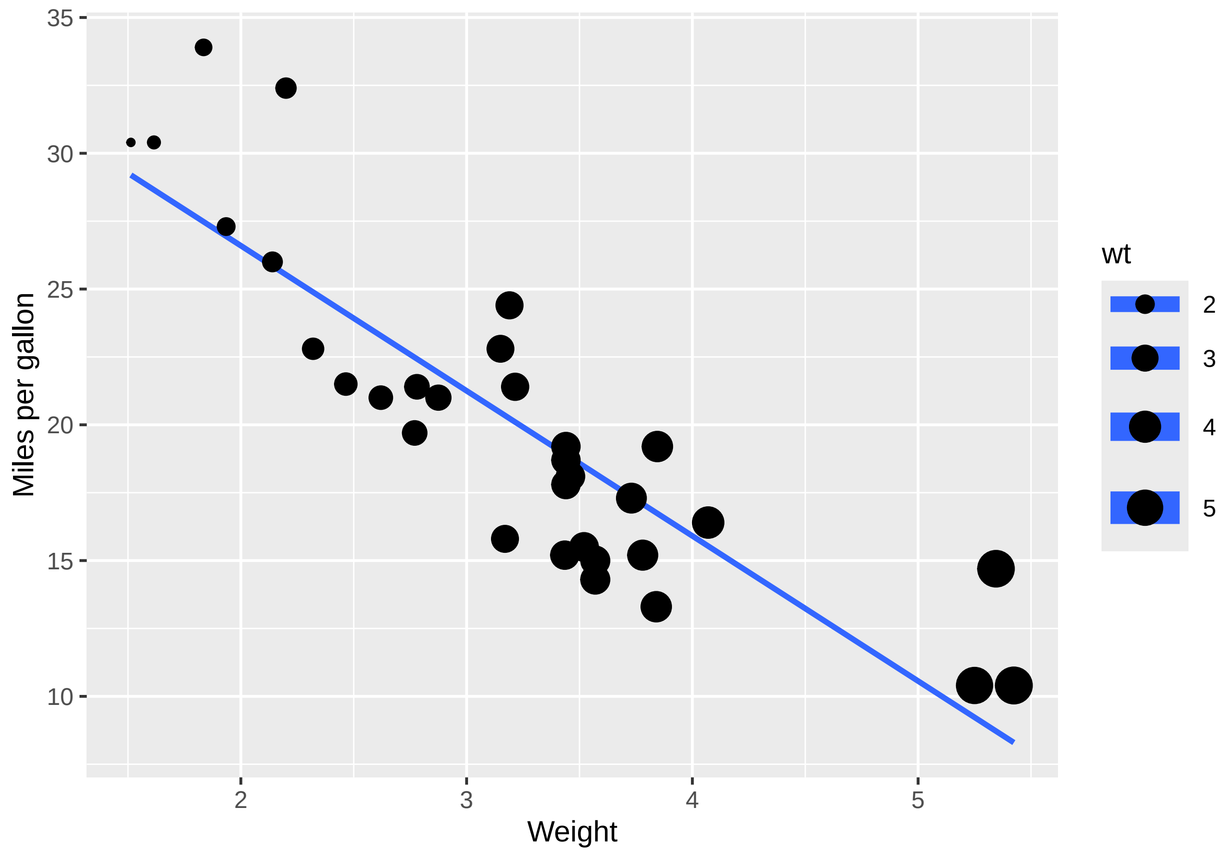


```
## make point size based on weight by two ways

#1. by applying it in main aesthetic (it will generate warning message as it implies to all layers of g
ggplot(mtcars, aes(x = wt, y = mpg, size = wt)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point () +
  xlab("Weight") +
  ylab("Miles per gallon")
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

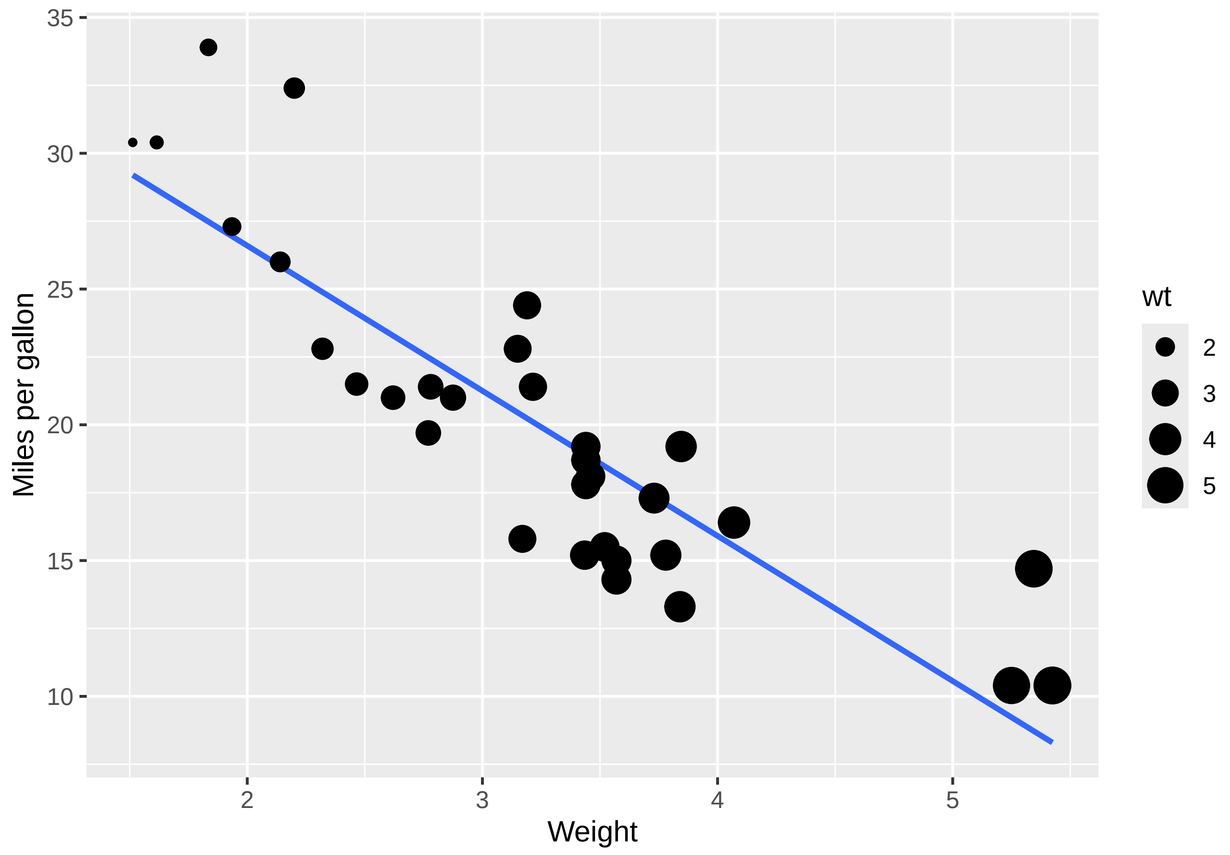
```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```

```
## Warning: The following aesthetics were dropped during statistical transformation: size.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a ‘group‘ aesthetic or to convert a numerical
##   variable into a factor?
```



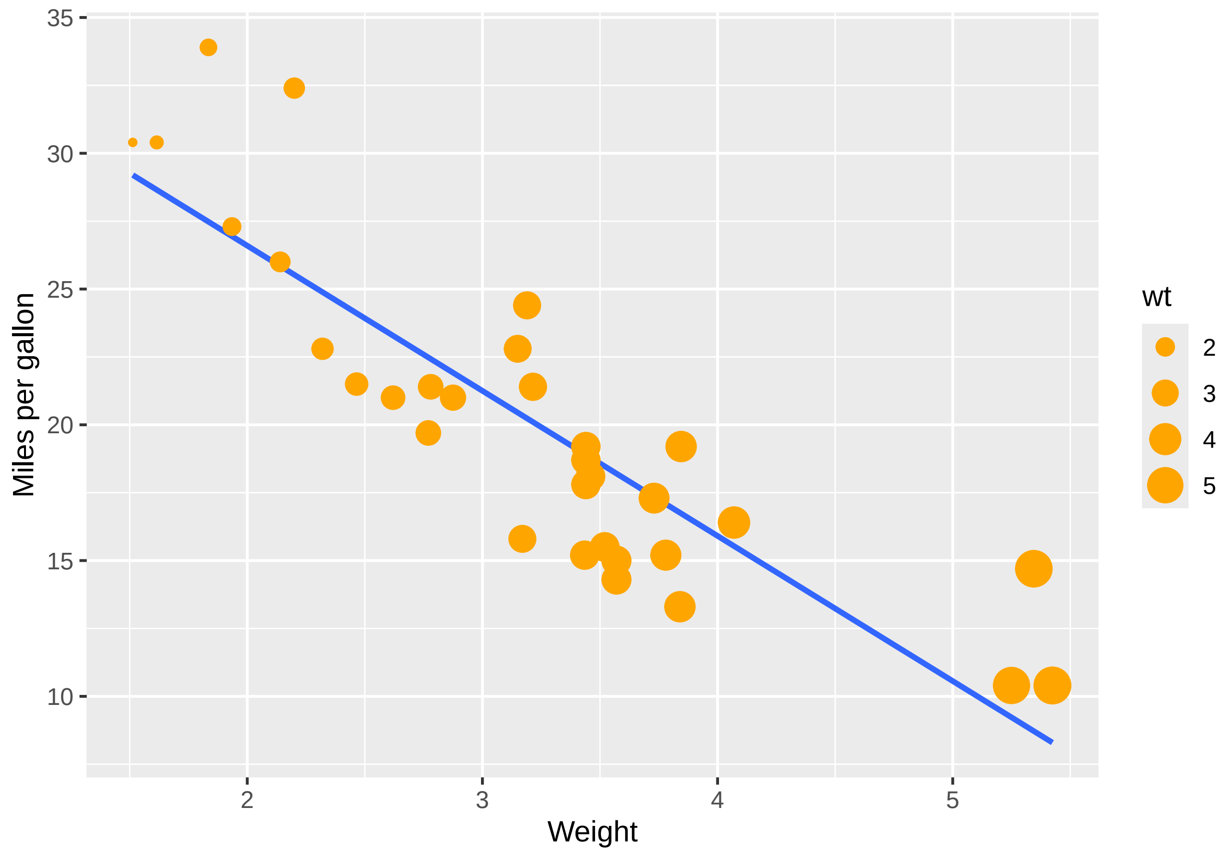
```
#2. Instead apply to only one layer
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point(aes(size = wt)) +
  xlab("Weight") +
  ylab("Miles per gallon")
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



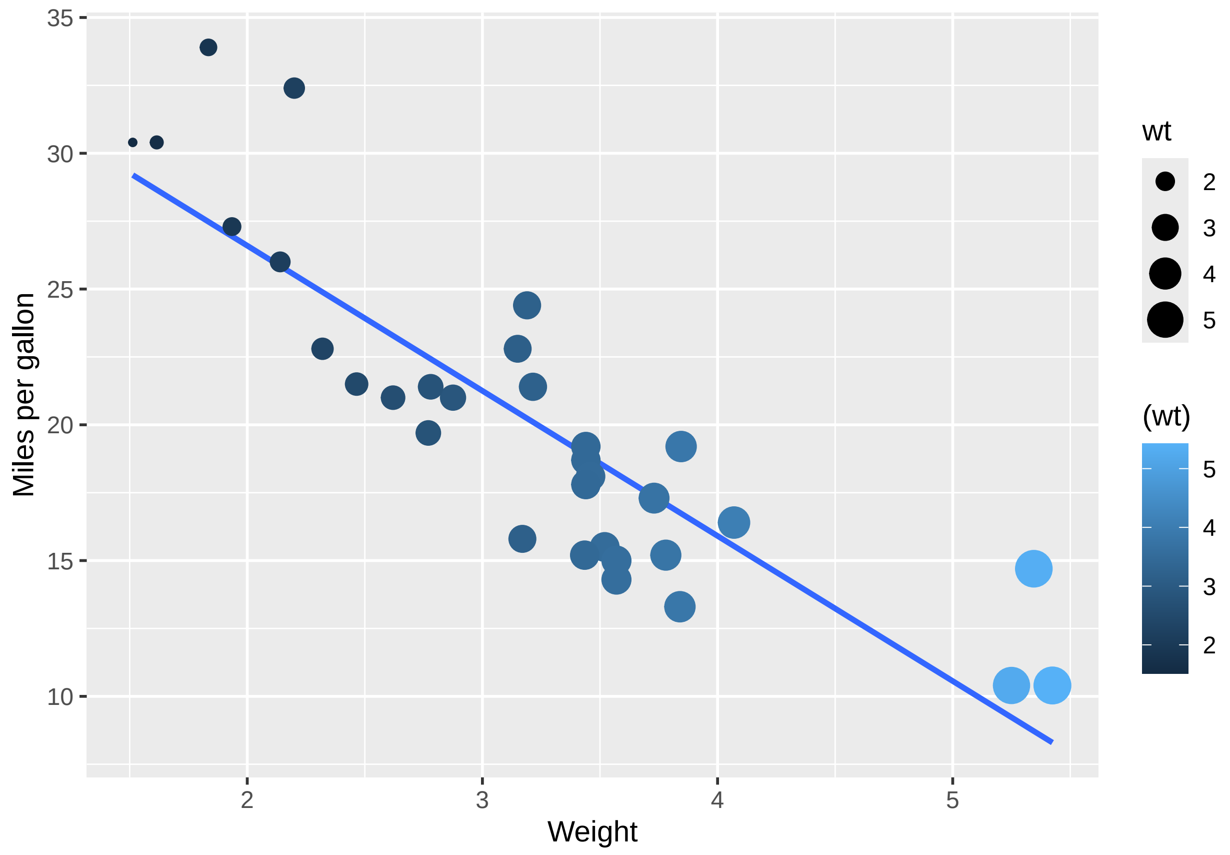
```
# add color to points
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point(aes(size = wt),color=("orange")) +
  xlab("Weight") +
  ylab("Miles per gallon")
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



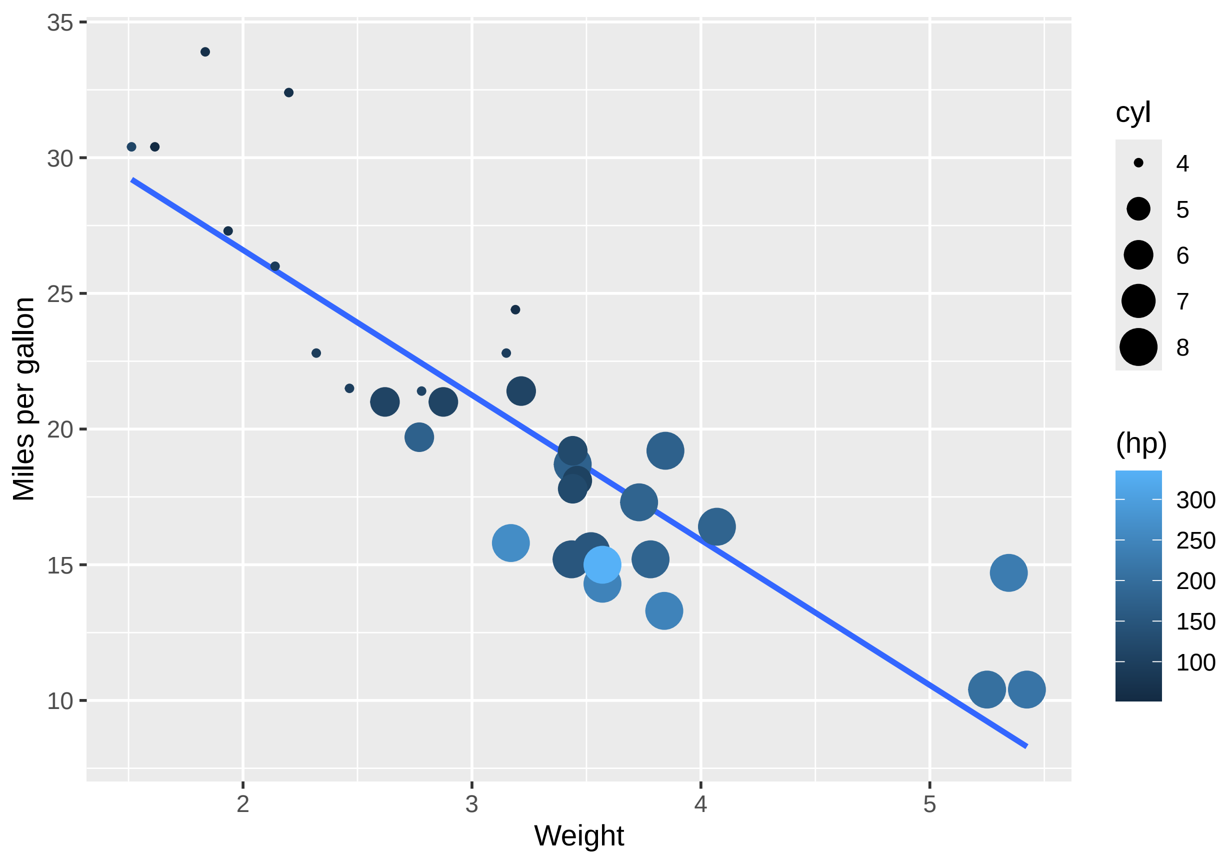
```r
# change color according to weight
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point(aes(size = wt,color=(wt))) +
  xlab("Weight") +
  ylab("Miles per gallon")
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



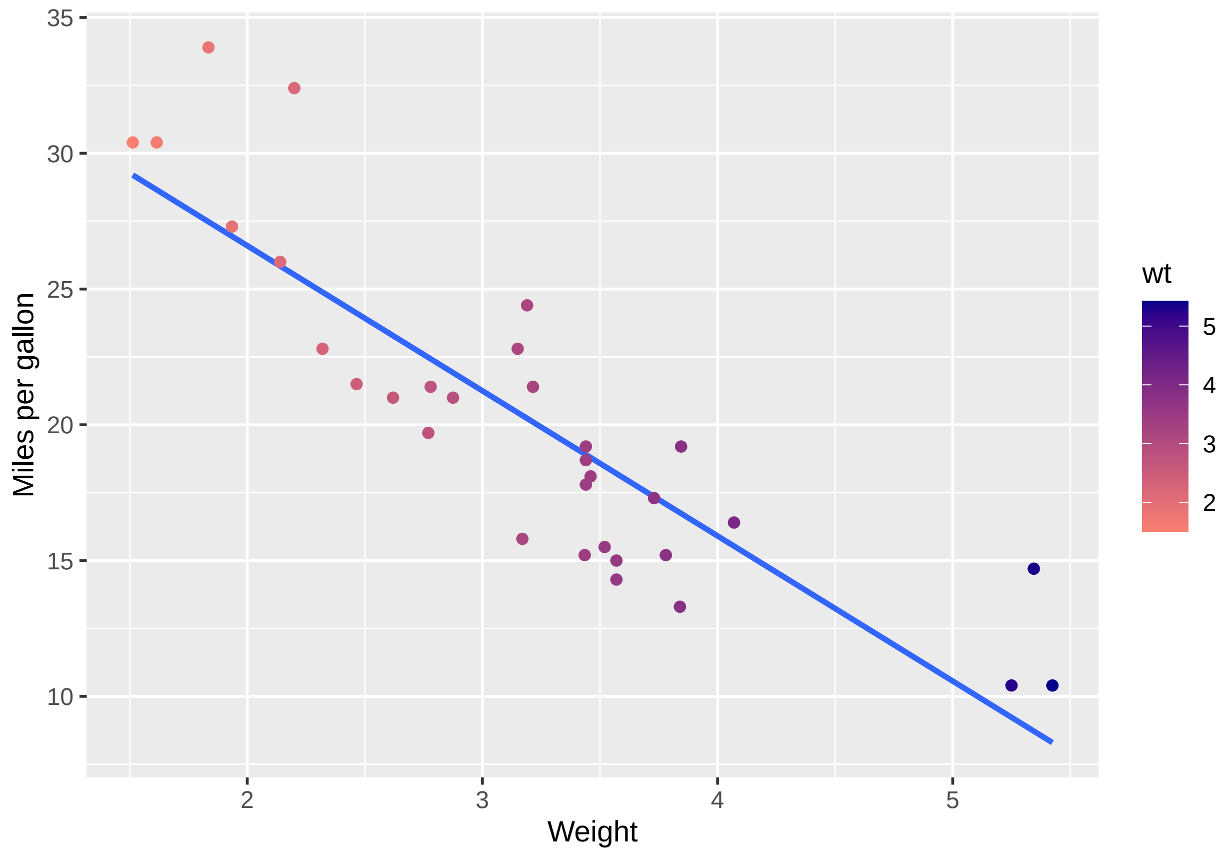
```r
##  can also change size of points and color according to other variables however interpretation can ge
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point(aes(size = cyl,color=(hp))) +
  xlab("Weight") +
  ylab("Miles per gallon")
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



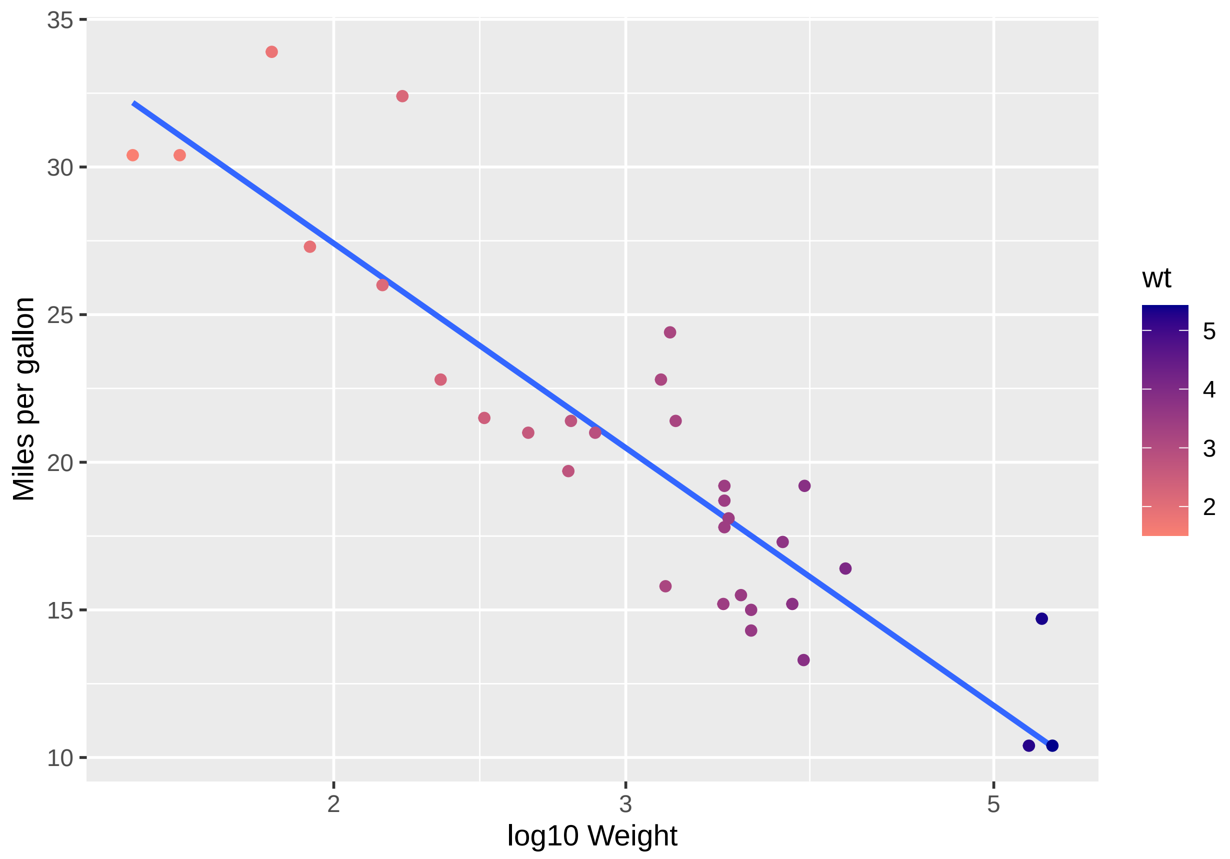
```
#Color gradients
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point(aes(color = wt)) +
  xlab("Weight") +
  ylab("Miles per gallon") +
  scale_colour_gradient(low = "salmon", high = "darkblue")
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



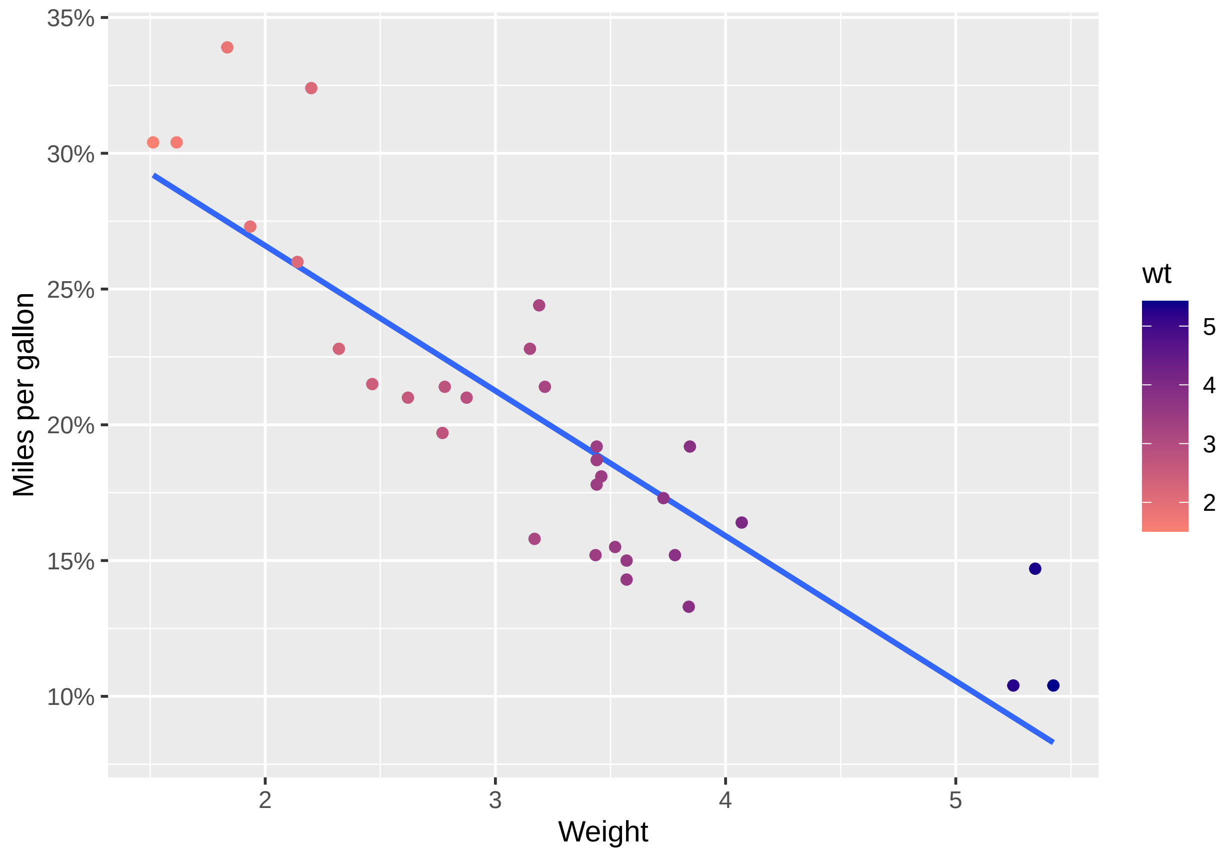
```
#Scale

#log10
###change labels for x axis
##automatically applies log10 to x-axis
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point(aes(color = wt)) +
  xlab("log10 Weight") +
  ylab("Miles per gallon") +
  scale_colour_gradient(low = "salmon", high = "darkblue")+
  scale_x_log10()
```

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



```
# scale: percentage
##divide y by 100 for correct percentage
##converts y axis to percentage
ggplot(mtcars, aes(x = wt, y = mpg/100)) +
  geom_smooth(method = lm, se = FALSE) +
  geom_point(aes(color = wt)) +
  xlab("Weight") +
  ylab("Miles per gallon") +
  scale_colour_gradient(low = "salmon", high = "darkblue")+
  scale_y_continuous(labels = scales::percent)
```

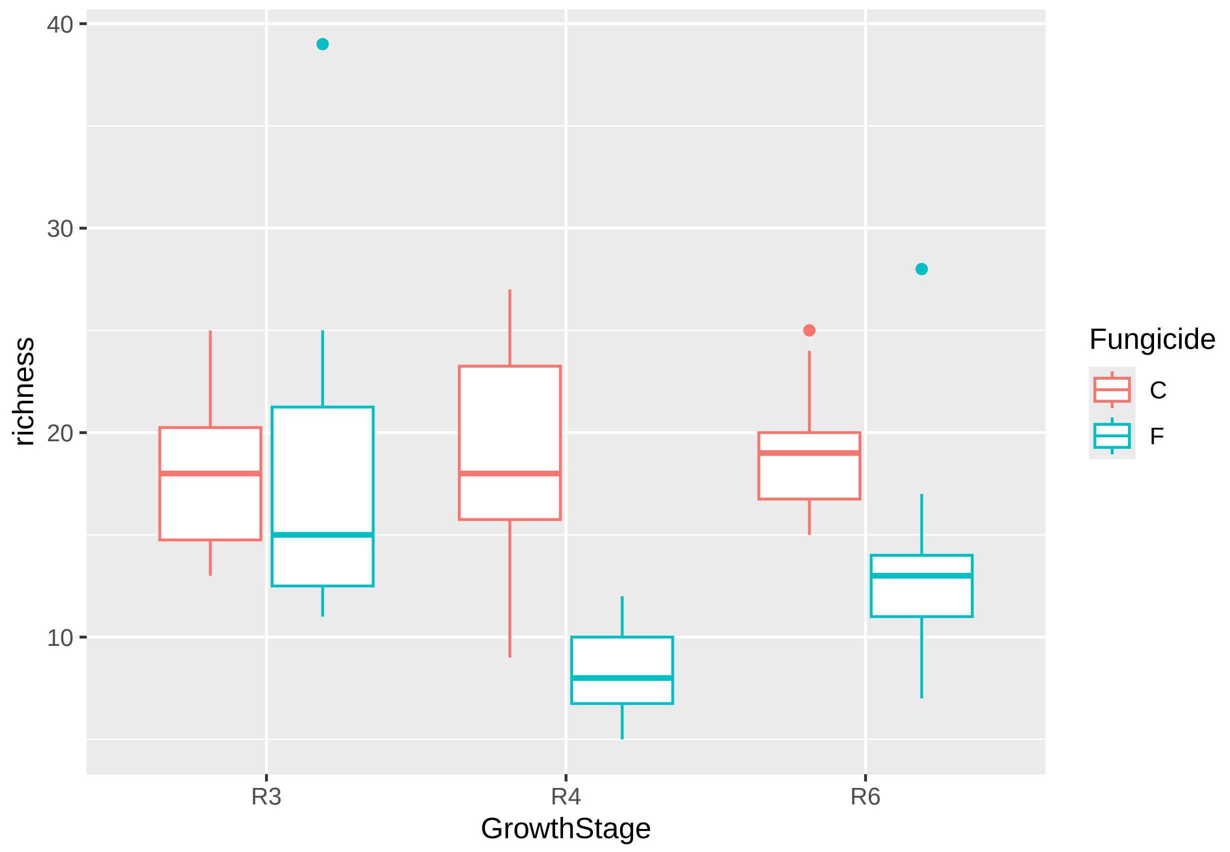
```
## `geom_smooth()` using formula = 'y ~ x'
```



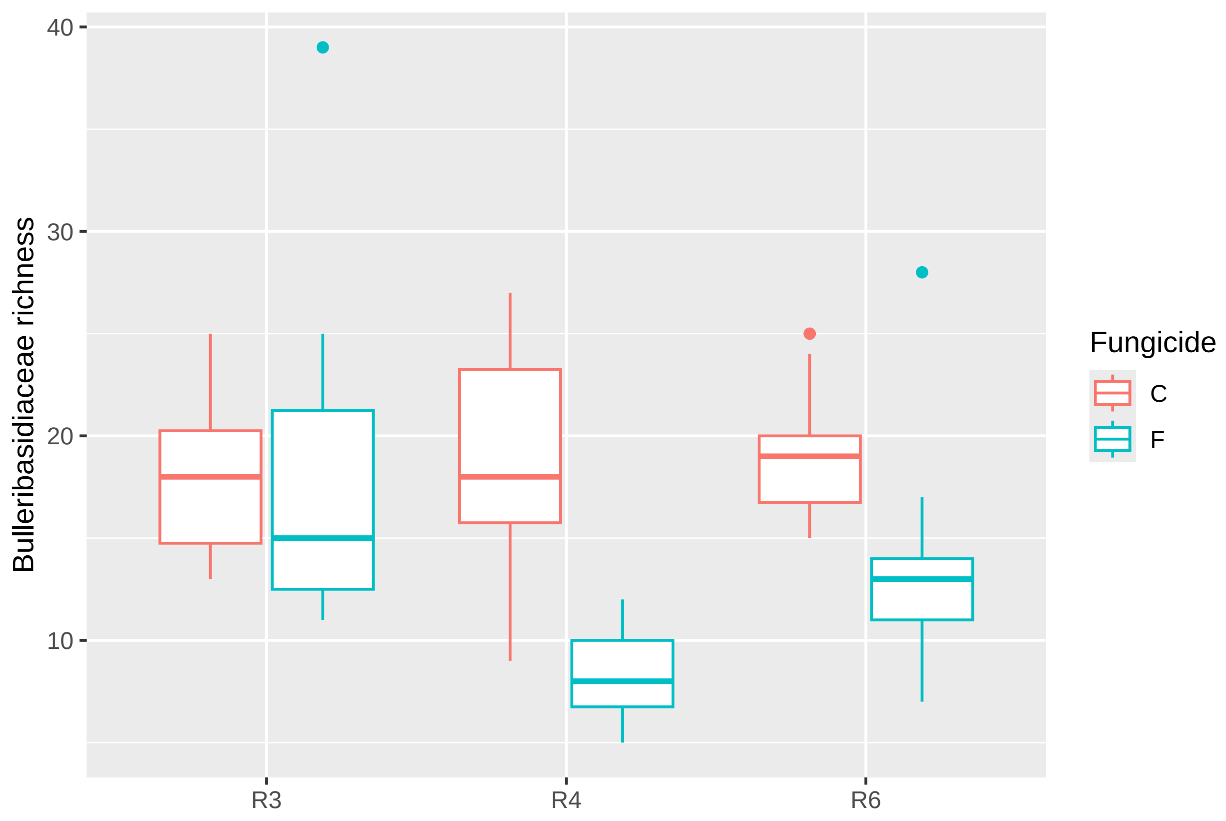
```r
####CATEGORICAL AND NUMERIC #####

#load data
bull.richness <- read.csv("/Users/Mamata/Downloads/Bull_richness.csv")
str(bull.richness)
```

```
## 'data.frame':    287 obs. of  16 variables:
##  $ SampleID        : chr  "Corn2017LeafObjective2Collection1T1R1CAH2" "Corn2017LeafObjective2Collecti
##  $ Crop            : chr  "Corn" "Corn" "Corn" "Corn" ...
##  $ Objective       : chr  "Objective 2" "Objective 2" "Objective 2" "Objective 2" ...
##  $ Collection      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Compartment     : chr  "Leaf" "Leaf" "Leaf" "Leaf" ...
##  $ DateSampled     : chr  "6/26/17" "6/26/17" "6/26/17" "6/26/17" ...
##  $ GrowthStage     : chr  "V6" "V6" "V6" "V6" ...
##  $ Treatment       : chr  "Conv." "Conv." "Conv." "Conv." ...
##  $ Rep             : chr  "R1" "R1" "R1" "R1" ...
##  $ Sample          : chr  "A" "B" "C" "A" ...
##  $ Fungicide       : chr  "C" "C" "C" "F" ...
##  $ Target_organism: chr  "Fungi" "Fungi" "Fungi" "Fungi" ...
##  $ Location        : chr  "Kellogg Biological Station" "Kellogg Biological Station" "Kellogg Biologica
##  $ Experiment      : chr  "LTER" "LTER" "LTER" "LTER" ...
##  $ Year            : int  2017 2017 2017 2017 2017 2017 2017 2017 2017 2017 ...
##  $ richness        : int  9 6 5 7 4 2 3 8 4 4 ...
```

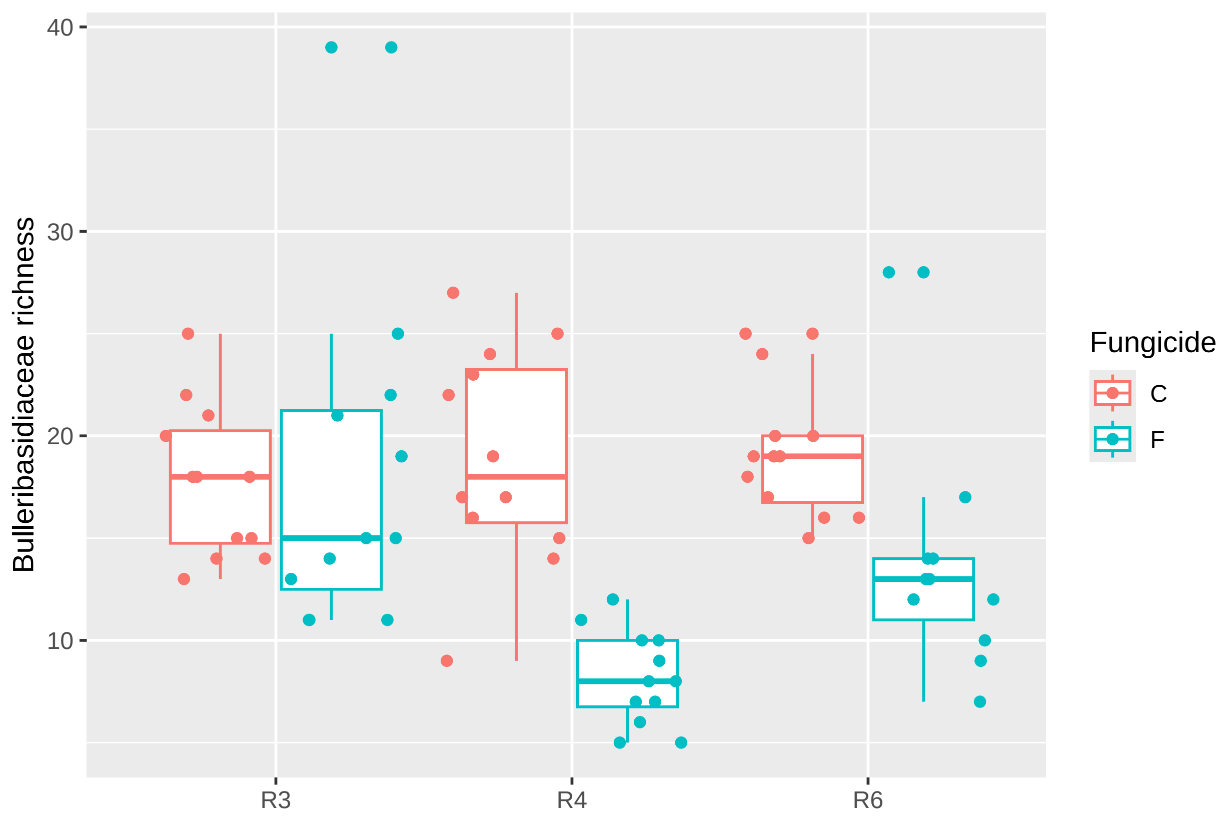
```r
#subset to soy data
bull.richness.soy.no.till <- bull.richness[bull.richness$Crop == "Soy" &
                                          bull.richness$Treatment == "No-till",]


#BoxPlot
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, color = Fungicide)) +
  geom_boxplot()
```



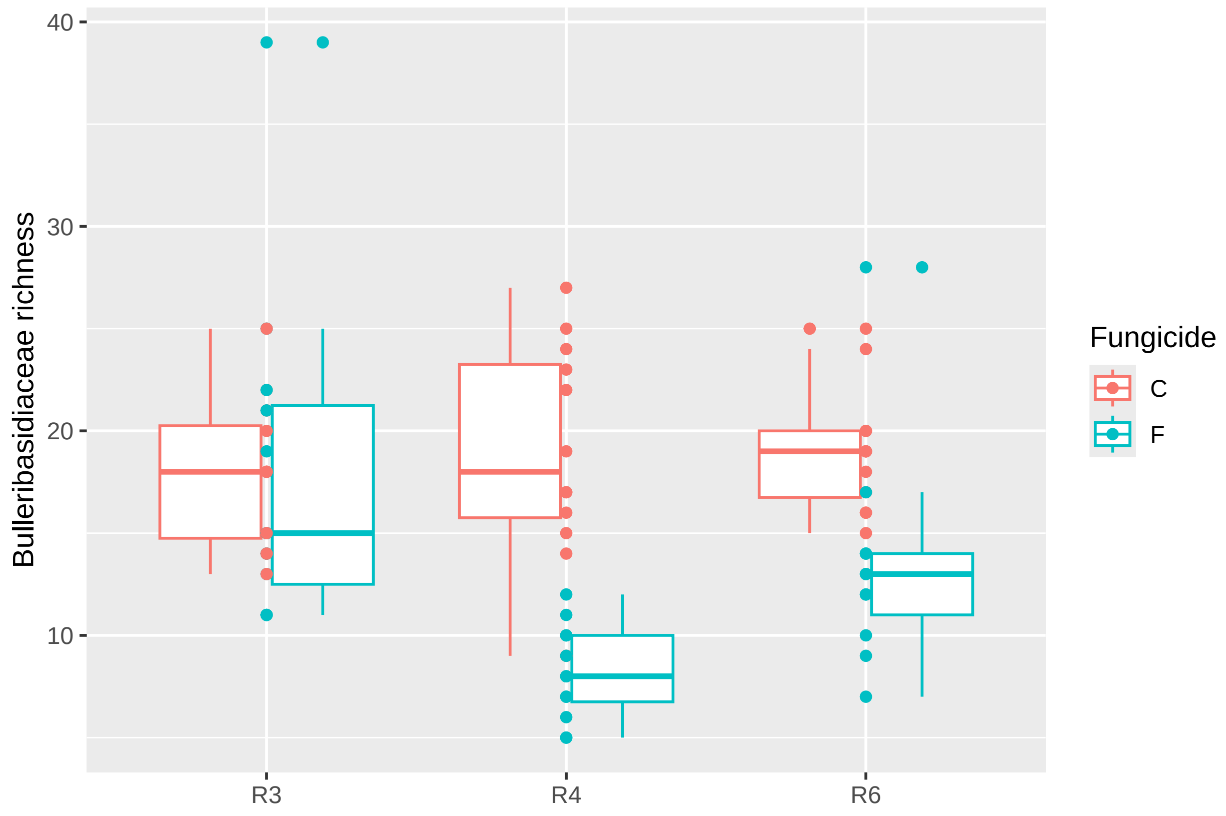
```r
#with labels
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, color = Fungicide)) +
  geom_boxplot() +
  xlab("") +
  ylab("Bulleribasidiaceae richness")
```



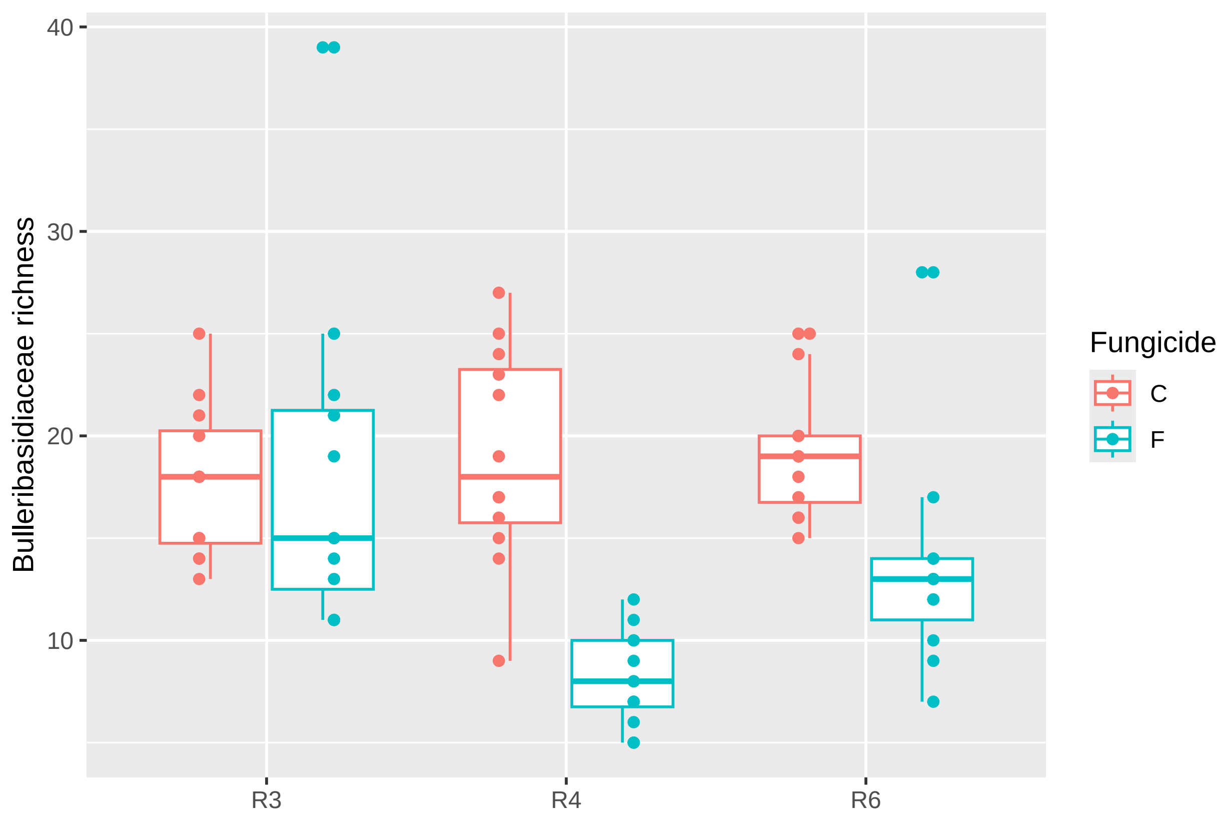
```r
#problem with this type of visualization is they don't show all of the data points

#adding layer to show all the data points
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, color = Fungicide)) +
  geom_boxplot() +
  xlab("") +
  ylab("Bulleribasidiaceae richness") +
  geom_point(position=position_jitterdodge(dodge.width=0.9))
```



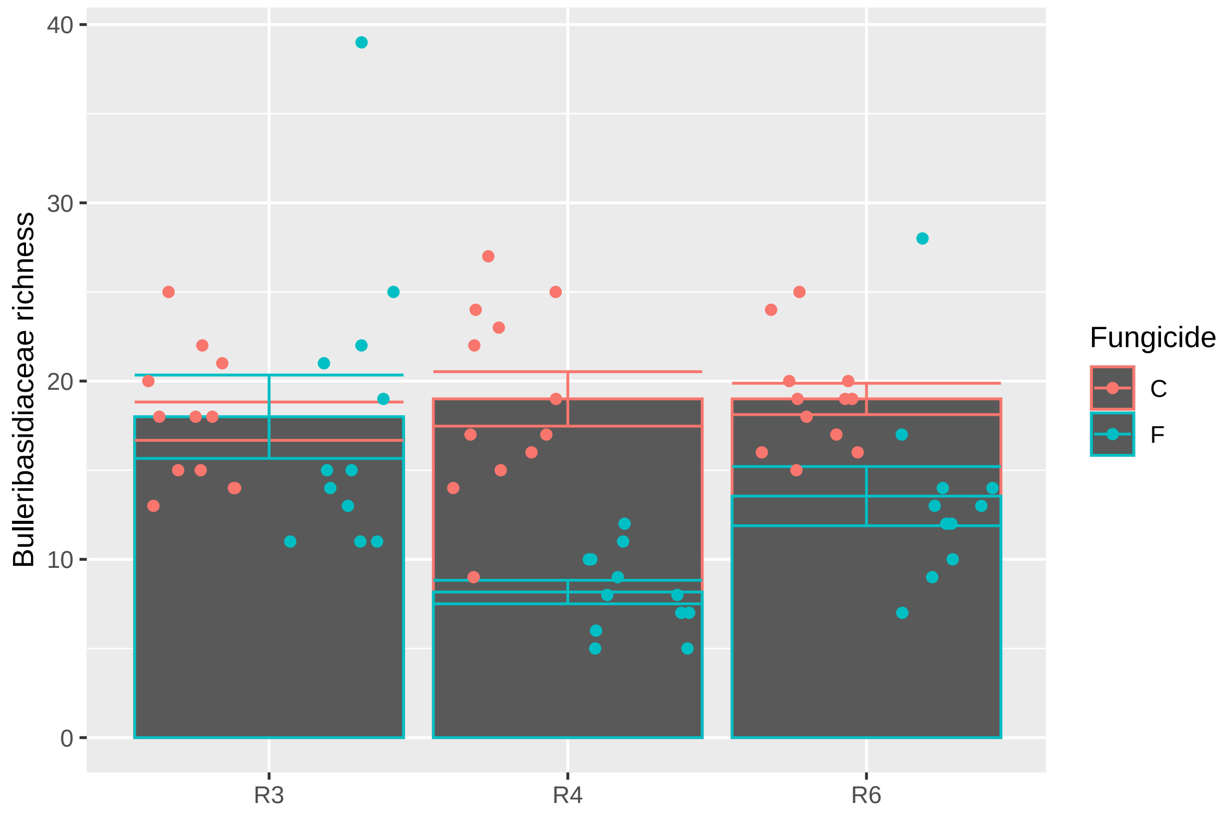
```r
## dodge is basically placing bars or box plots side by side instead of stacking them and jitter_dodge

##if not dodge
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, color = Fungicide)) +
  geom_boxplot() +
  xlab("") +
  ylab("Bulleribasidiaceae richness") +
  geom_point()
```



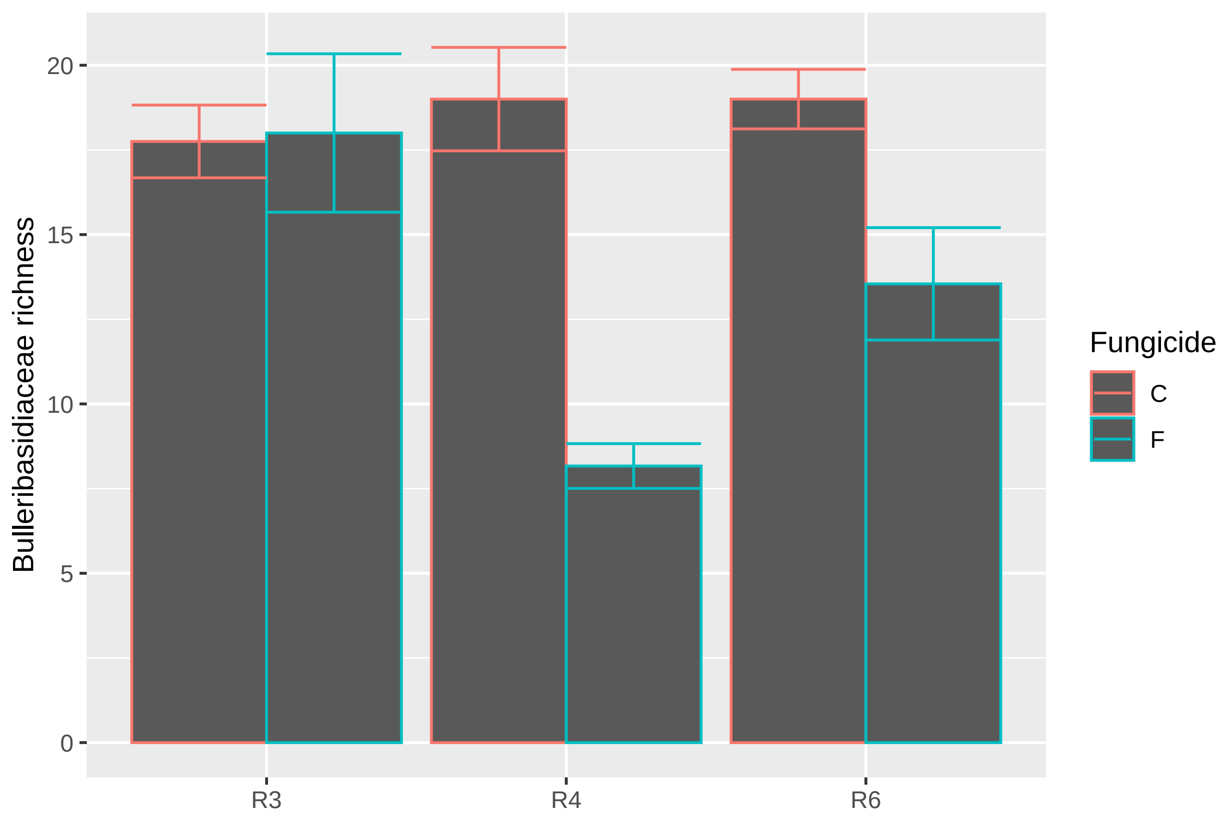
```r
##if not jitterdodge
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, color = Fungicide)) +
  geom_boxplot() +
  xlab("") +
  ylab("Bulleribasidiaceae richness") +
  geom_point(position=position_dodge(width=0.9))
```



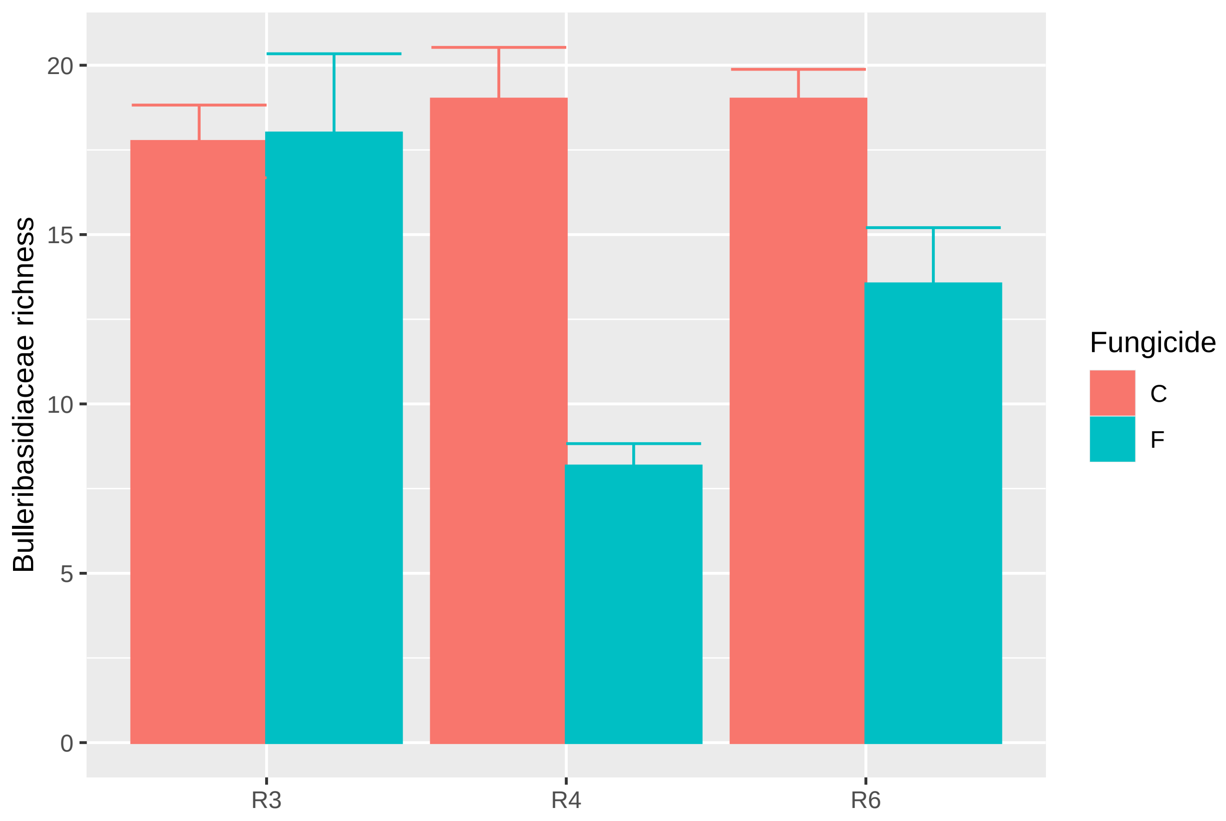
```
### BARCHART ###

# barplots
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, color = Fungicide)) +
  stat_summary(fun=mean,geom="bar") +
  stat_summary(fun.data = mean_se, geom = "errorbar") +
  xlab("") +
  ylab("Bulleribasidiaceae richness") +
  geom_point(position=position_jitterdodge(dodge.width=0.9))
```



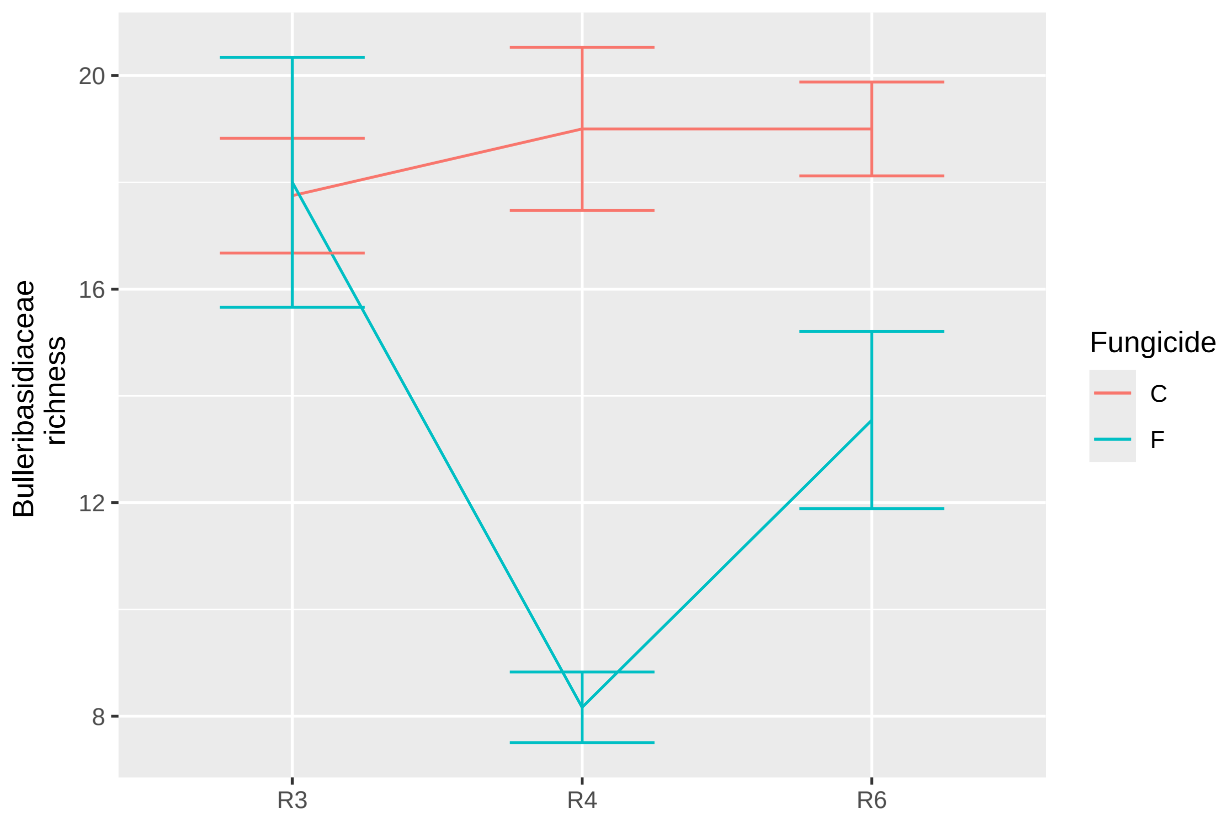
```r
## We got overlapping bars so we need to dodge the bars

# Dodge bars and error bars
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, color = Fungicide)) +
  stat_summary(fun=mean,geom="bar", position = "dodge") +
  stat_summary(fun.data = mean_se, geom = "errorbar", position = "dodge") +
  xlab("") +
  ylab("Bulleribasidiaceae richness")
```



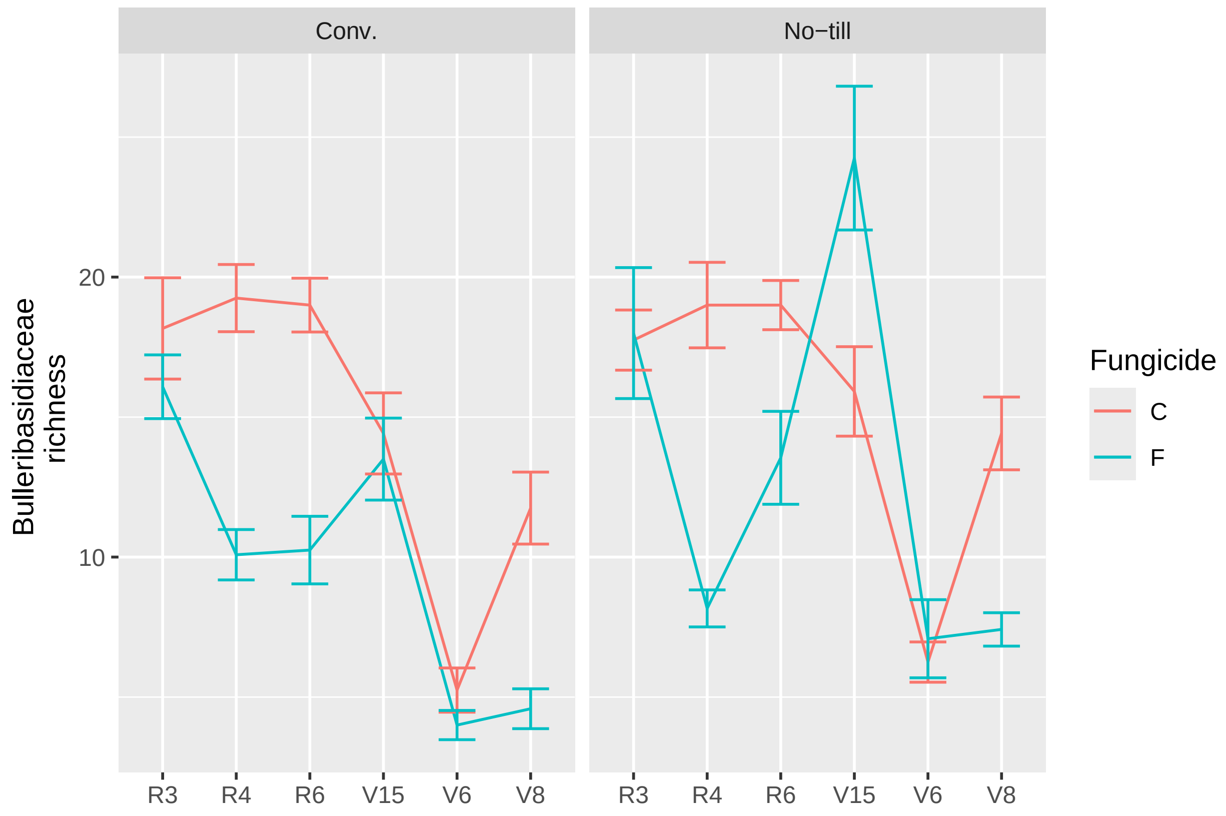
```r
# It gives nice figure but bar fill doesnot look good

# Difference between color and fill
## Color controls outside color such as color of error bars, points, bars, lines
## Fill actually fills the bar with different color

# to change colors of bar filled and error bars, points
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, color = Fungicide, fill = Fungicid
  stat_summary(fun=mean,geom="bar", position = "dodge") +
  stat_summary(fun.data = mean_se, geom = "errorbar", position = "dodge") +
  xlab("") +
  ylab("Bulleribasidiaceae richness")
```



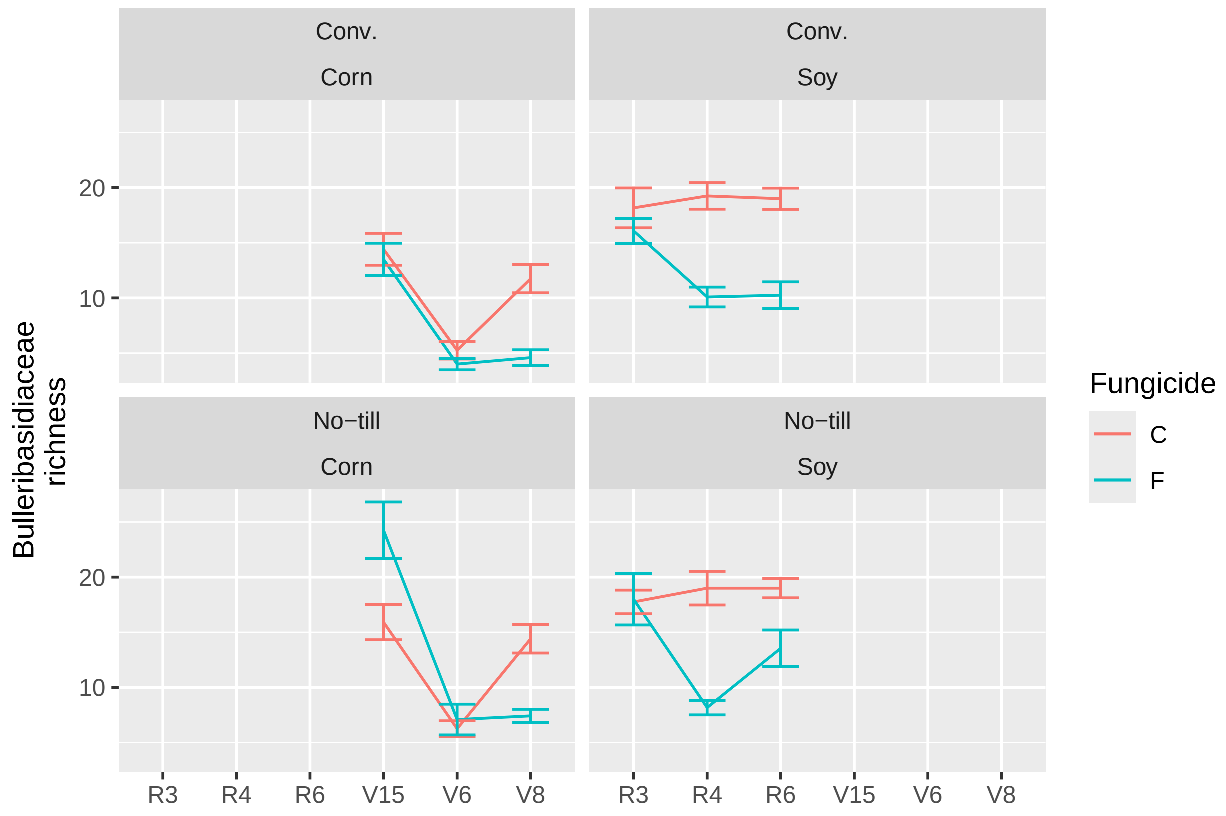
```
### Lines connnecting ####

##Lines

## we use group function which means that we are going to treat fungicide as main grouping variable for
ggplot(bull.richness.soy.no.till, aes(x = GrowthStage, y = richness, group = Fungicide, color = Fungicid
  stat_summary(fun=mean,geom="line") +
  stat_summary(fun.data = mean_se, geom = "errorbar", width = 0.5) +
  ylab("Bulleribasidiaceae \n richness") +
  xlab("")
```



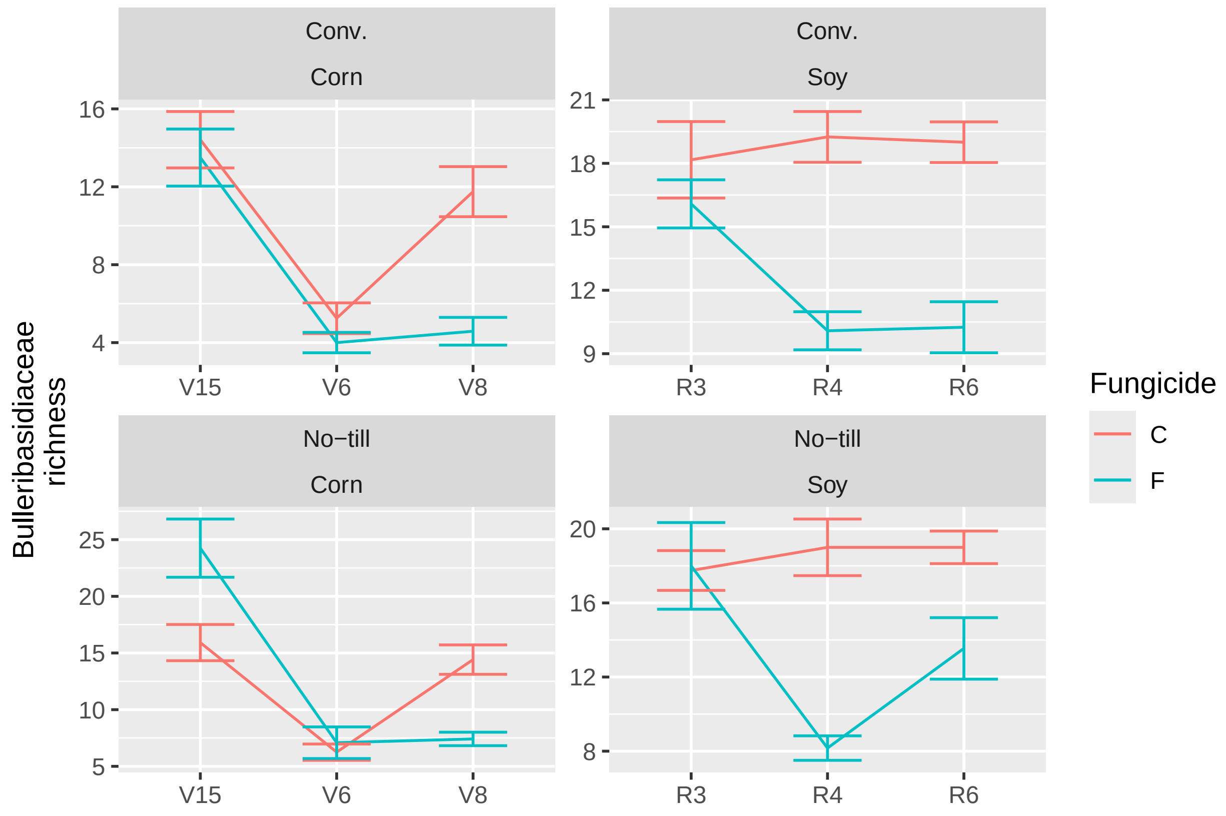
```r
## Faceting

# facet wrap for making multiple plots for same variable but split by categorical variable
ggplot(bull.richness, aes(x = GrowthStage, y = richness, group = Fungicide, color = Fungicide)) +
  stat_summary(fun=mean,geom="line") +
  stat_summary(fun.data = mean_se, geom = "errorbar", width = 0.5) +
  ylab("Bulleribasidiaceae \n richness") +
  xlab("") +
  facet_wrap(~Treatment)
```



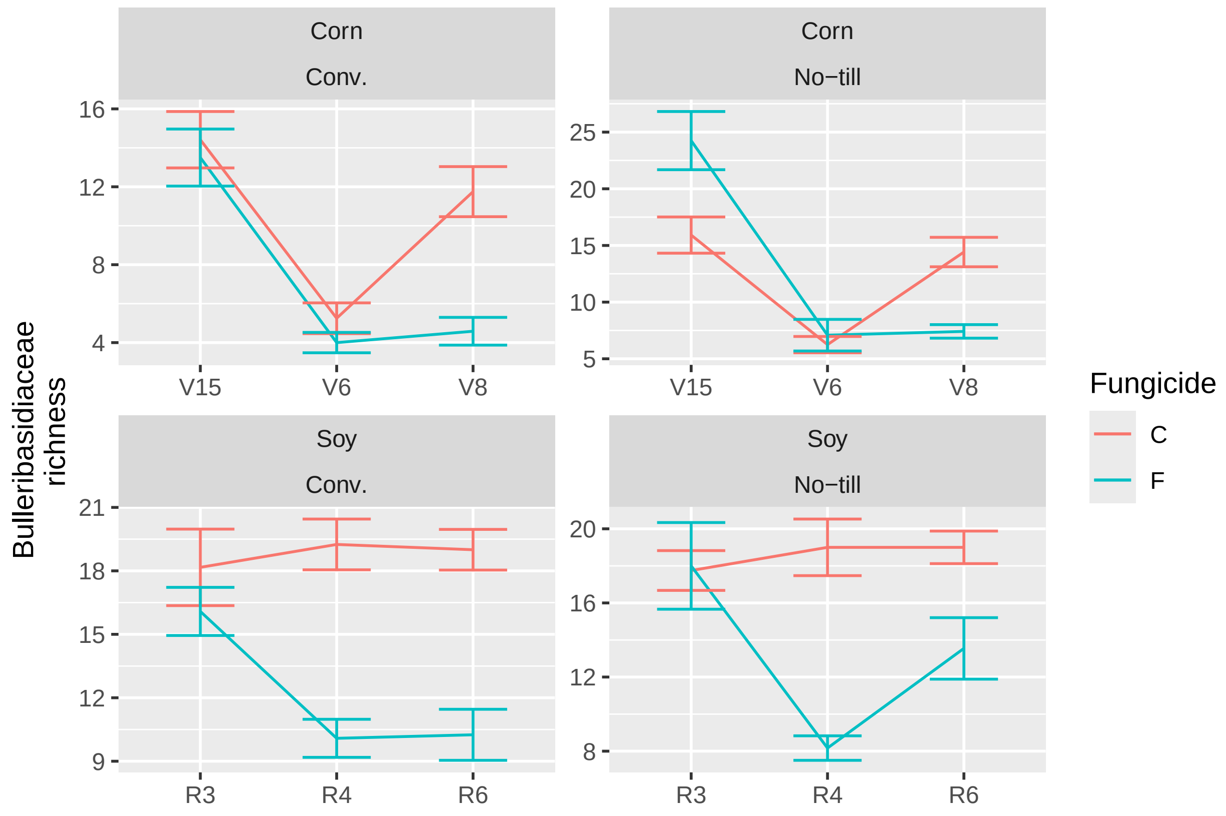
```
# for interaction
ggplot(bull.richness, aes(x = GrowthStage, y = richness, group = Fungicide, color = Fungicide)) +
  stat_summary(fun=mean,geom="line") +
  stat_summary(fun.data = mean_se, geom = "errorbar", width = 0.5) +
  ylab("Bulleribasidiaceae \n richness") +
  xlab("") +
  facet_wrap(~Treatment*Crop)
```



```r
# Problem is it shows all the variables in X axis that doesnot have data as well

# to solve the problem
ggplot(bull.richness, aes(x = GrowthStage, y = richness, group = Fungicide, color = Fungicide)) +
  stat_summary(fun=mean,geom="line") +
  stat_summary(fun.data = mean_se, geom = "errorbar", width = 0.5) +
  ylab("Bulleribasidiaceae \n richness") +
  xlab("") +
  facet_wrap(~Treatment*Crop, scales = "free")
```



```
# change the order of facet
ggplot(bull.richness, aes(x = GrowthStage, y = richness, group = Fungicide, color = Fungicide)) +
  stat_summary(fun=mean,geom="line") +
  stat_summary(fun.data = mean_se, geom = "errorbar", width = 0.5) +
  ylab("Bulleribasidiaceae \n richness") +
  xlab("") +
  facet_wrap(~Crop*Treatment, scales = "free")
```



Corn
Conv.
Corn
No−till
Soy
Conv.
Soy
No−till
Bulleribasidiaceae richness
16
12
8
4
25
20
15
10
5
21
18
15
12
9
20
16
12
8
V15
V6
V8
R3
R4
R6
Fungicide
C
F