Data Visualization; ggplot2

Jennifer Brazeal

9/9/19

We're going to replicate the plots from Day 3 using ggplot2 instead of base R plotting

```
?CO2
CO2 <- datasets::CO2
```

What is ggplot2

It is a visualization package that is part of the "tidyverse"

https://ggplot2.tidyverse.org/

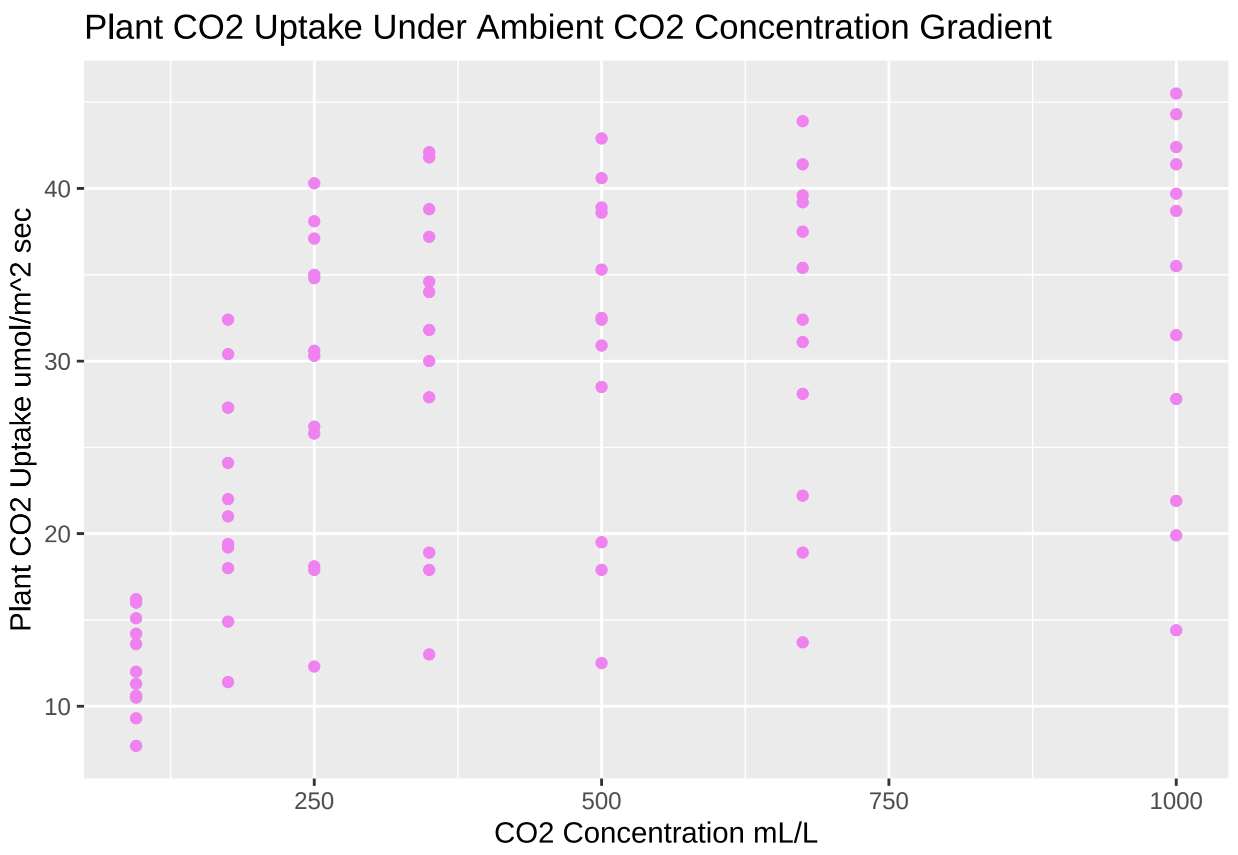
https://r4ds.had.co.nz/data-visualisation.html

Data: ggplot2 expects data to be in data frame format, with columns representing variables you will visualize.
Aesthetics: aes maps the actual data to an aesthetic, such as color or position.
Geometric: geoms represent the actual thing you see on your plot.

You need these three things at minimum to visualize data in ggplot2. You can also build off of skeleton or base ggplots to add to and/or modify your plots.

Example formatting:

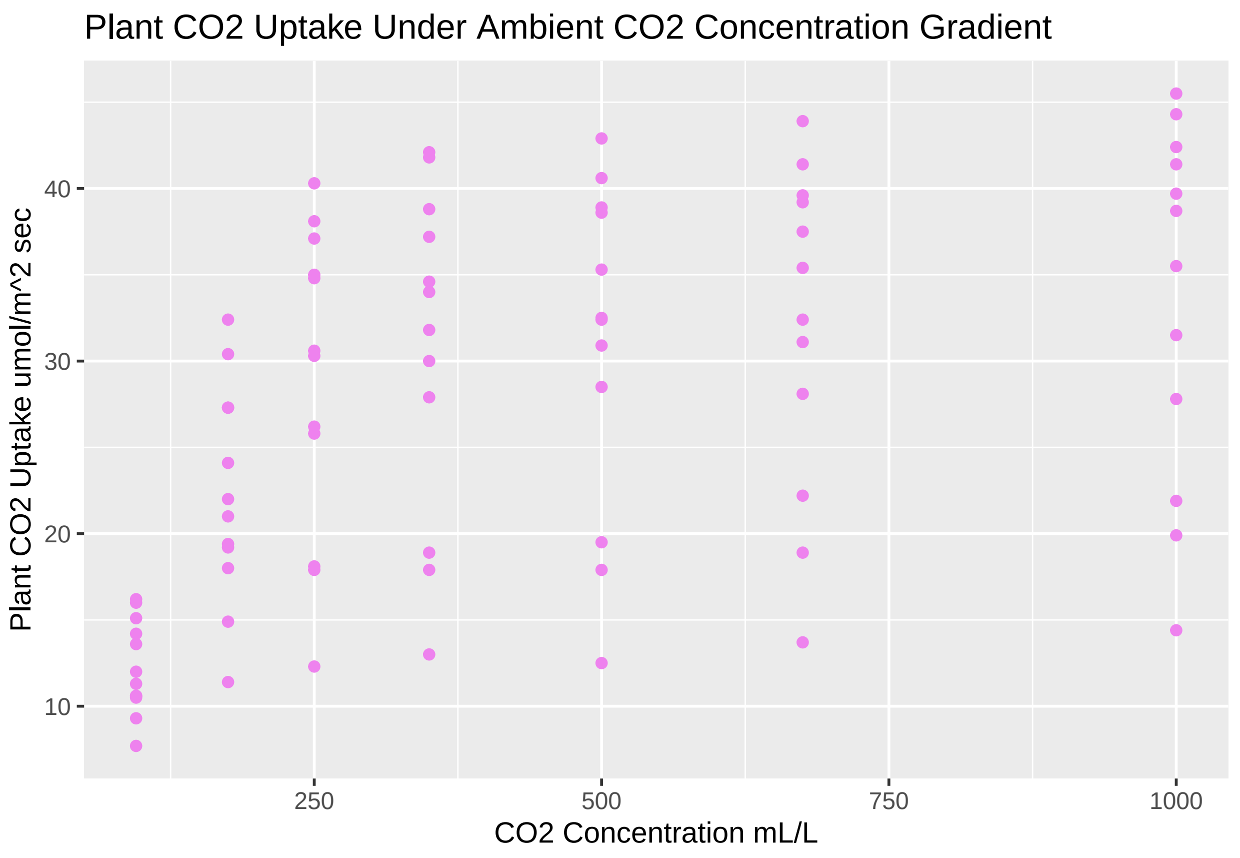
```
ggplot() +

  geom_point(data = CO2, aes(x = conc, y = uptake), color = 'violet') +

  ggtitle("Plant CO2 Uptake Under Ambient CO2 Concentration Gradient") +

  xlab("CO2 Concentration mL/L") +

  ylab("Plant CO2 Uptake umol/m^2 sec")
```



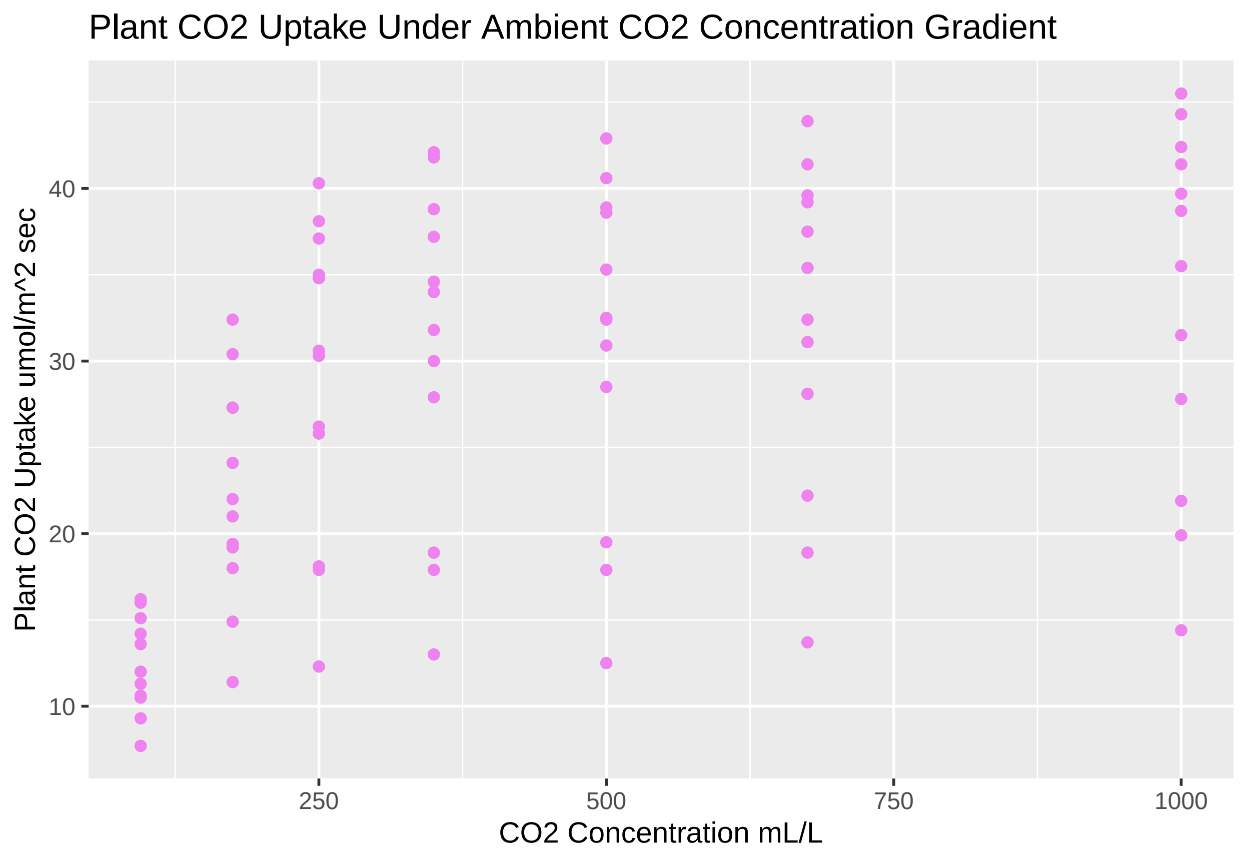
The ggplot() funtion initializes the plot. It can be empty, or you can use it to set the default data and aes from the beginning. Then add to this base plot specification using a geom function to plot points, lines, etc. You can also add other data frames, new colors, etc. on top of that.

Same as:

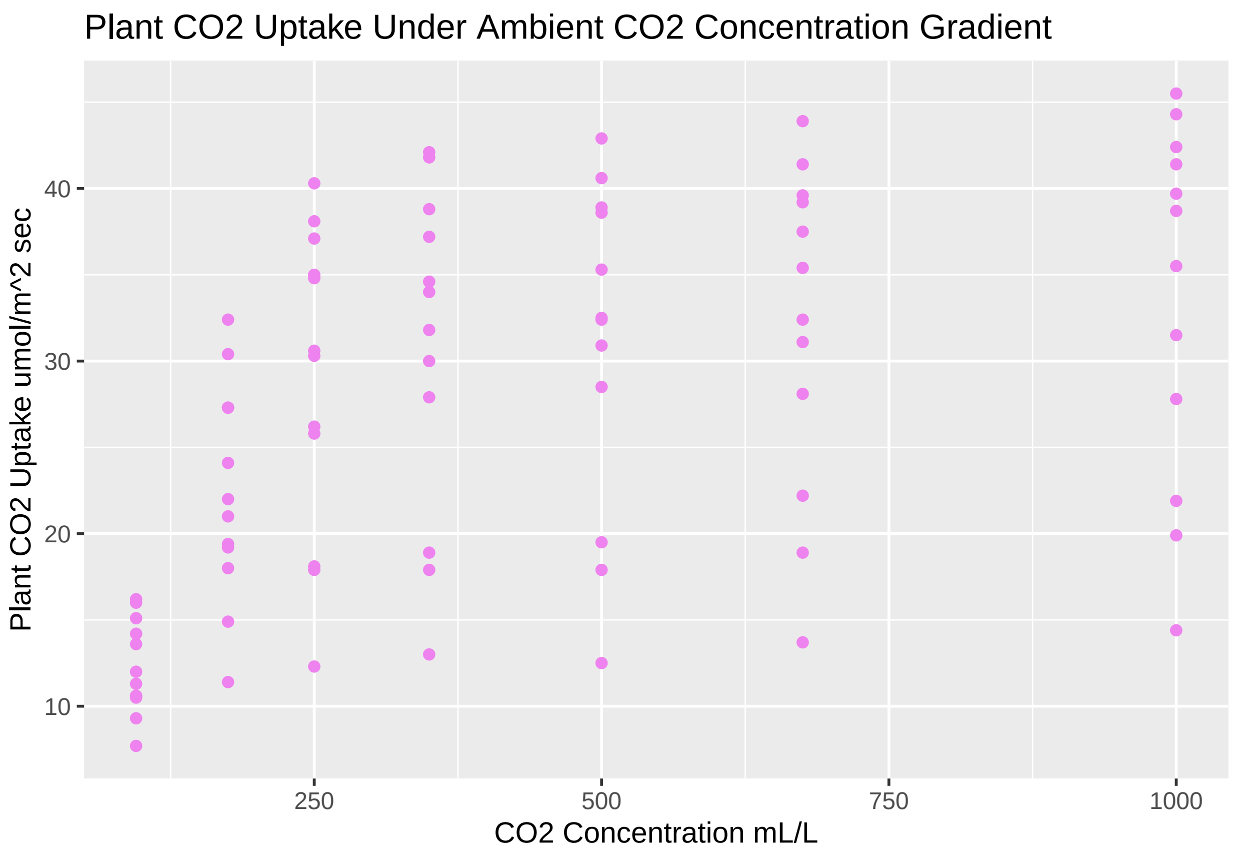
```r
ggplot() +

  geom_point(data = CO2, aes(x = conc, y = uptake), color = 'violet') +

  ggtitle("Plant CO2 Uptake Under Ambient CO2 Concentration Gradient") +

  xlab("CO2 Concentration mL/L") + ylab("Plant CO2 Uptake umol/m^2 sec")
```



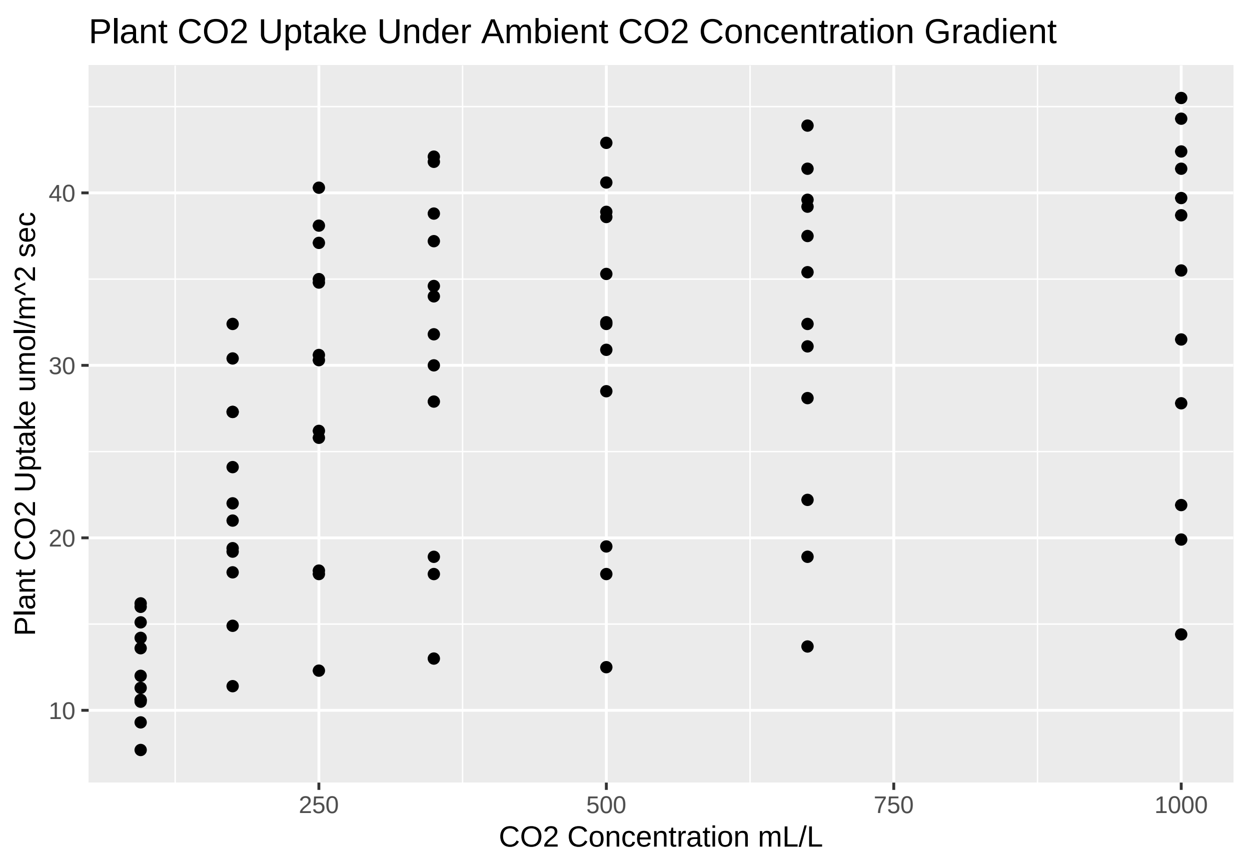
```r
ggplot(data = CO2) +

  geom_point(aes(x = conc, y = uptake), color = 'violet') +

  ggtitle("Plant CO2 Uptake Under Ambient CO2 Concentration Gradient") +

  xlab("CO2 Concentration mL/L") + ylab("Plant CO2 Uptake umol/m^2 sec")
```



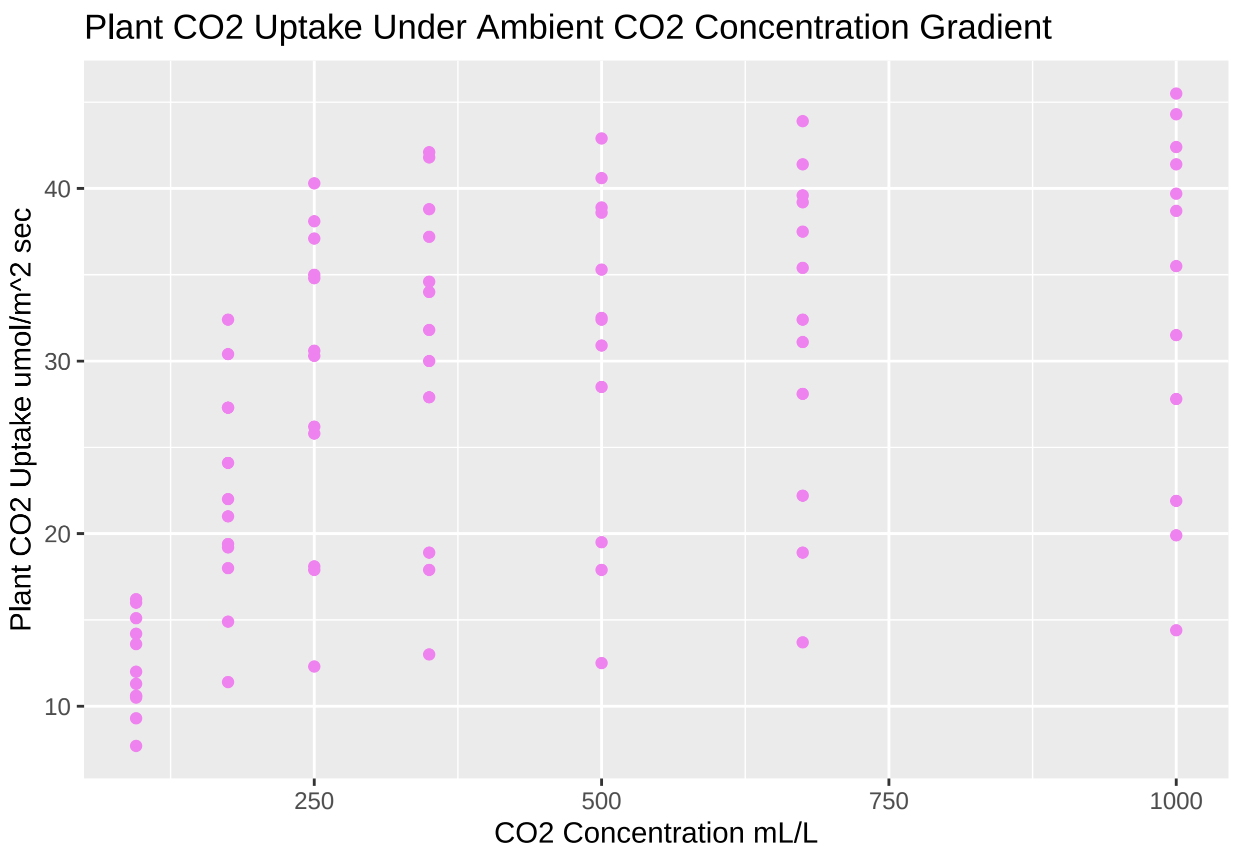
```
ggplot(data = CO2,  aes(x = conc, y = uptake)) +

  geom_point(color = 'violet') +

  ggtitle("Plant CO2 Uptake Under Ambient CO2 Concentration Gradient") +

  xlab("CO2 Concentration mL/L") + ylab("Plant CO2 Uptake umol/m^2 sec")
```



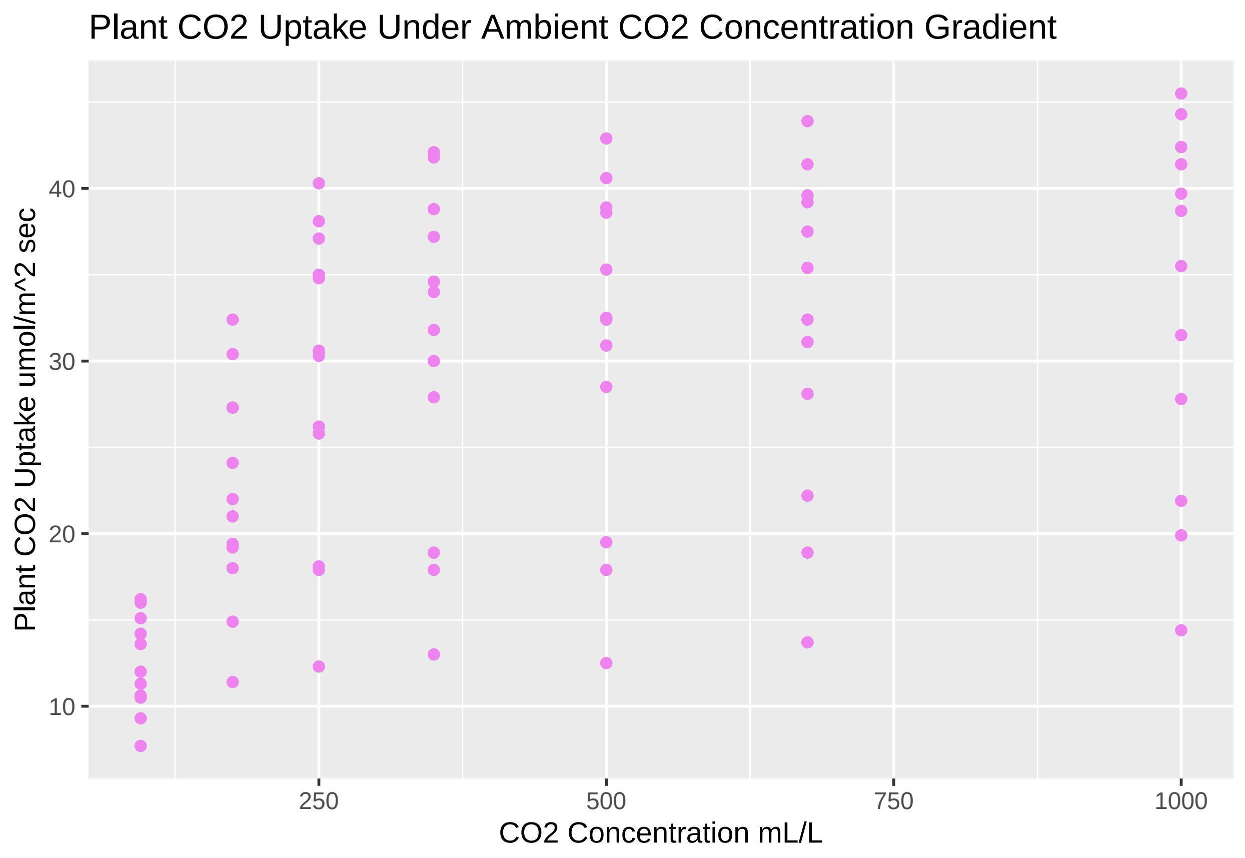
```
ggplot(data = CO2, aes(x = conc, y = uptake), color = 'violet') +

  geom_point() +

  ggtitle("Plant CO2 Uptake Under Ambient CO2 Concentration Gradient") +

  xlab("CO2 Concentration mL/L") + ylab("Plant CO2 Uptake umol/m^2 sec")
```



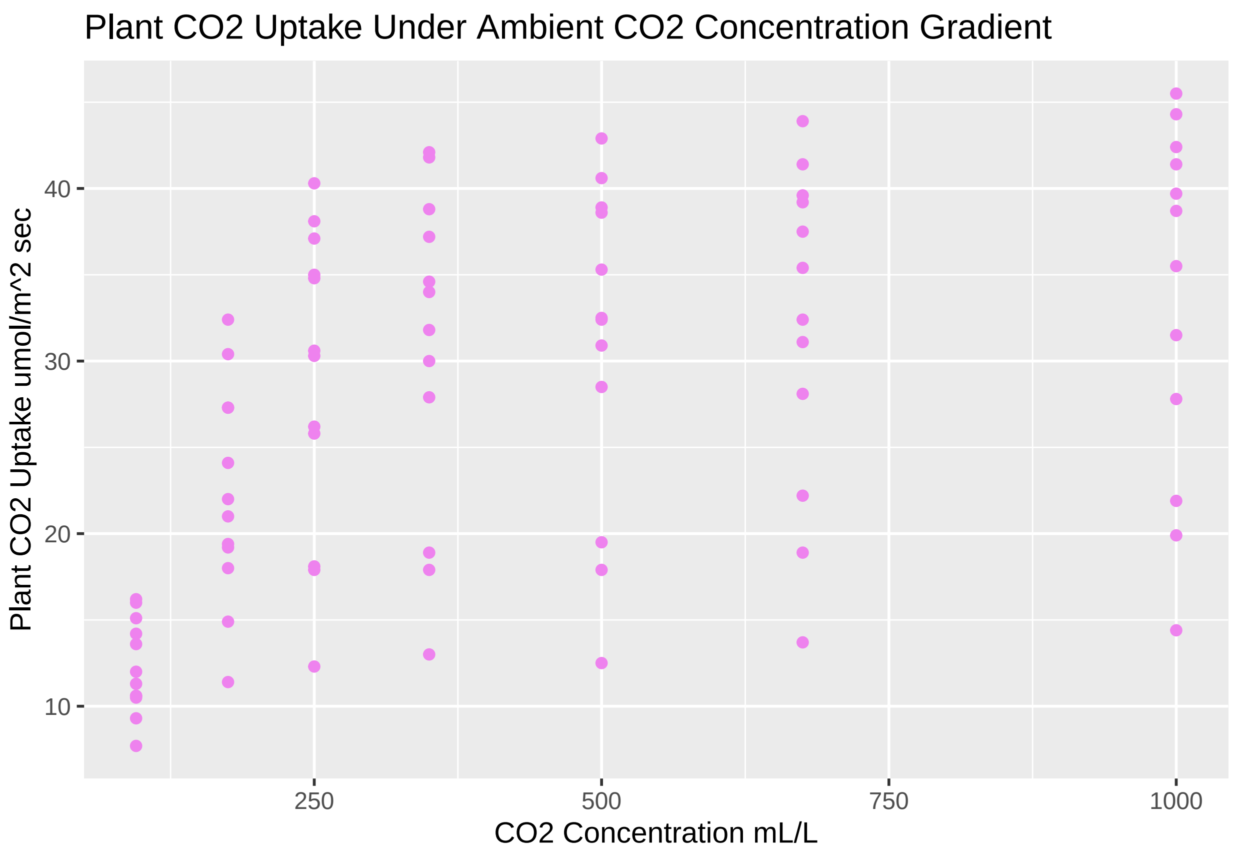
```
p <- ggplot()

p + geom_point(data = CO2, aes(x = conc, y = uptake), color = 'violet') +

  ggtitle("Plant CO2 Uptake Under Ambient CO2 Concentration Gradient") +

  xlab("CO2 Concentration mL/L") + ylab("Plant CO2 Uptake umol/m^2 sec")
```



```r
p <- ggplot(data = CO2)

p + geom_point(aes(x = conc, y = uptake), color = 'violet') +

  ggtitle("Plant CO2 Uptake Under Ambient CO2 Concentration Gradient") +

  xlab("CO2 Concentration mL/L") + ylab("Plant CO2 Uptake umol/m^2 sec")
```

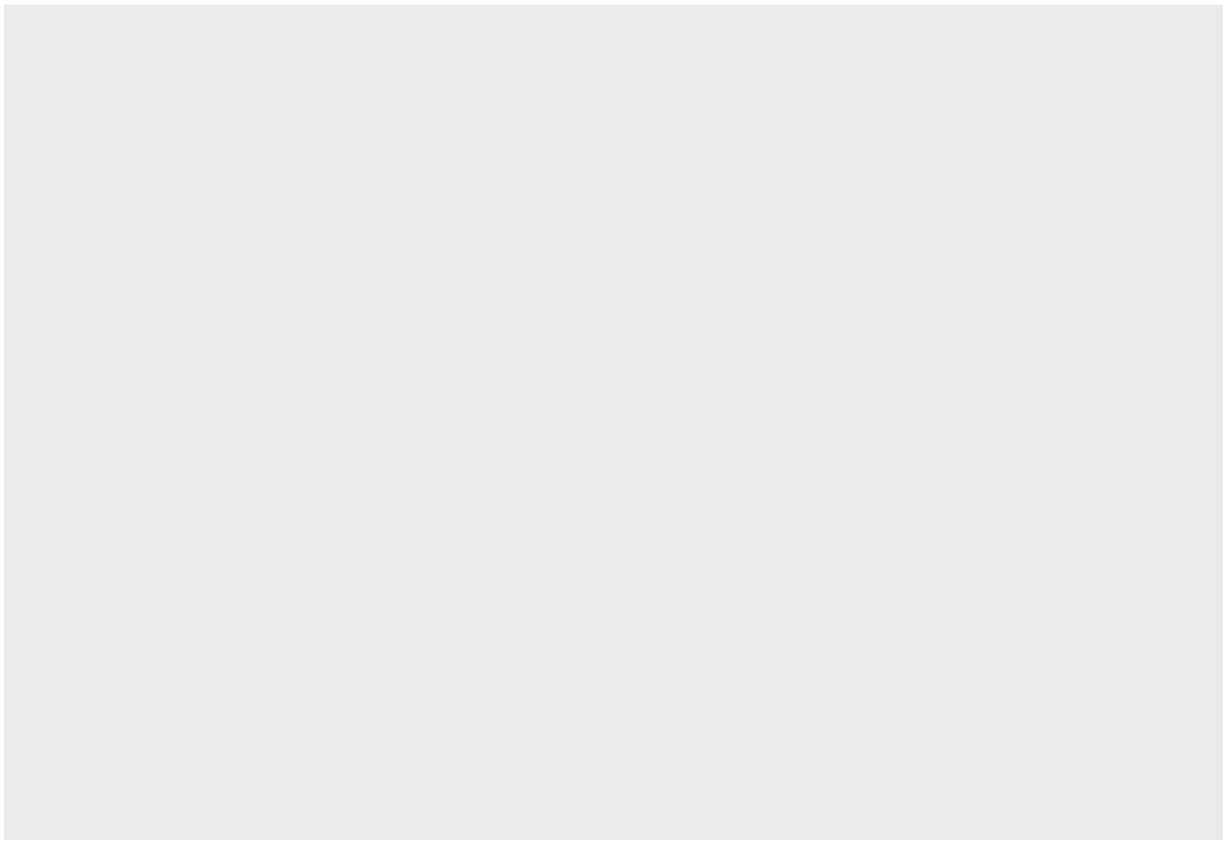


```
p <- ggplot(data = CO2)

q <- p + geom_point(aes(x = conc, y = uptake), color = 'violet')

q  +  ggtitle("Plant CO2 Uptake Under Ambient CO2 Concentration Gradient") +

  xlab("CO2 Concentration mL/L") + ylab("Plant CO2 Uptake umol/m^2 sec")
```

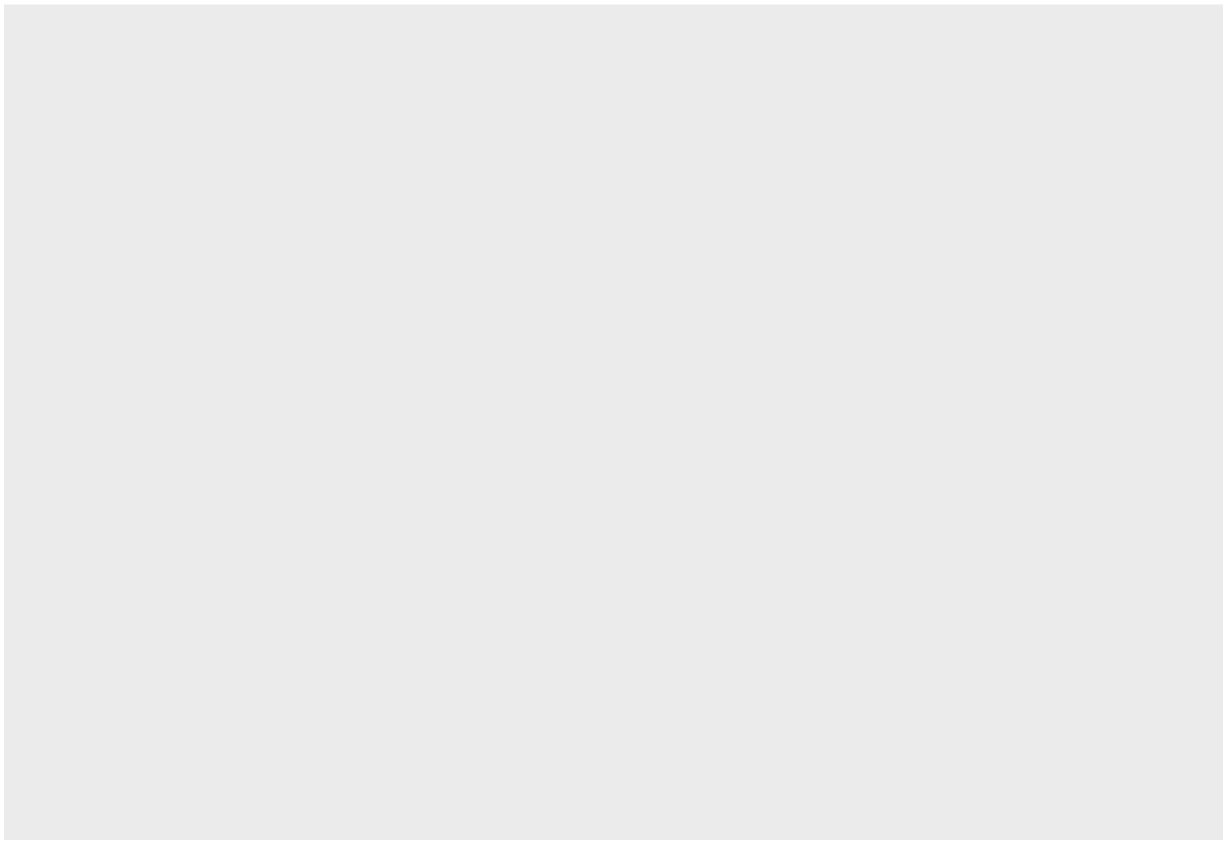


What happens if you just use ggplot() without a geom?

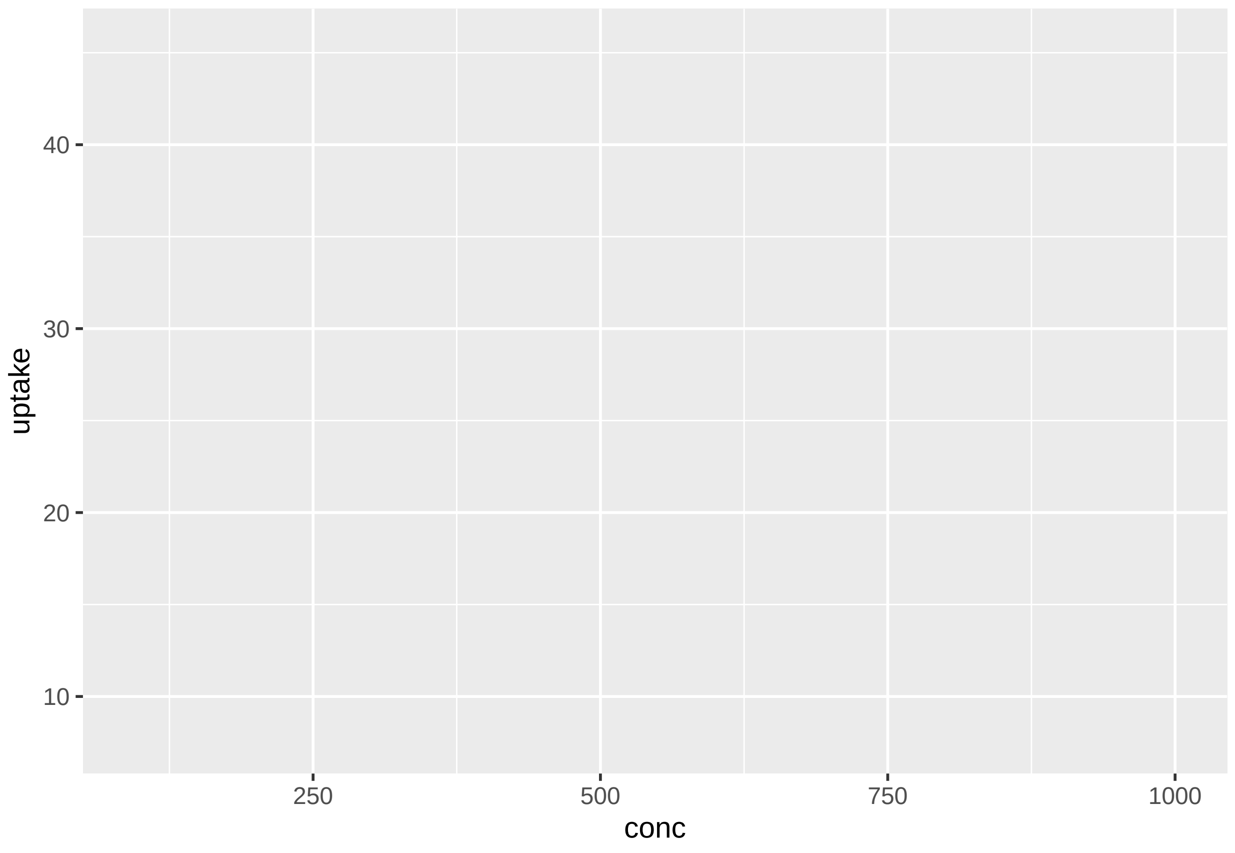
```r
ggplot()
```



```r
ggplot(data = CO2)
```



```r
ggplot(data = CO2,  aes(x = conc, y = uptake))
```



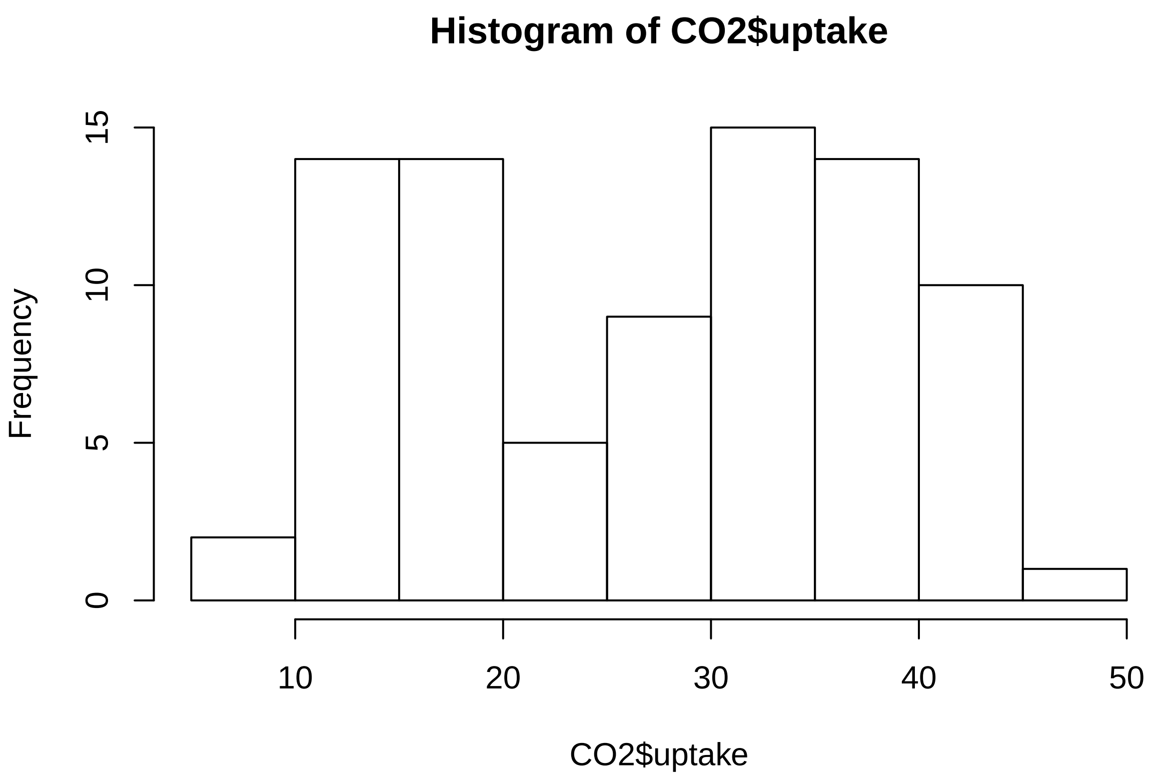
Different Plot Types

Plotting univariate data

Histograms - groups the data into bins (default or custom defined) spanning the range of the data, and displays the frequency of each bin.

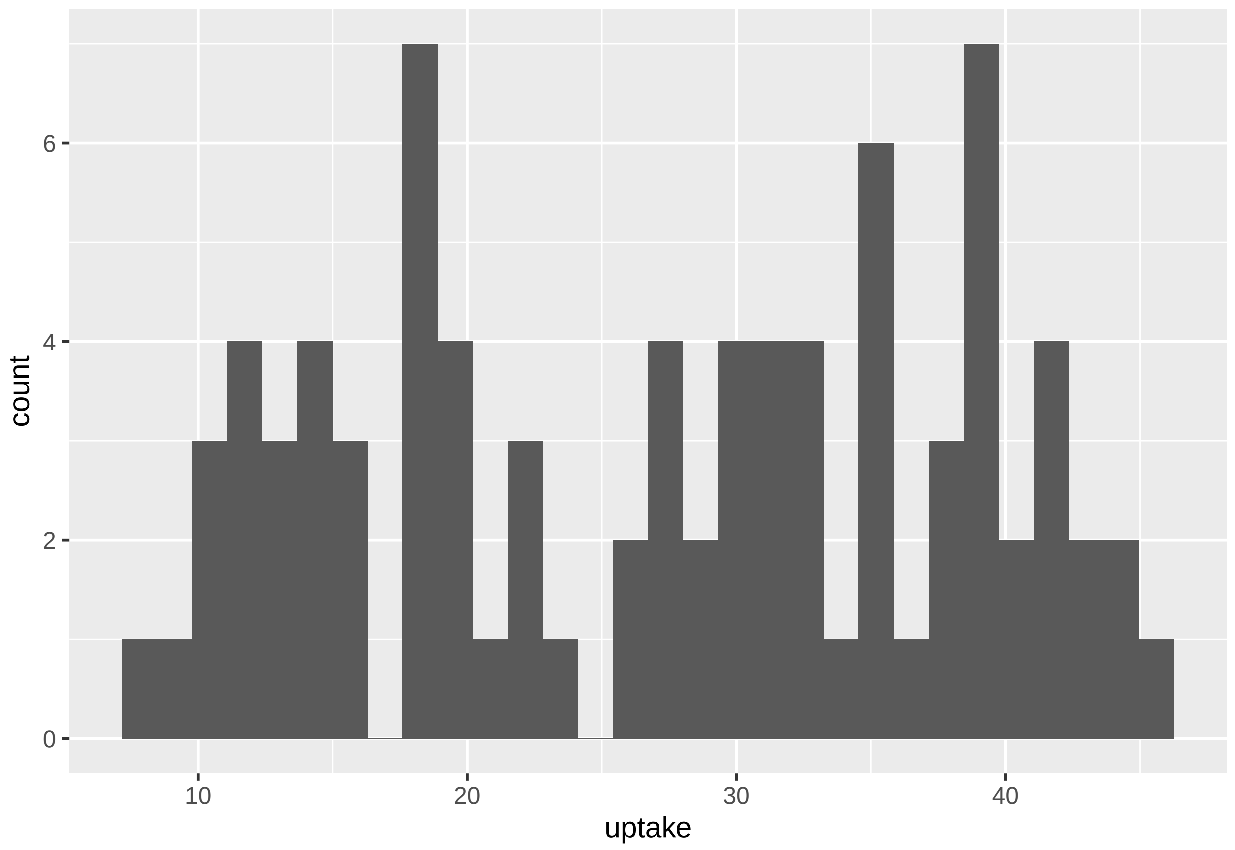
Examples:

```
hist(CO2$uptake)
hist(CO2$uptake,
     breaks = 10)
```



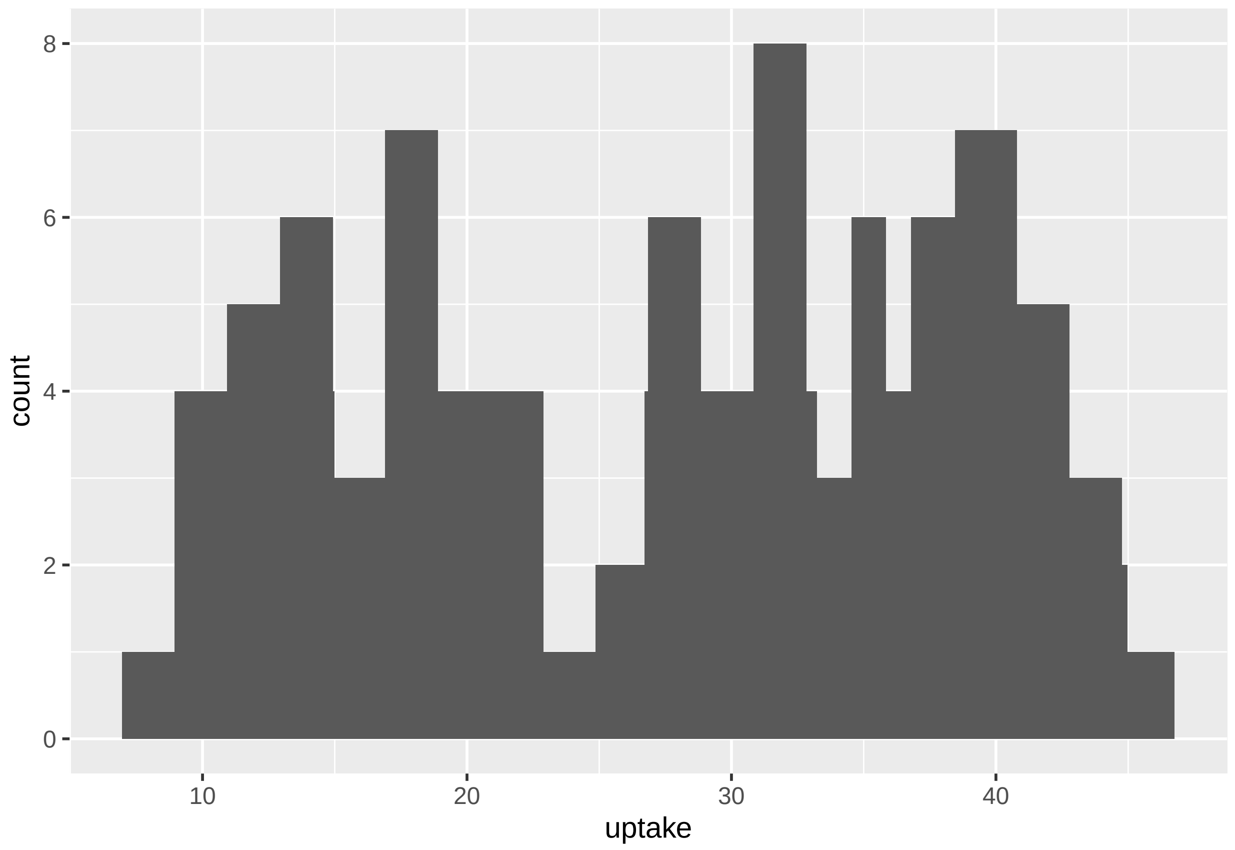
```
# Since we'll be using uptake as an example, let's set a base ggplot called p.uptake:

p.uptake <- ggplot(data = CO2, aes(uptake))

p.hist <- p.uptake +
  geom_histogram()

# Note, once you save a full data, aes, geom set to an object, you can print plot now by just typing ob
p.hist
```

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



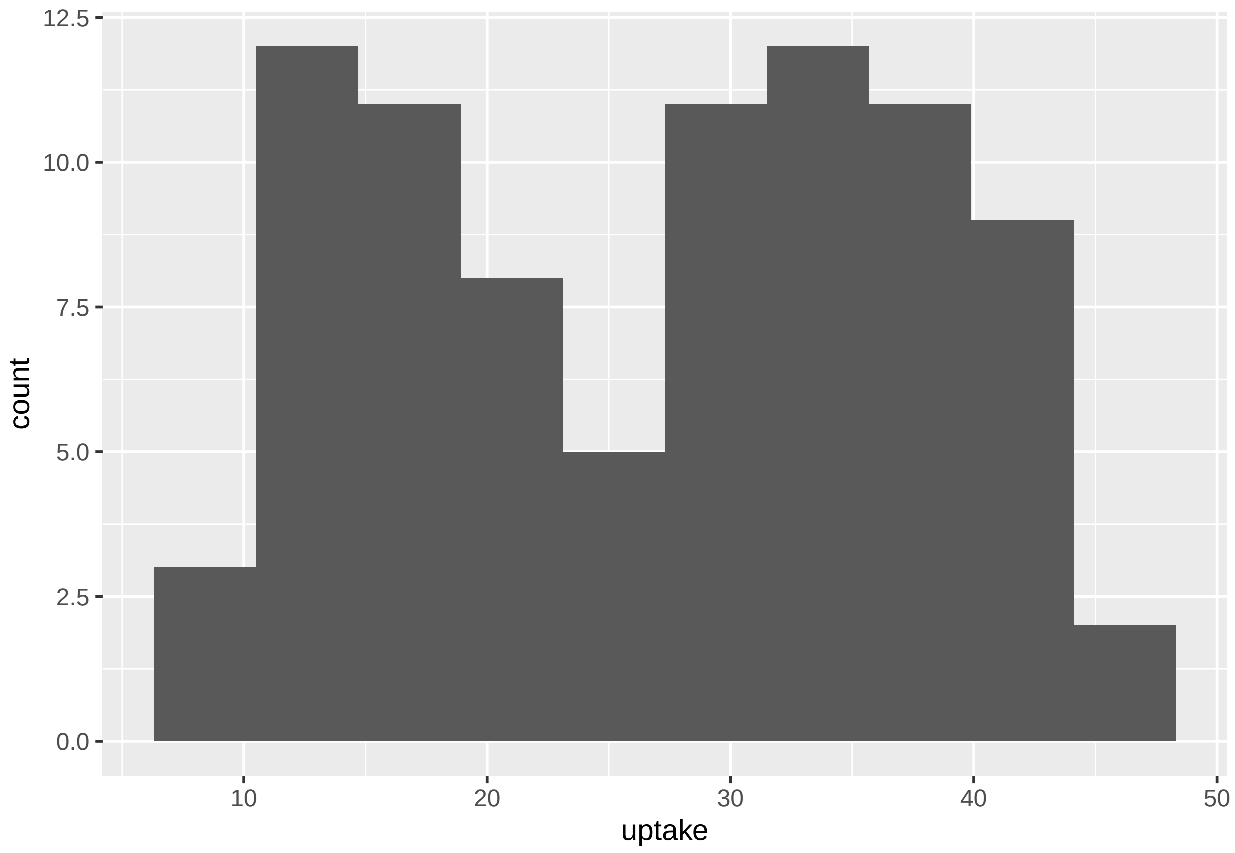
```
# and change the number of bins:
p.hist +  stat_bin(bins = 20)
```

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

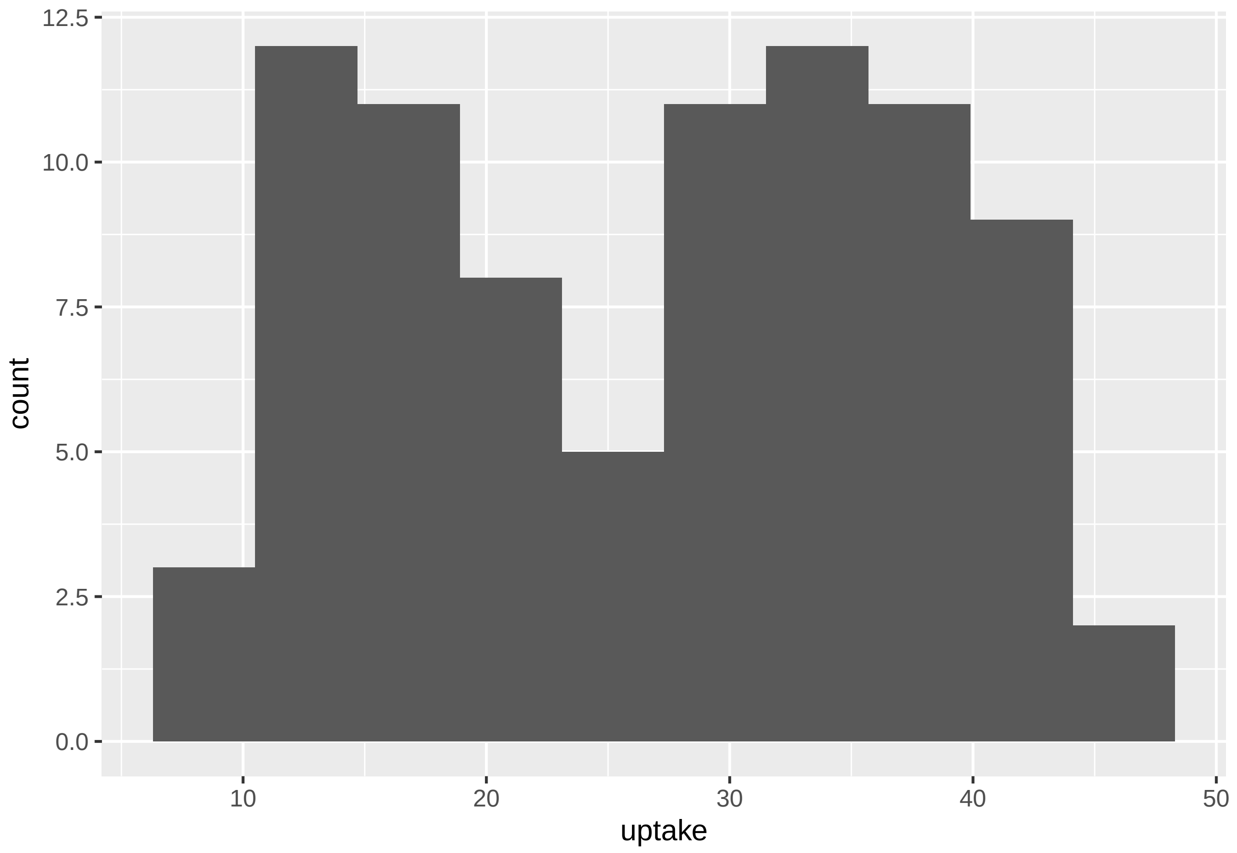


```r
p.hist +  stat_bin(bins = 10)
```

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



```r
# or set a binwidth from the beginning:

ggplot(data = CO2, aes(uptake)) +
  geom_histogram(binwidth = 4.2)
```



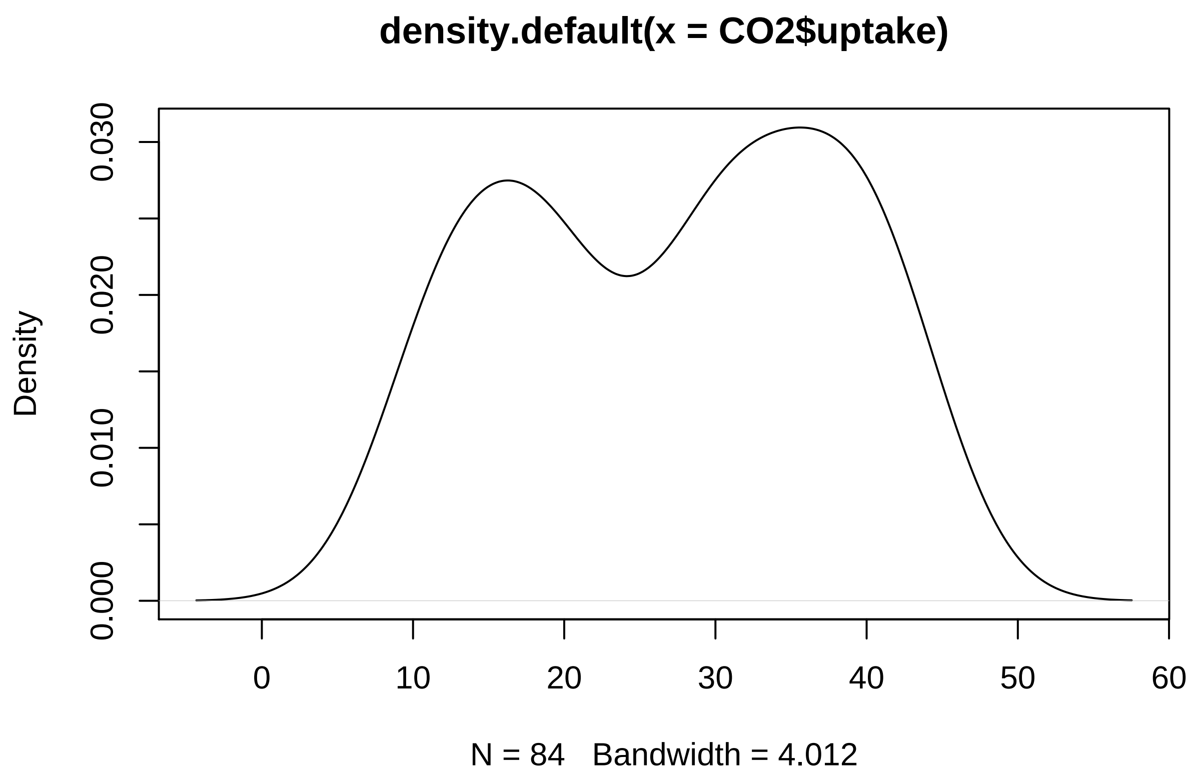
```
# https://www.qimacros.com/histogram-excel/how-to-determine-histogram-bin-interval/
```

Density plots - Smooths univariate data into a continuous density along its range.

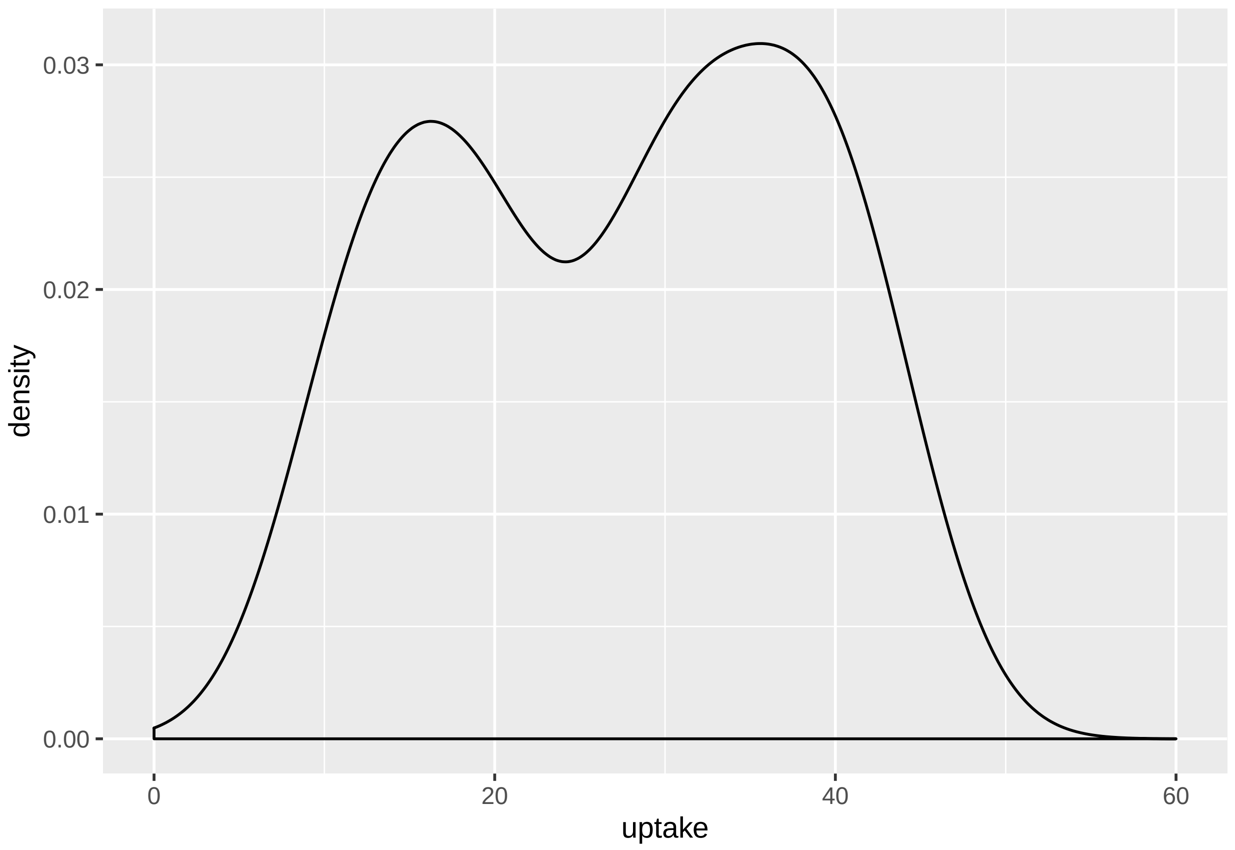
```
?density # Combine with plot() to vizualize; i.e. plot(density(x))
```

Example:

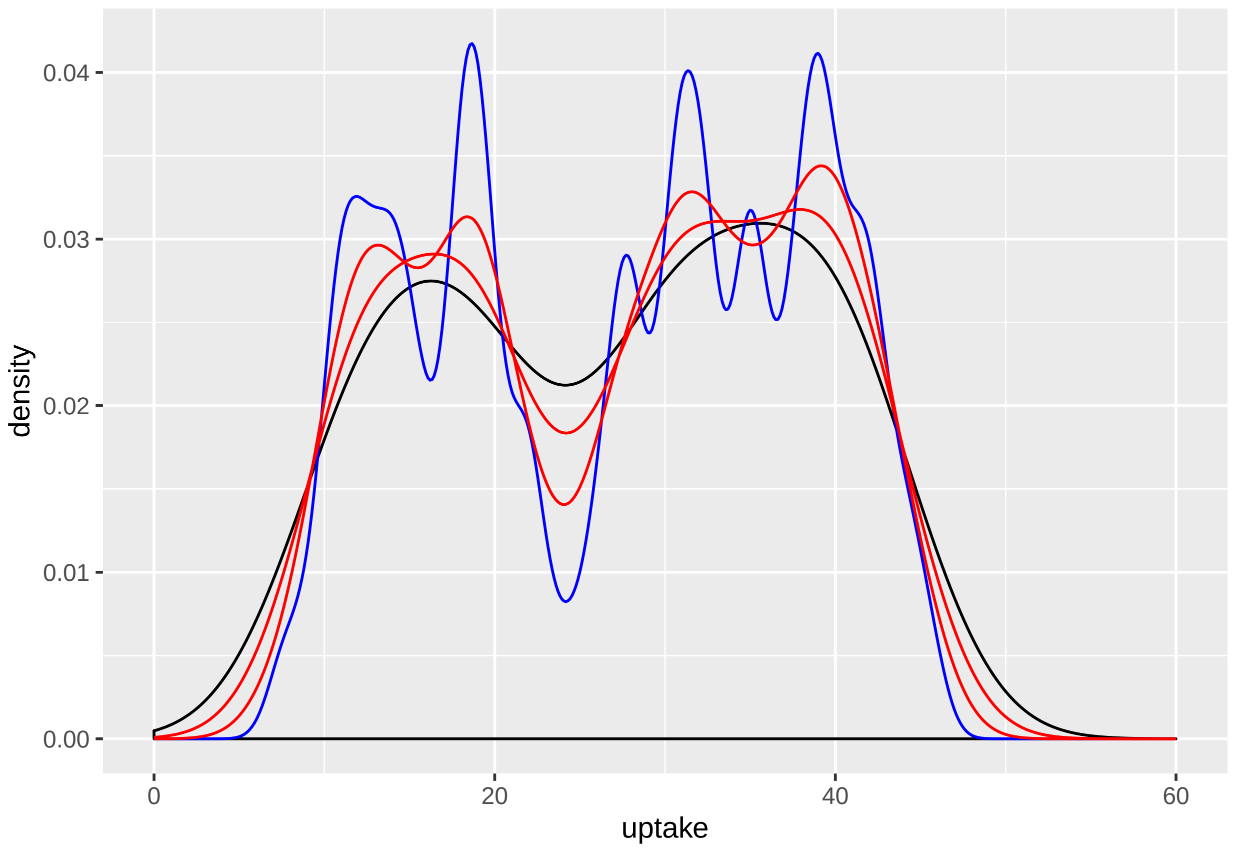
```
plot(density(CO2$uptake))
```



```
p.uptake +
  geom_density() +
  xlim(0, 60)
```



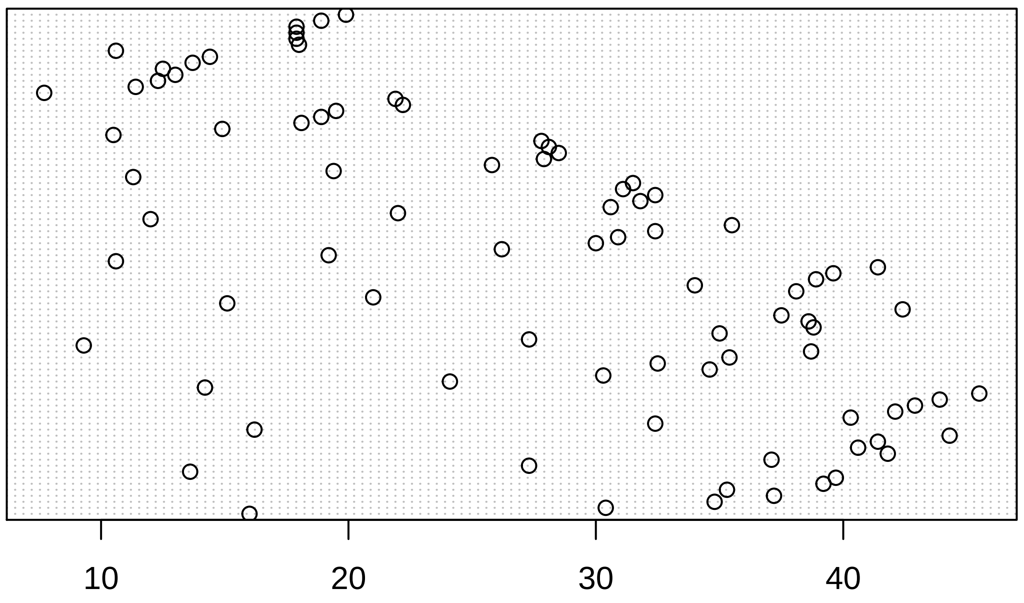
```
# explore different bandwiths:

p.uptake +
  geom_density() +
  stat_density(bw = 1,
               geom = "line",
               col = "blue") +
  stat_density(bw = 2,
               geom = "line",
               col = "red") +
    stat_density(bw = 3,
               geom = "line",
               col = "red") +
  xlim(0, 60)
```



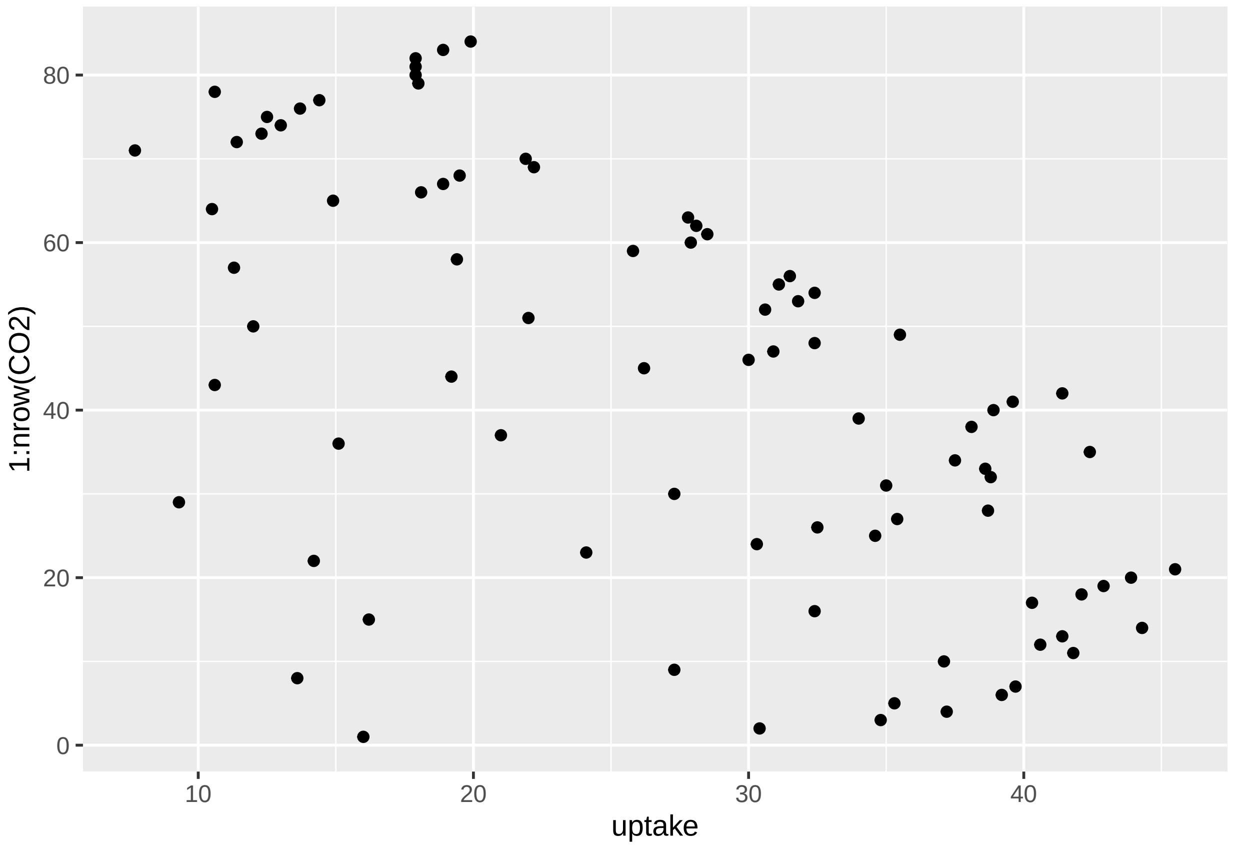
Dotcharts - Displays all point values along one dimension. The X-axis corresponds to the value you are plotting, while the Y-axis separates out the different data points

Examples:

```
dotchart(CO2$uptake)
```

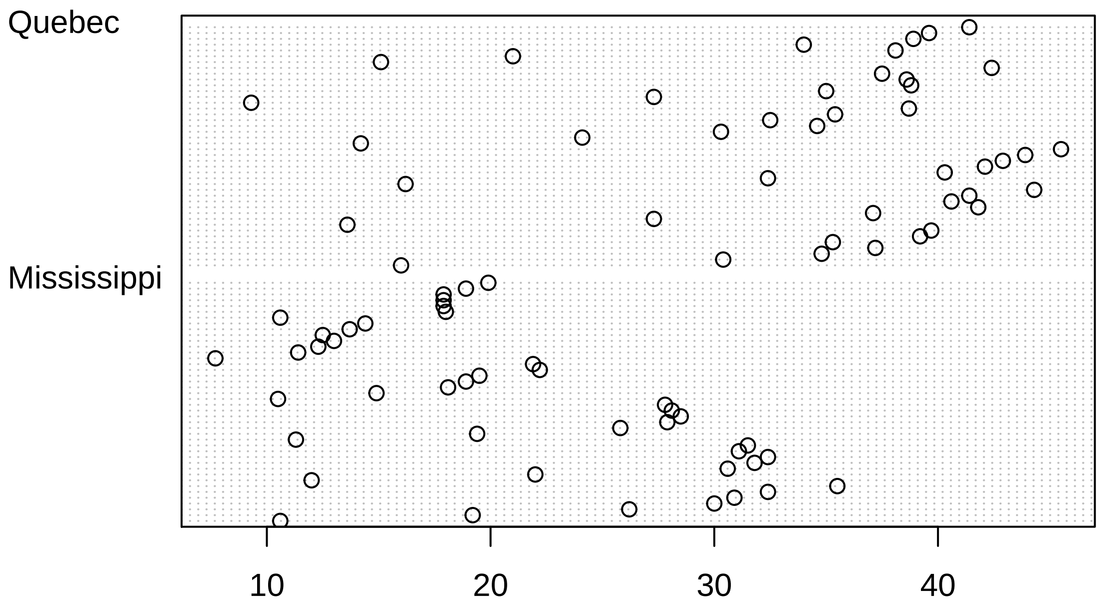


```r
# create your own Y-axis index to separate out data points using
# 1:No. rows in data frame.

ggplot(CO2) +
  geom_point(aes(x = uptake, y = 1:nrow(CO2)))
```

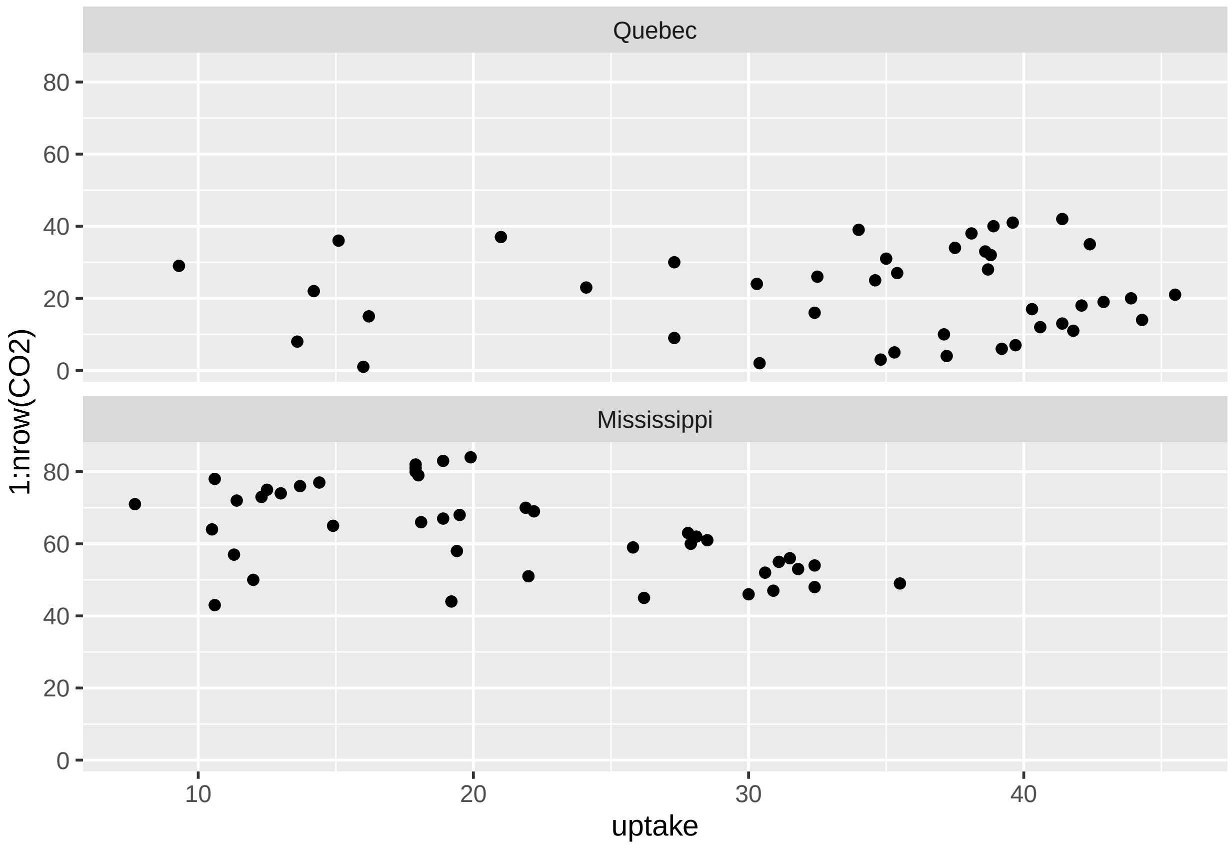


Note, you can also specify a grouping factor to divide the points and visually compare by group

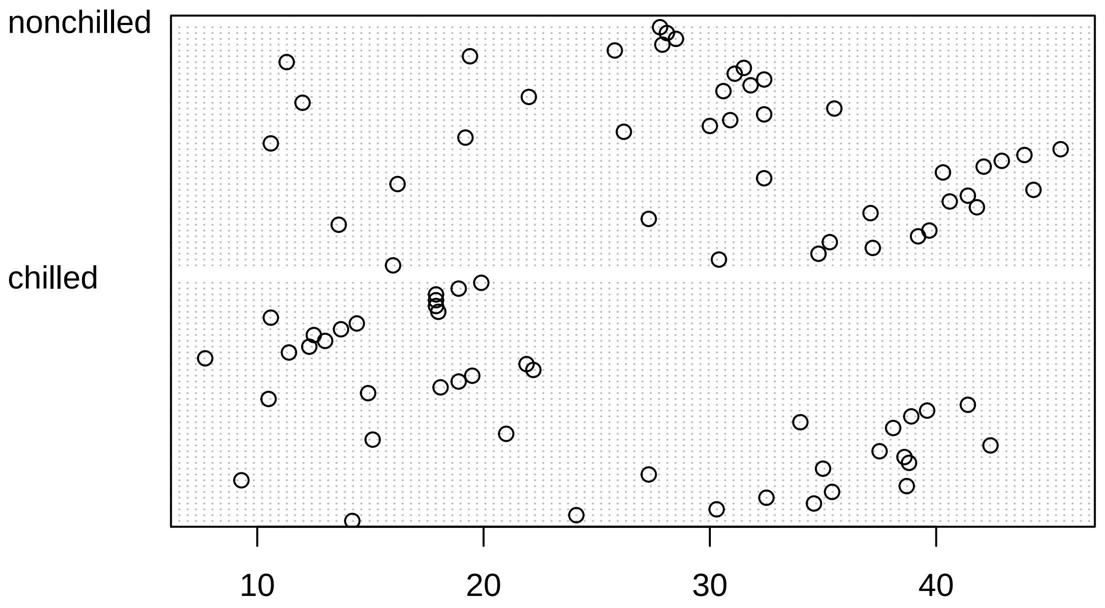
```r
# separate out data by plant origin

dotchart(CO2$uptake, groups = CO2$Type)
```



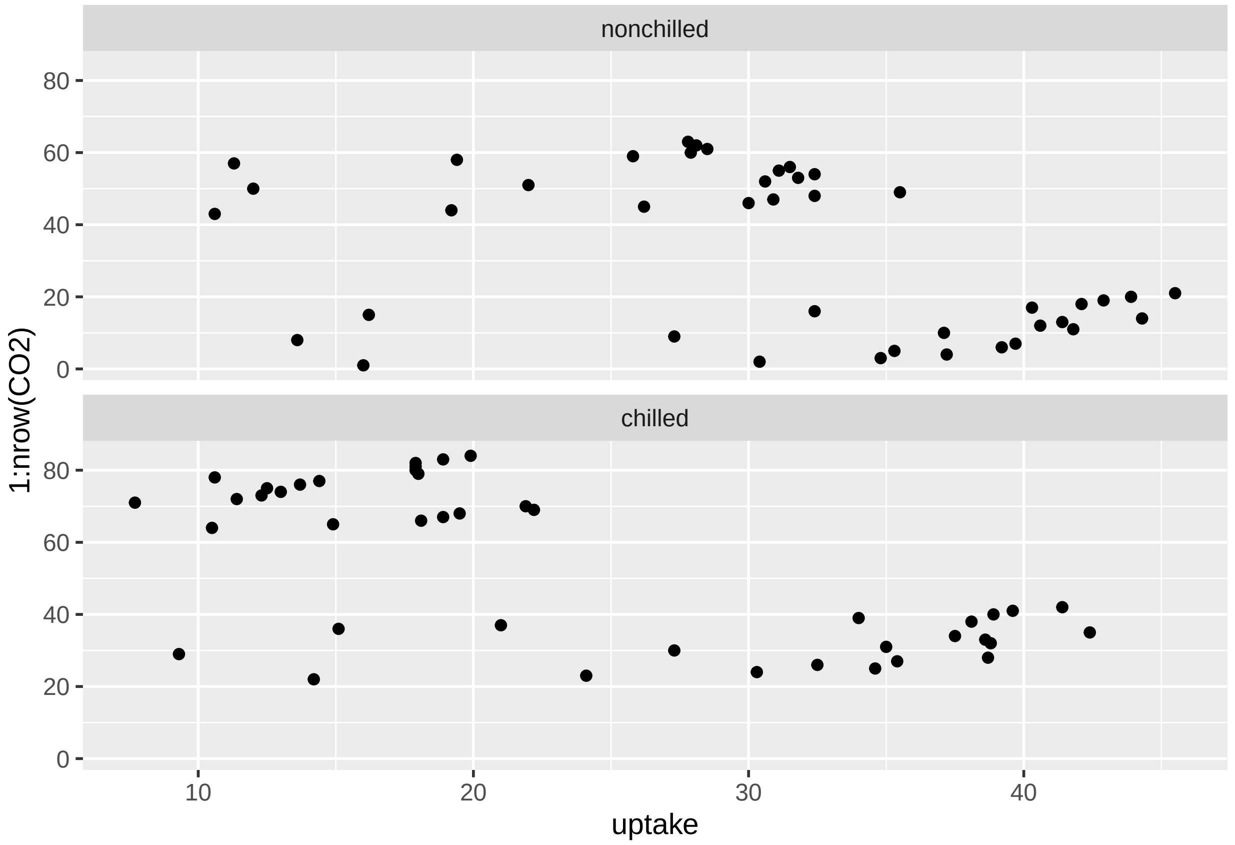
```r
ggplot(CO2, aes(x = uptake, y = 1:nrow(CO2))) +
  geom_point() +
  facet_wrap(~ Type, nrow = 2)
```



```r
# And do the same for the different treatments (chilled vs. non-chilled)
dotchart(CO2$uptake, groups = CO2$Treatment)
```

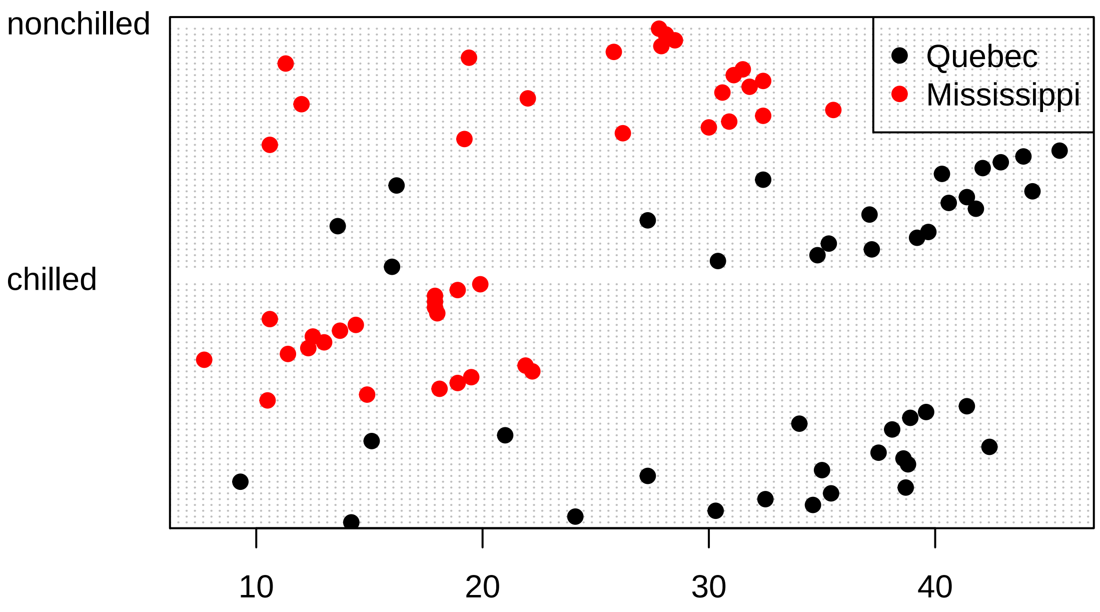


```r
CO2b <- rbind(CO2[CO2$Treatment == "chilled",],
              CO2[CO2$Treatment == "nonchilled",])

ggplot(CO2, aes(x = uptake, y = 1:nrow(CO2))) +
  geom_point() +
  facet_wrap(~ Treatment, nrow = 2)
```

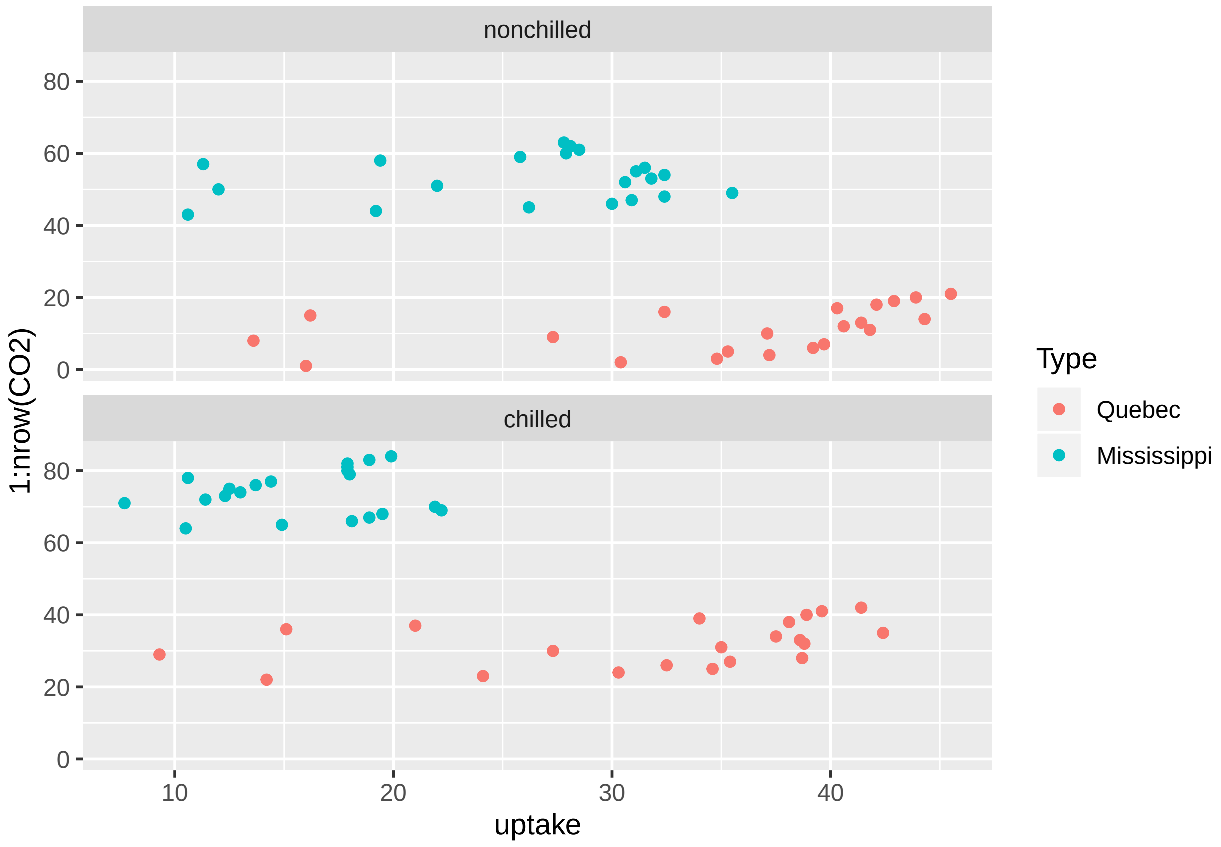


You might also be interested in what the data looks like for different plant origins when divided up by treatment. You can use the “col” argument to color by a factor level:

```
dotchart(CO2$uptake,
         groups = CO2$Treatment,
         col = CO2$Type,
         pch = 19)

legend(x = "topright",
       legend = levels(CO2$Type),
       col = 1:length(CO2$Type),
       pch = 19)
```



```
ggplot(CO2, aes(x = uptake, y = 1:nrow(CO2))) +
  geom_point(mapping = aes(col = Type)) +
  facet_wrap(~ Treatment, nrow = 2)
```

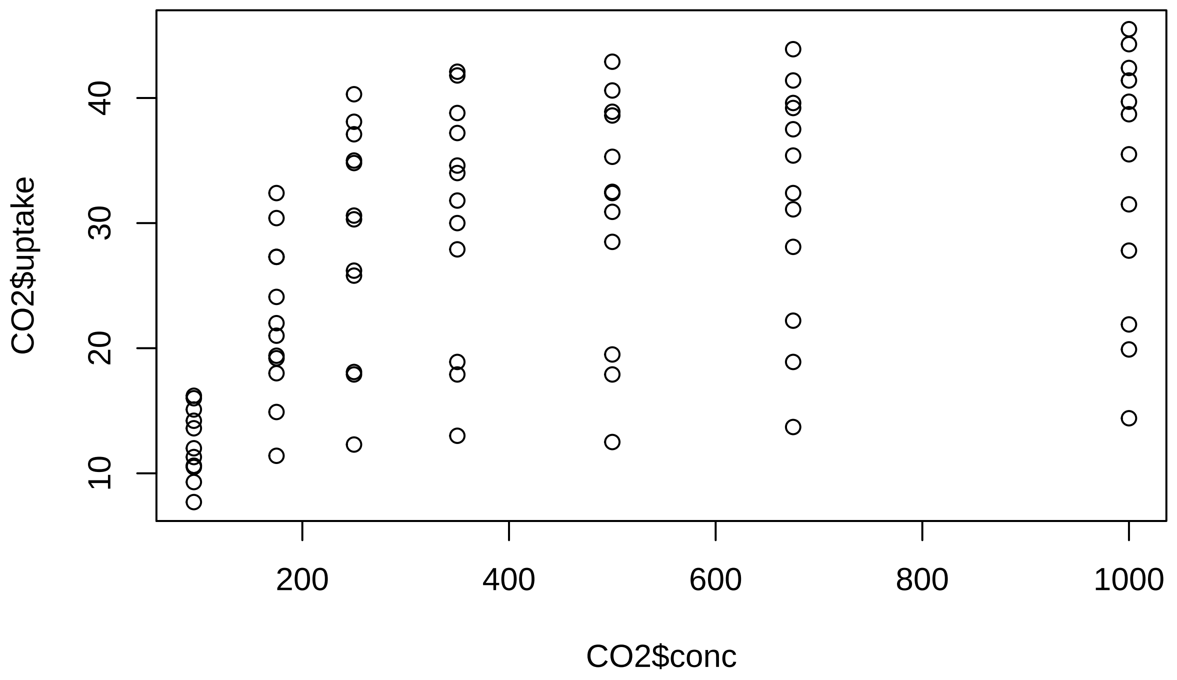


Plotting two or more variables

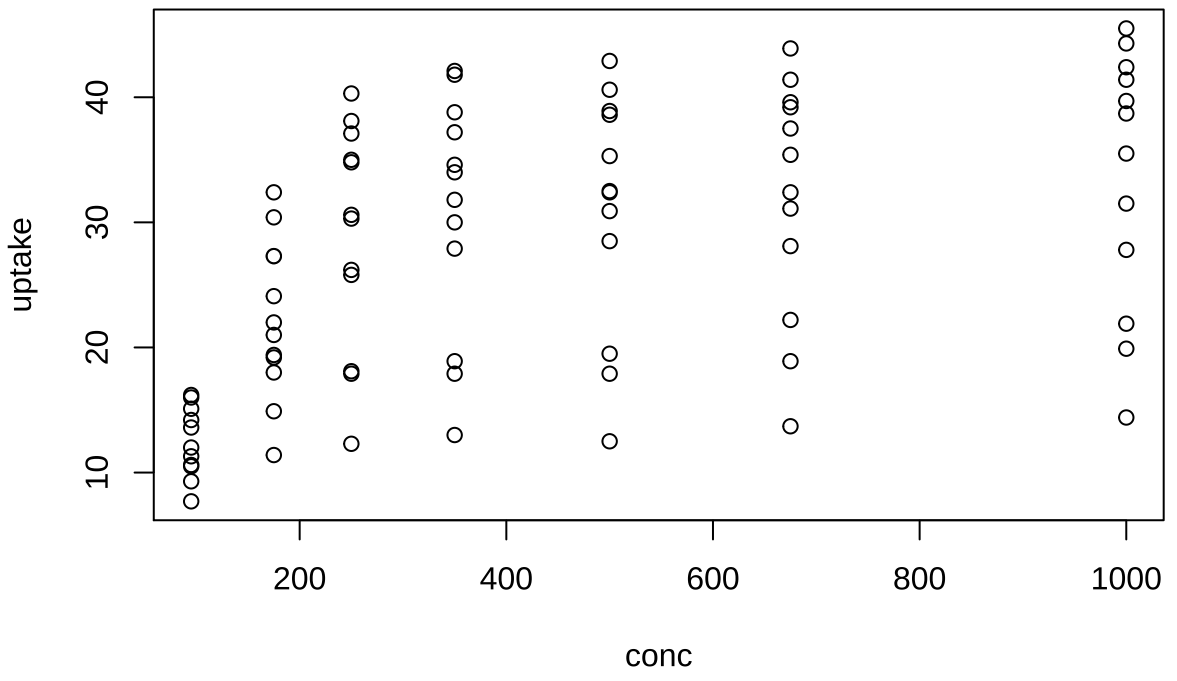
Scatterplots: 2 continuous variables

We can use a scatterplot to look at how uptake rate changes with ambient CO2 concentration:

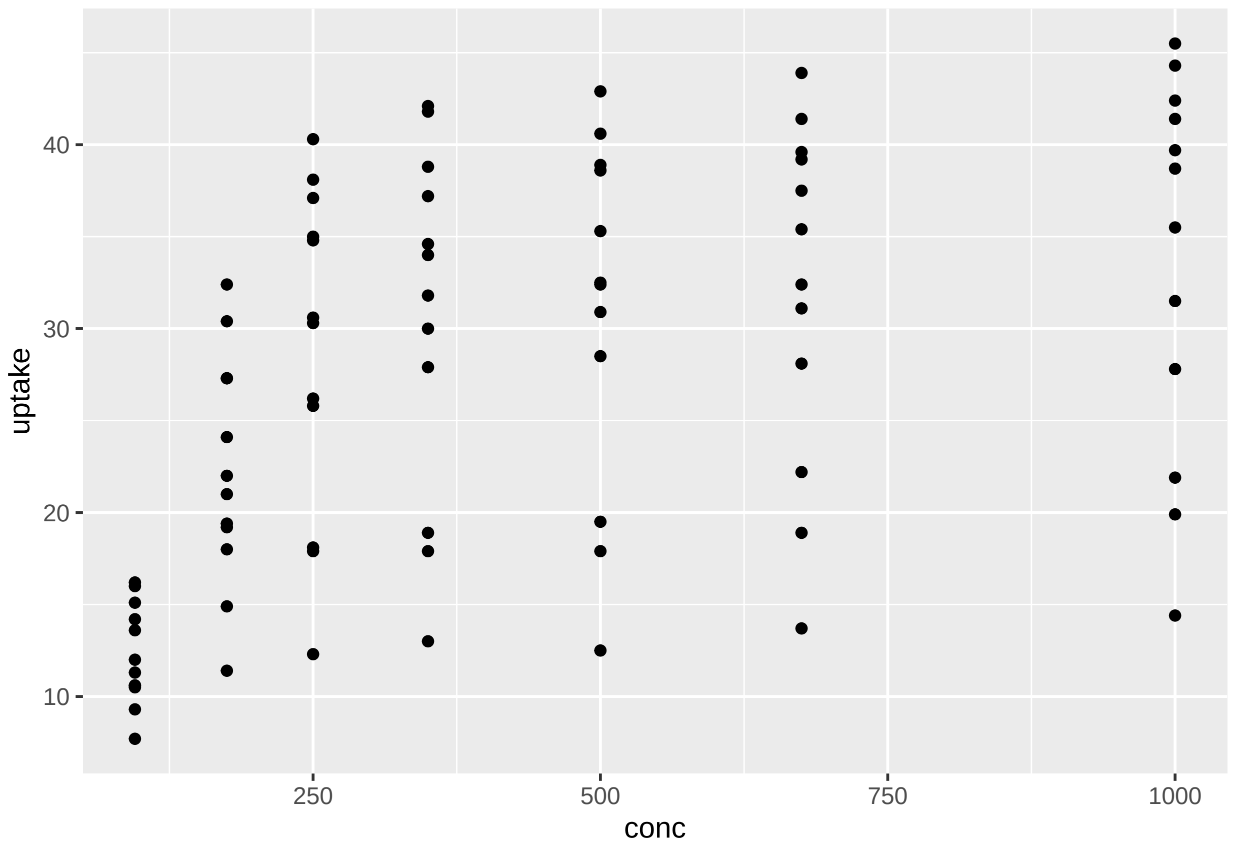
```r
plot(x = CO2$conc,
     y = CO2$uptake)
```



```r
# an alternative way of writing the above is:

plot(formula = uptake ~ conc,
     data = CO2)
```

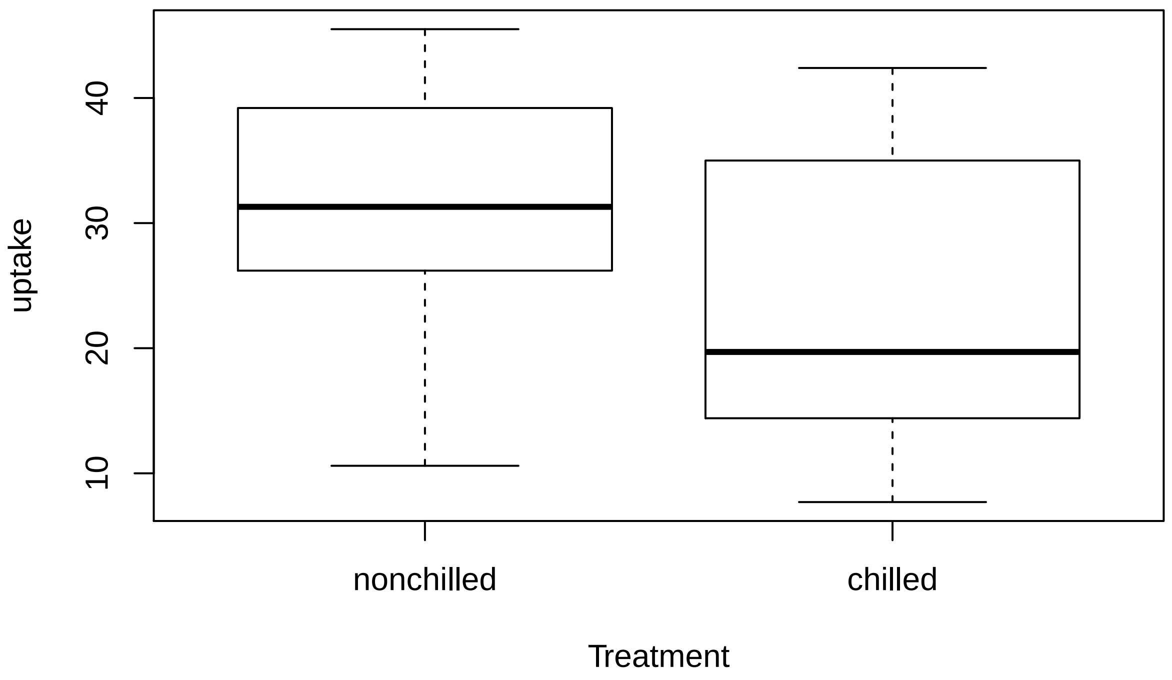


```r
ggplot() +
  geom_point(data = CO2, aes(conc, uptake))
```

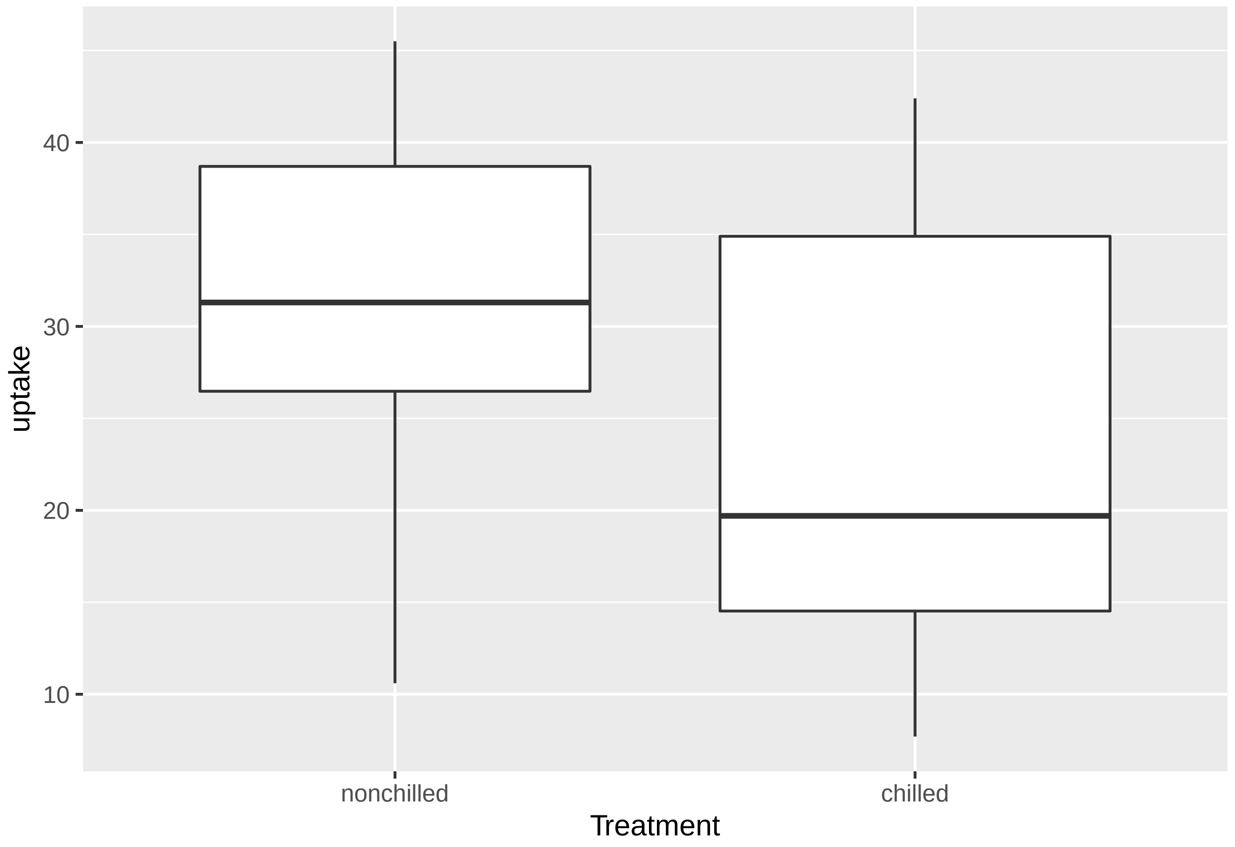


Box plots - Groups the data along discrete factors, and displays the median, interquartile range (whiskers), and outliers

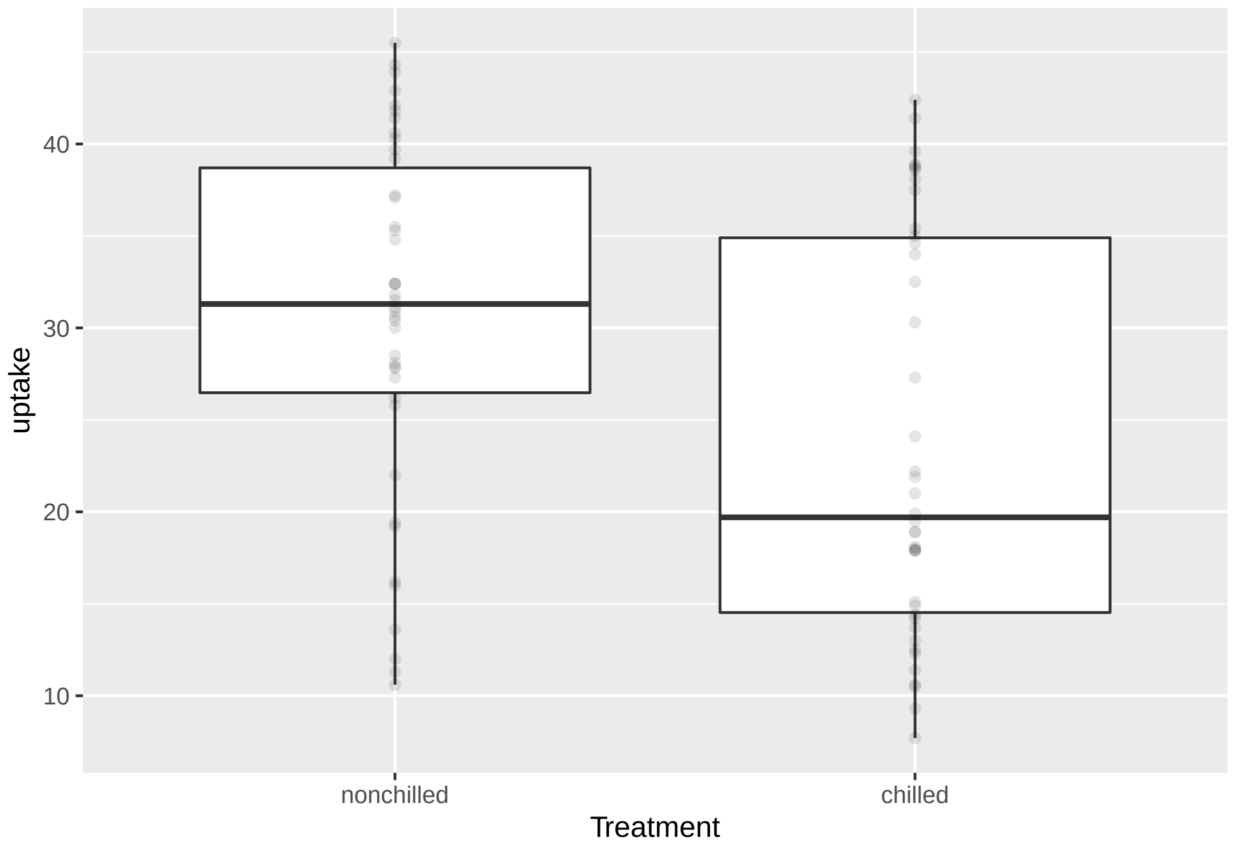
```
?boxplot
boxplot(formula = uptake ~ Treatment,
        data = CO2)

boxplot(formula = uptake ~ Treatment,
        data = CO2)
```



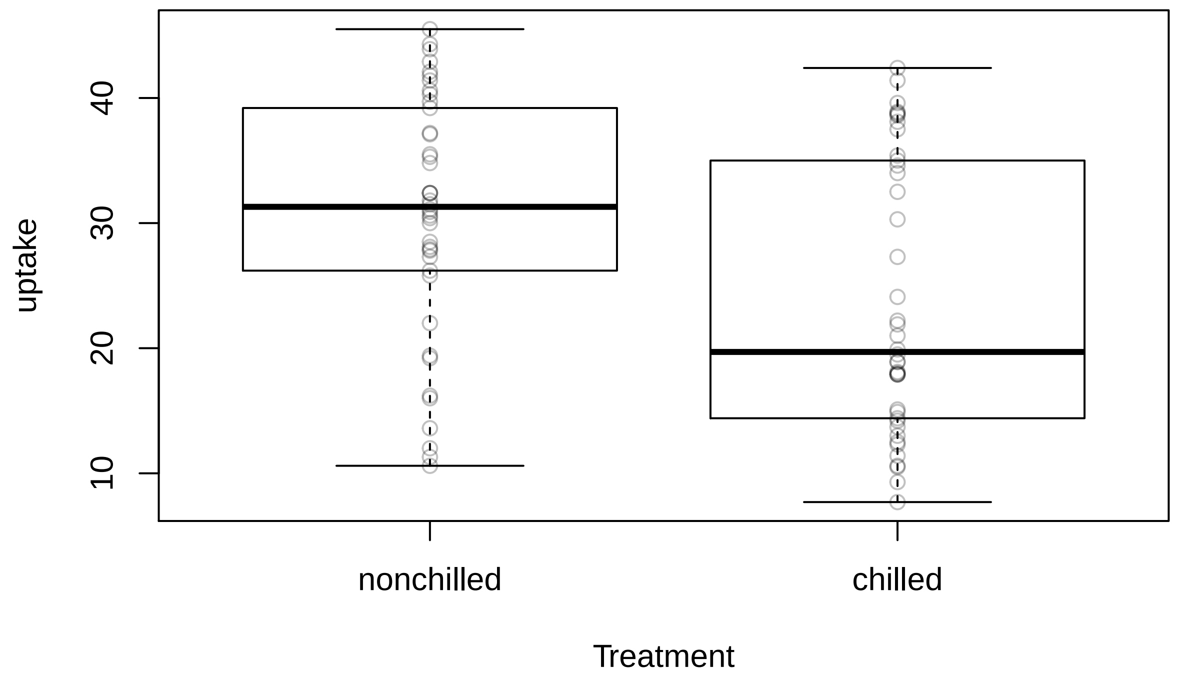
```r
# Do the same in ggplot:
ggplot() +
  geom_boxplot(data = CO2,
               aes(x = Treatment, y = uptake))
```



```r
# add original data points and use alpha to set transparency of points:
ggplot() +
  geom_boxplot(data = CO2,
               aes(x = Treatment, y = uptake)) +

  geom_point(data = CO2,
             aes(x = Treatment, y = uptake), alpha = 0.1)
```



```r
# You can do that in base R plot functions too, but would need more code to set transparency:


  t_col <- function(color, percent = 50, name = NULL) {
    rgb.val <- col2rgb(color)
    t.col <- rgb(rgb.val[1], rgb.val[2], rgb.val[3],
                 max = 255,
                 alpha = (100-percent)*255/100,
                 names = name)
    invisible(t.col)
  }


boxplot(formula = uptake ~ Treatment,
        data = CO2)
points(x = CO2$Treatment, y = CO2$uptake,
       col = t_col("black", percent = 75))
```


uptake
10
20
30
40
nonchilled
chilled
Treatment