Data Visualization

Jennifer Brazeal

9/5/19

Data visualization is an important tool for both data exploration and for communicating your results effectively to peers and to the public. Today we will show you some of the things you can do using base R plotting functions to explore the CO2 dataset. A package called ggplot2, part of a larger set of packages known as the "tidyverse," is popular for data visualization. We will introduce you to ggplot2 on the final day of class, but we first want to give you a good grounding in base R plotting functions.

Before we get started, let's save the CO2 data again from the internal `datasets` package as an object in our workspace. This dataset contains observations of CO2 uptake by plants originating from different locations (Quebec and Mississippi) and under different temperature treatments (chilled and not chilled) across different fixed ambient CO2 concentrations.

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

The head() function lets us take a quick look at the data - three columns of character vectors and two numeric columns:

```
head(CO2)
```

```
##   Plant   Type  Treatment conc uptake
## 1   Qn1 Quebec nonchilled   95   16.0
## 2   Qn1 Quebec nonchilled  175   30.4
## 3   Qn1 Quebec nonchilled  250   34.8
## 4   Qn1 Quebec nonchilled  350   37.2
## 5   Qn1 Quebec nonchilled  500   35.3
## 6   Qn1 Quebec nonchilled  675   39.2
```

str() shows us more specific information about the structure of the data:

```
str(CO2)
```

```
## Classes 'nfnGroupedData', 'nfGroupedData', 'groupedData' and 'data.frame':   84 obs. of  5 variables
##  $ Plant    : Ord.factor w/ 12 levels "Qn1"<"Qn2"<"Qn3"<..: 1 1 1 1 1 1 1 2 2 2 ...
##  $ Type     : Factor w/ 2 levels "Quebec","Mississippi": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Treatment: Factor w/ 2 levels "nonchilled","chilled": 1 1 1 1 1 1 1 1 1 1 ...
##  $ conc     : num  95 175 250 350 500 675 1000 95 175 250 ...
##  $ uptake   : num  16 30.4 34.8 37.2 35.3 39.2 39.7 13.6 27.3 37.1 ...
##  - attr(*, "formula")=Class 'formula'  language uptake ~ conc | Plant
##   .. ..- attr(*, ".Environment")=<environment: R_EmptyEnv>
##  - attr(*, "outer")=Class 'formula'  language ~Treatment * Type
##   .. ..- attr(*, ".Environment")=<environment: R_EmptyEnv>
##  - attr(*, "labels")=List of 2
##   ..$ x: chr "Ambient carbon dioxide concentration"
##   ..$ y: chr "CO2 uptake rate"
##  - attr(*, "units")=List of 2
##   ..$ x: chr "(uL/L)"
##   ..$ y: chr "(umol/m^2 s)"
```

Questions: what units are the uptake rate in?

Different Plot Types

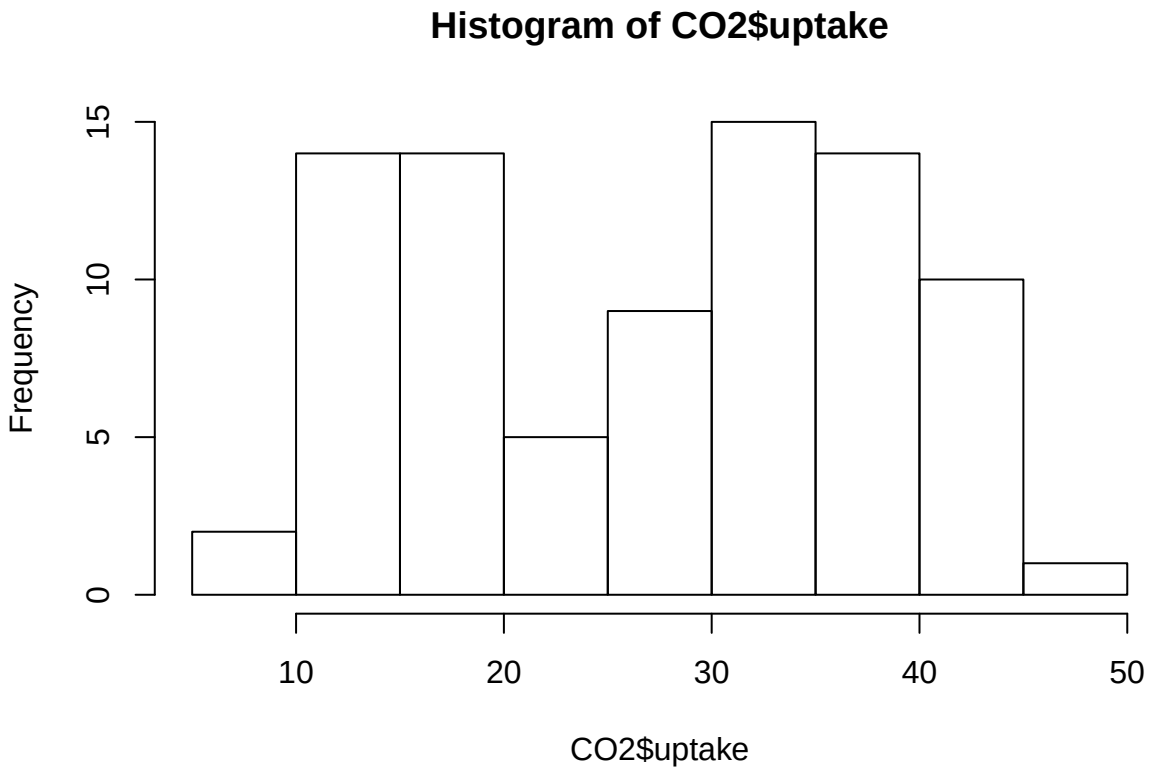
Plotting one continuous variable

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

```
?hist
```

Examples:

```
hist(CO2$uptake)
```



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

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

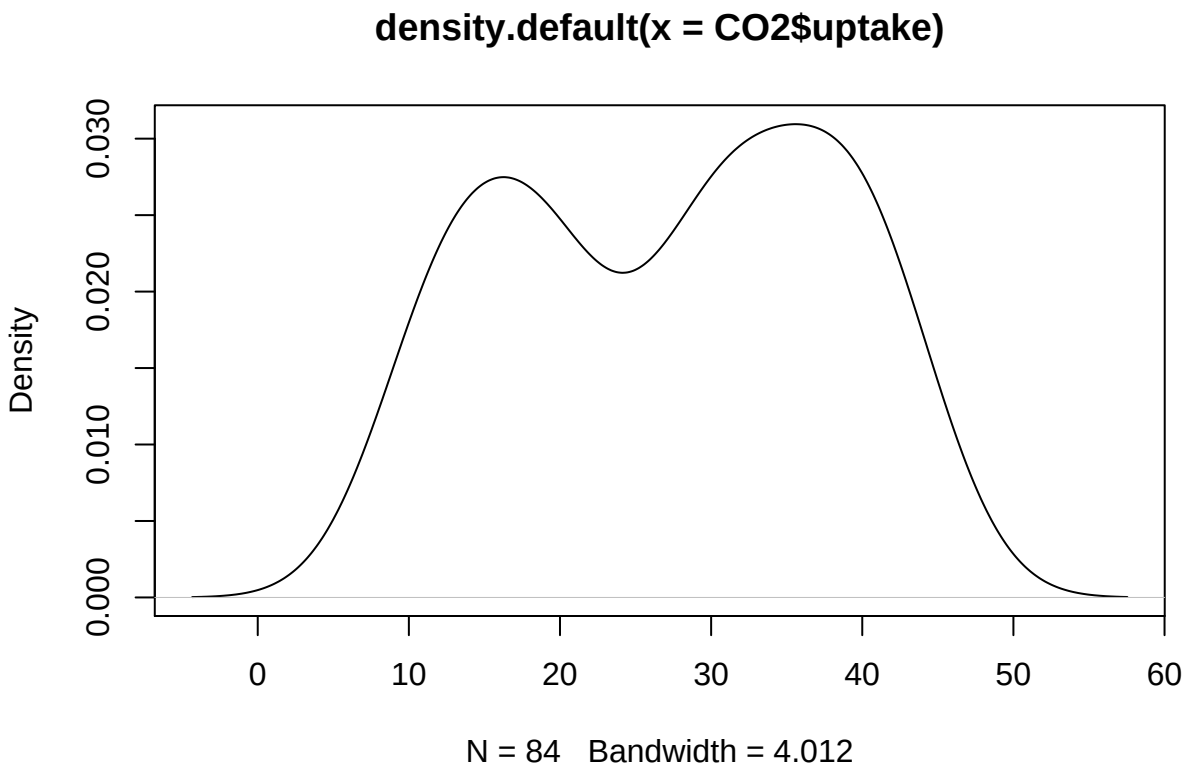
Example:

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

```
##
## Call:
##  density.default(x = CO2$uptake)
##
## Data: CO2$uptake (84 obs.);  Bandwidth 'bw' = 4.012
##
##        x                y
##  Min.   :-4.337   Min.   :2.286e-05
##  1st Qu.:11.132   1st Qu.:2.863e-03
##  Median :26.600   Median :2.125e-02
```

```
##  Mean   :26.600   Mean   :1.615e-02
##  3rd Qu.:42.068   3rd Qu.:2.649e-02
##  Max.   :57.537   Max.   :3.095e-02
```

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

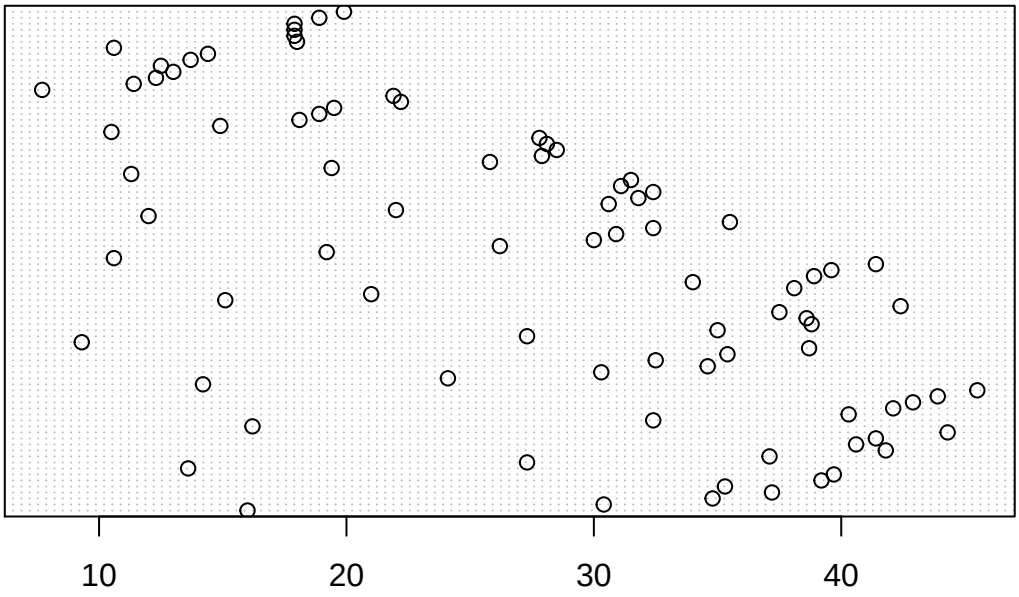


Dotcharts - Displays all point values along one dimension. The X-axis corresponds to the value you are plotting, while the Y-axis separates out the different data points

```
?dotchart
```

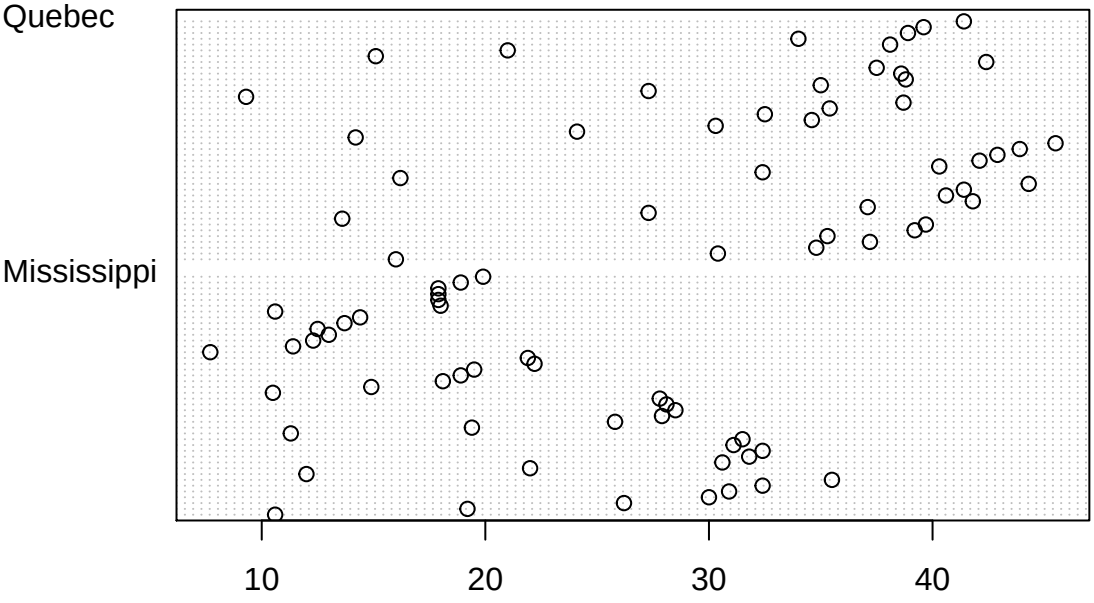
Examples:

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

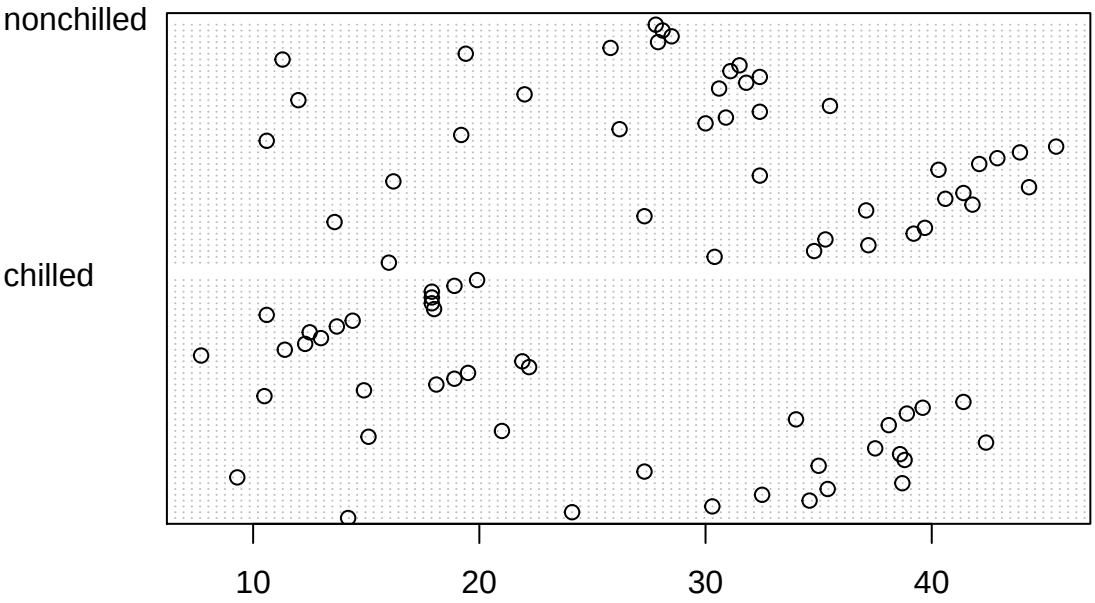


Note, you can also specify a grouping factor to divide the points and visually compare by group

```
dotchart(CO2$uptake, groups = CO2$Type) # separate out data by plant origin
```

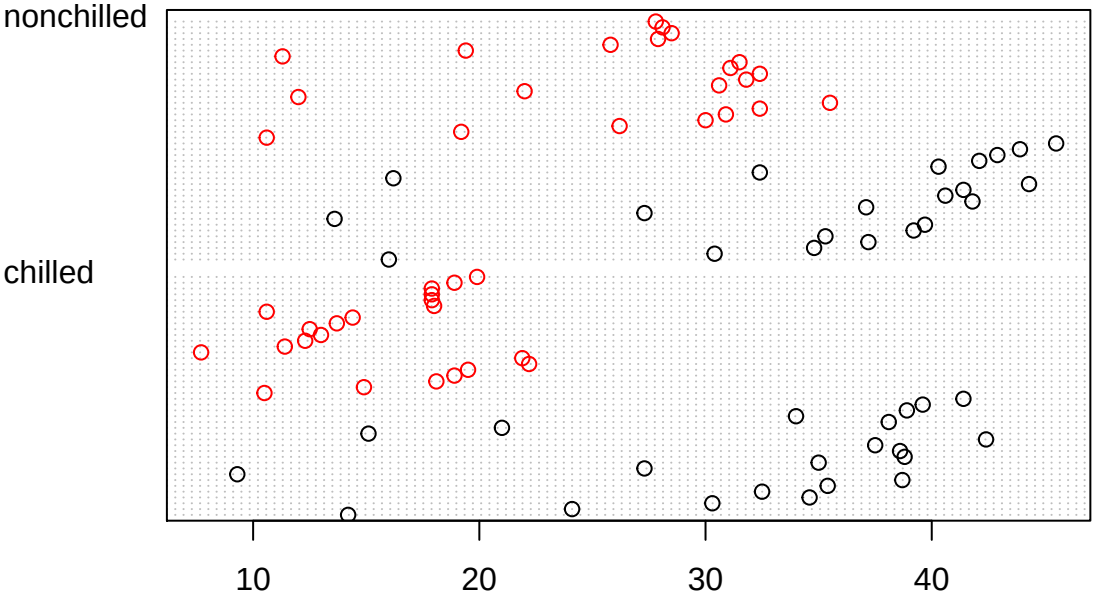



```
dotchart(CO2$uptake, groups = CO2$Treatment) # And do the same for the different treatments (chilled vs
```



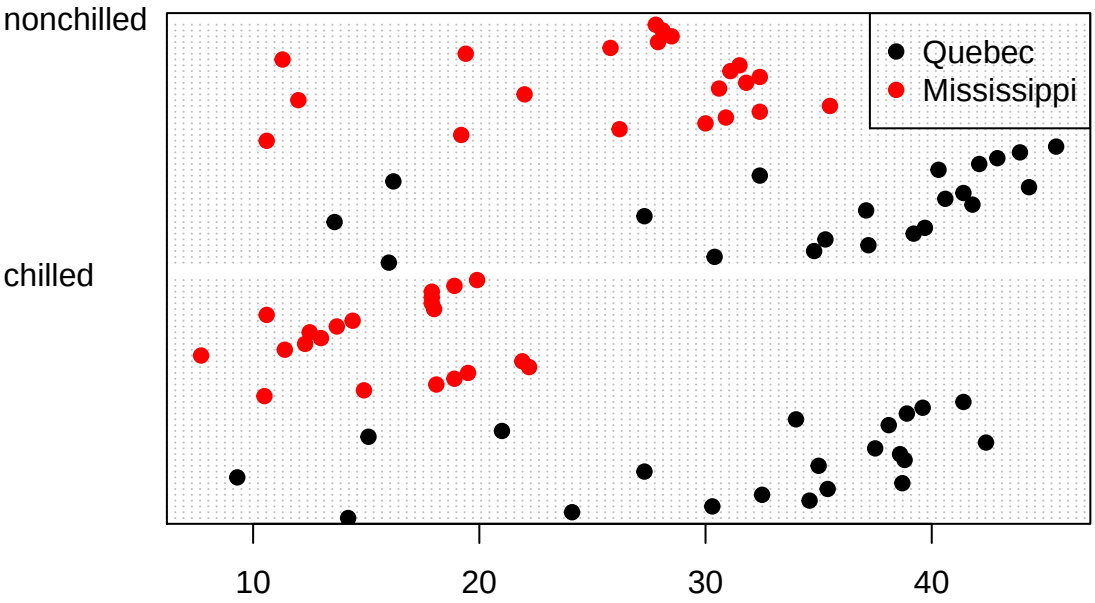
You might also be interested in what the data looks like for different plant origins when divided up by treatment. You can use the "col" argument to color by a factor level:

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



But, which color corresponds to Quebec, and which to Mississippi? You can add a legend using the same factor levels to make this clear:

```
dotchart(CO2$uptake,
         groups = CO2$Treatment,
         col = CO2$Type,
         pch = 19) # Set the marker type so it is the same in the plot and in the legend.

legend(x = "topright",
       legend = levels(CO2$Type), # Using levels() here pulls out the unique factor levels for your leg
       col = 1:length(CO2$Type), # For coloring the marker, you want a series of numbers that are the s
       pch = 19)
```



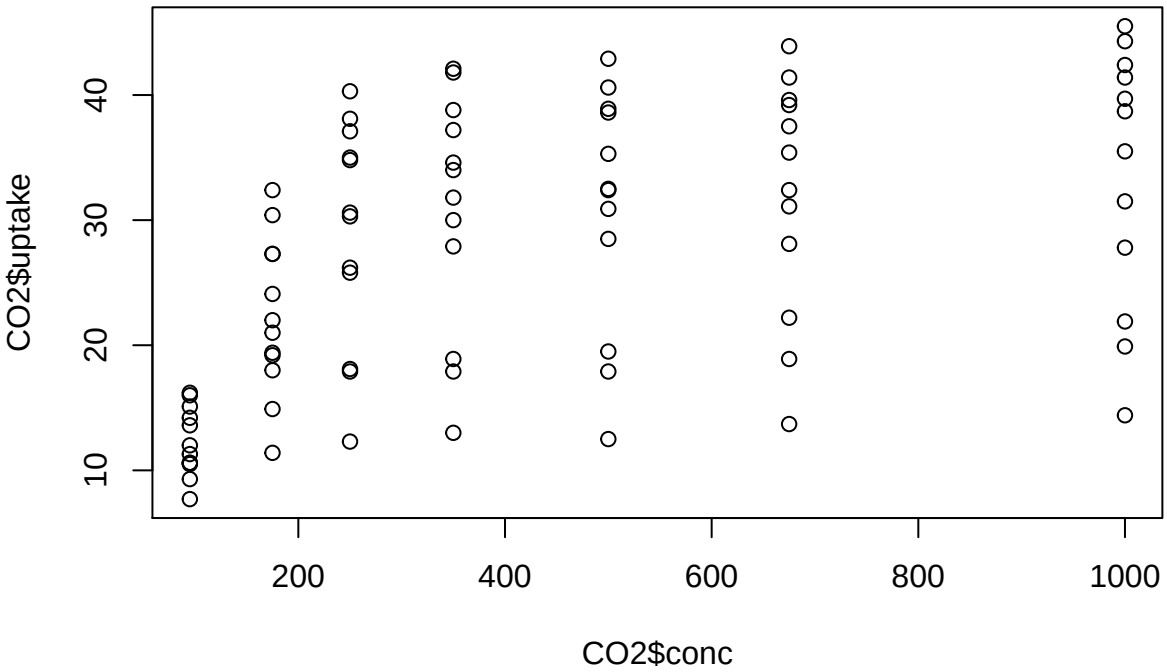
Question: so far in our visual data exploration, what have you noticed about CO2 uptake by plant country origin? What about by treatment?

Plotting two variables

Scatterplots: 2 continuous variables

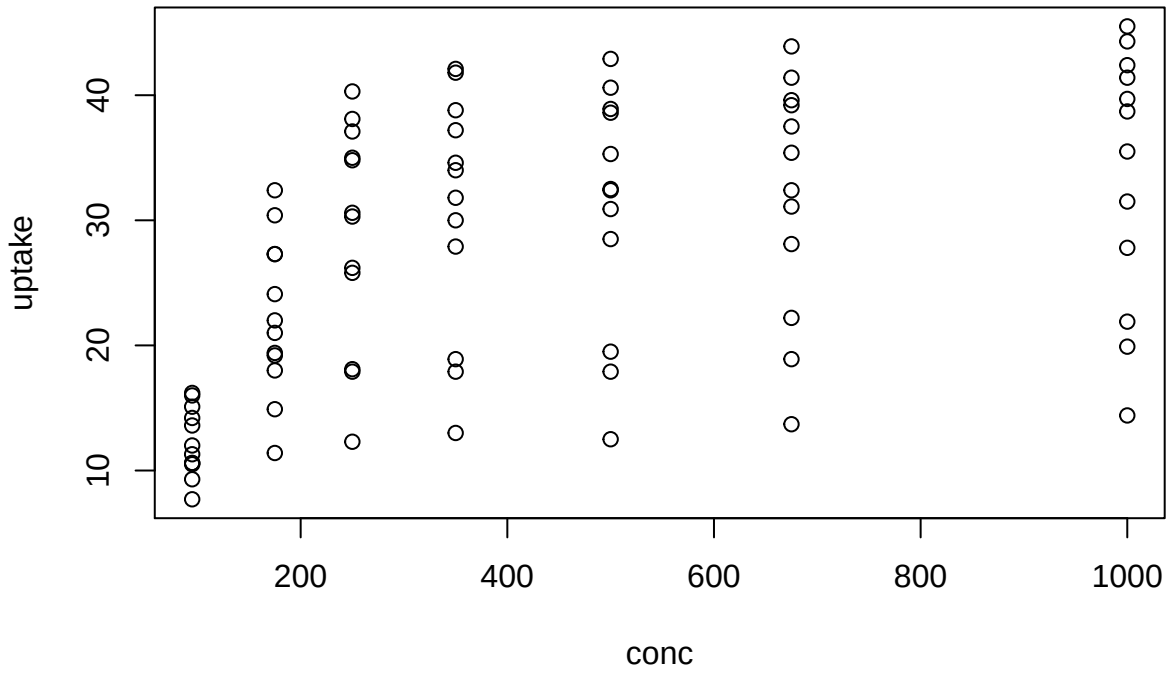
We can use a scatterplot to look at how uptake rate changes with ambient CO2 concentration:

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




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

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

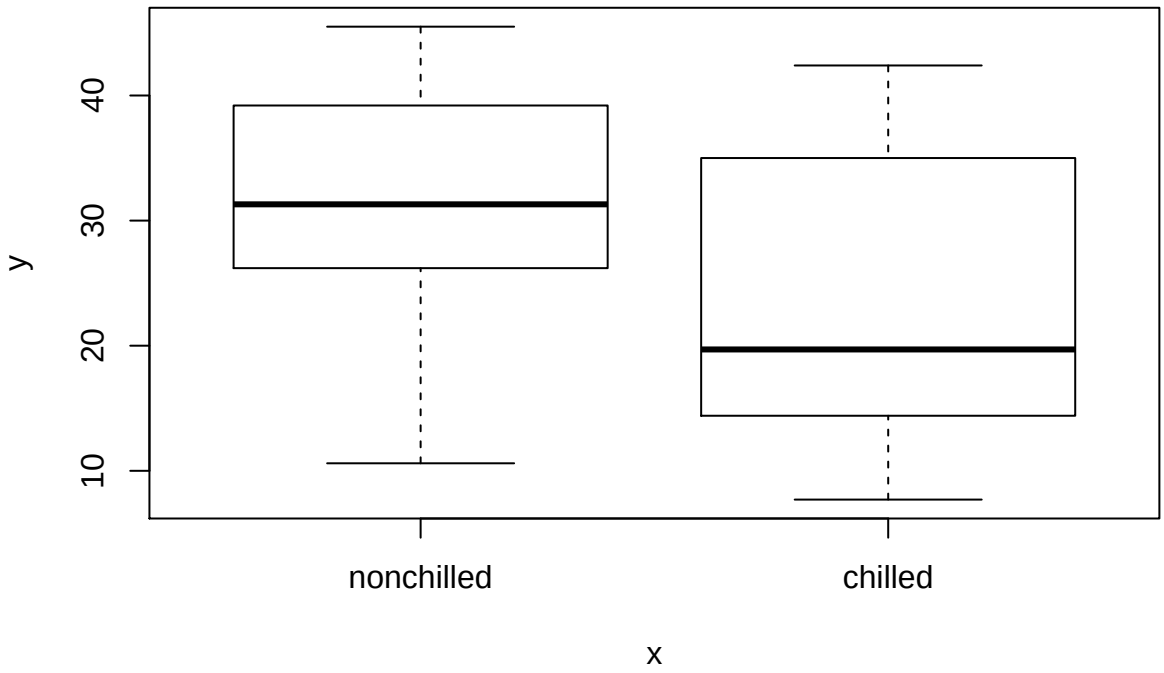


Tip: the tilde signifies that the left side variable is a response (i.e., the "y" variable) to the right side variable (i.e., the "x" variable). The same notation is used in statistical modeling, which you will see tomorrow.

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

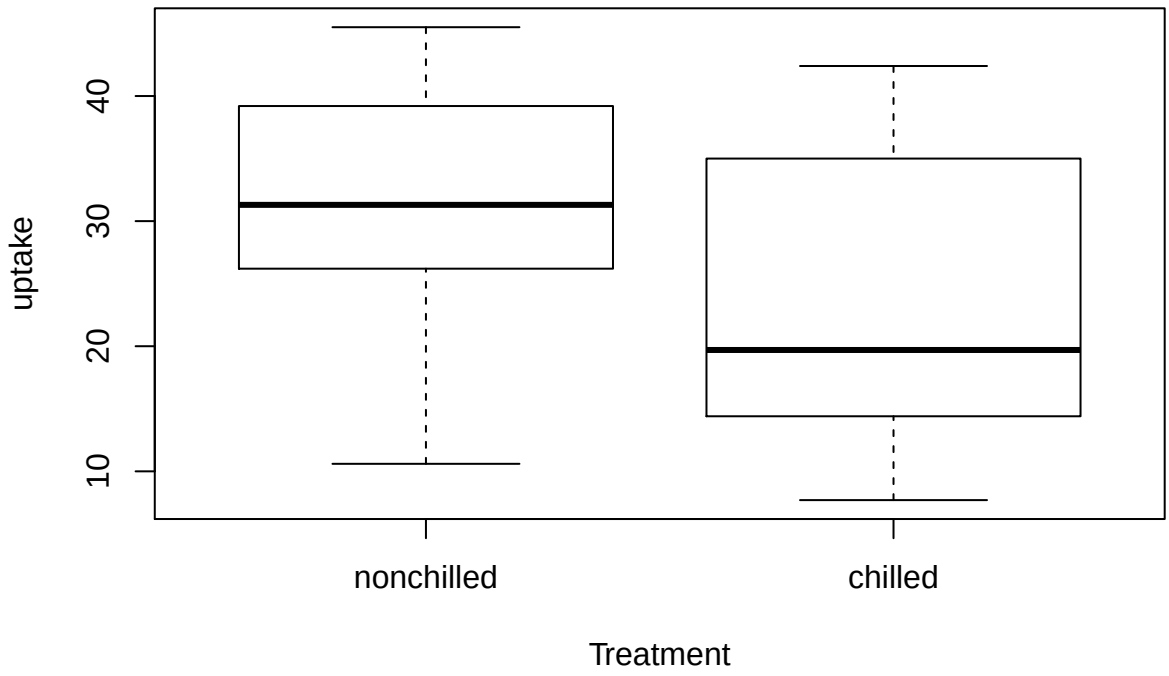
You can do this 2 ways: specify a categorical (factor level) X variable in plot(), which will automatically produce a boxplot, or use the boxplot() function directly.

```
plot(x = CO2$Treatment,
     y = CO2$uptake)
```



Before using the boxplot function, let's take a quick look at its help file to understand the different arguments:

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



Characteristics of good data plots:

- A purpose, clearly defined set of variables, and a message to convey.
- Succinct and uncluttered so as to allow easy interpretation of the purpose.

- Comprehensive, representing the data accurately and fairly.

Let's now take a closer look at the different plot types using our CO2 dataset to ask biological questions and explore relationships.

Scatterplots

changing ambient CO2 should affect CO2 uptake - we predict that less CO2 available might result in a slower uptake rate. Can we see this in the data?

We are interested in two numeric variables, so let's look at a scatterplot of our data:

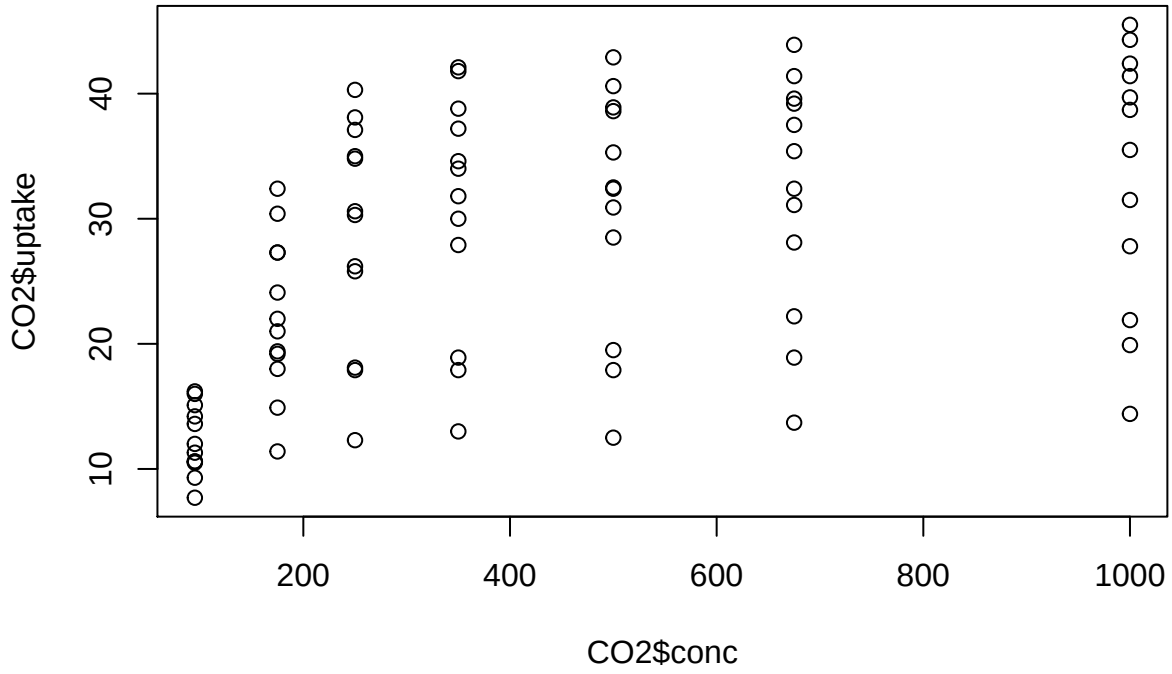
```
class(CO2$conc)
```

```
## [1] "numeric"
```

```
class(CO2$uptake)
```

```
## [1] "numeric"
```

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



The ambient concentration variable was a fixed variable, so we have several observations within single values, which is why the plot looks discontinuous. We might also be interested in the averages of CO2 uptake within each ambient CO2 concentration:

First let's get the means:

```
(uptakeMeans <- aggregate(uptake ~ conc, data = CO2, FUN = mean))
```

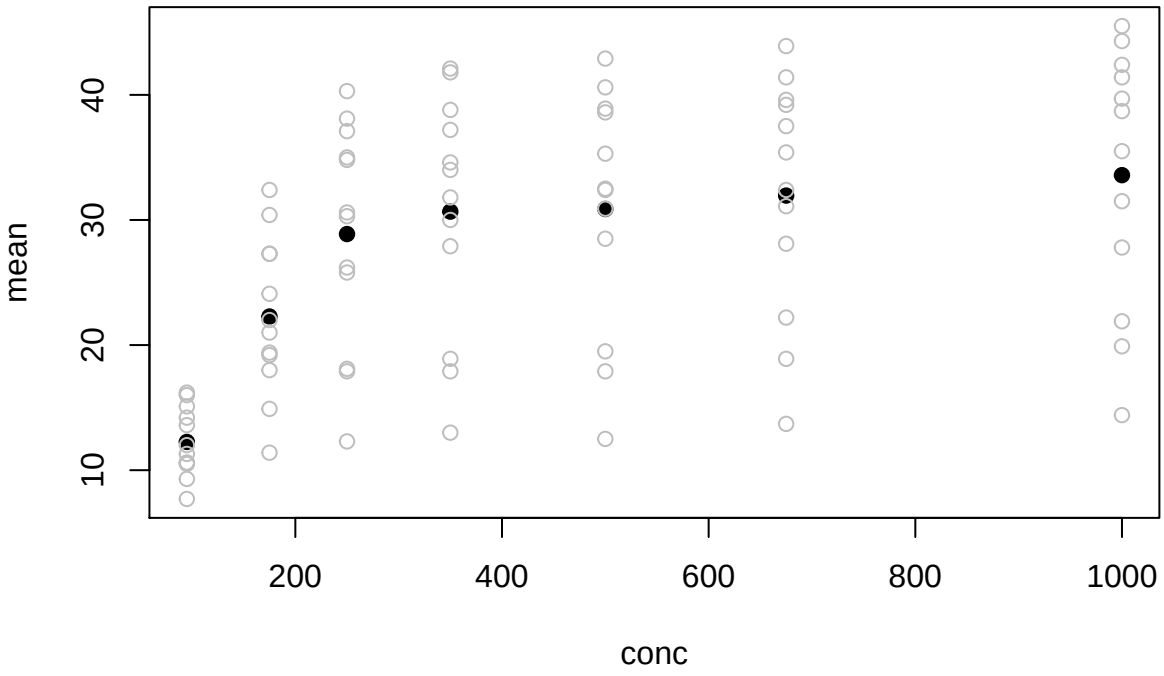
```
##   conc   uptake
## 1   95 12.25833
## 2  175 22.28333
## 3  250 28.87500
## 4  350 30.66667
```

```
## 5  500 30.87500
## 6  675 31.95000
## 7 1000 33.58333
```

```r
names(uptakeMeans) <- c("conc", "mean")
```

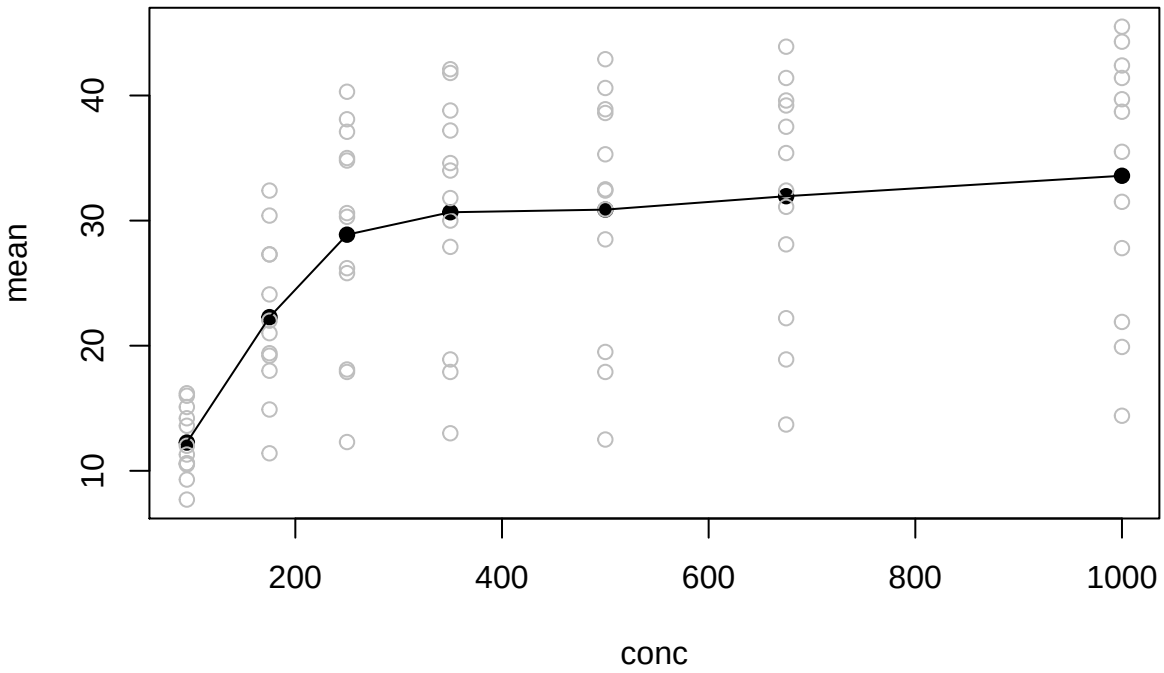
We could plot the means, then add the original data points to see the total sample spread of data around the mean. (Honestly, this is just to illustrate the ability to add things to a plot with functions such as points and lines)

```r
plot(mean ~ conc,
     data = uptakeMeans,
     pch = 19, # set the pch to a solid dot to differentiation the mean from the rest of the data
     ylim = c(min(CO2$uptake), max(CO2$uptake))) # Change the y limits to accomodate the total data spr

points(CO2$uptake ~ CO2$conc, col = "grey")
```



For fun, lets add a line across the means too.

```r
plot(mean ~ conc,
     data = uptakeMeans,
     pch = 19, # set the pch to a solid dot to differentiation the mean from the rest of the data
     ylim = c(min(CO2$uptake), max(CO2$uptake))) # Change the y limits again to accomodate the total da

points(CO2$uptake ~ CO2$conc, col = "grey")

lines(mean ~ conc,
      data = uptakeMeans)
```

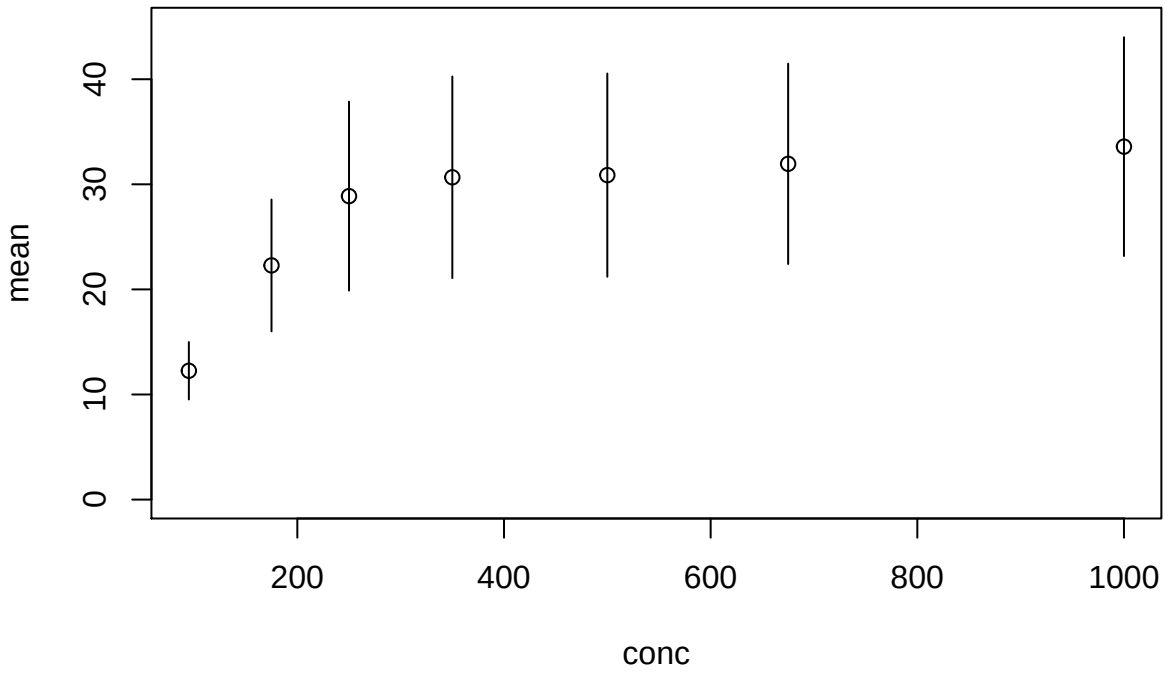


But instead of plotting all the data with the means, it's usually a good idea to plot some kind of error bar around each mean to succinctly show variation around the average. Different error bars serve different purposes. Since we are just interested in looking at the spread of our uptake data at each ambient CO2 concentration, we will add standard deviation bars.

```
# First, let's get the sds, aggregated by concentration.
(uptakeSD <- aggregate(uptake ~ conc, data = CO2, FUN = sd))
```

```
##   conc    uptake
## 1   95  2.735111
## 2  175  6.271992
## 3  250  8.988895
## 4  350  9.596527
## 5  500  9.671056
## 6  675  9.534960
## 7 1000 10.411867
```

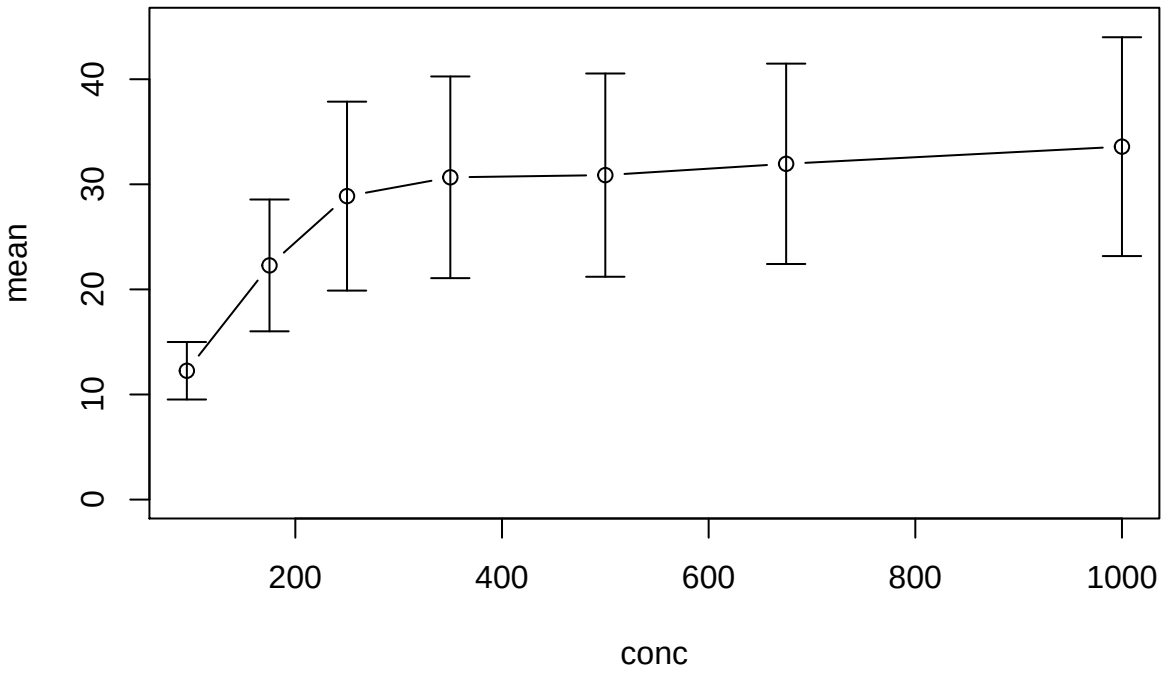
```
names(uptakeSD) <- c("conc", "sd")

plot(mean ~ conc,
     data = uptakeMeans,
     ylim = c(0, 45)) # we again expand the default Y-axis limits to accomodate the sd bars. I chose th

# like lines() and points(), arrows() adds arrows to an already existing plot. It literally plots arrow

arrows(x0 = uptakeMeans$conc,
       y0 = uptakeMeans$mean - uptakeSD$sd,
       y1 = uptakeMeans$mean + uptakeSD$sd,
       length = 0)
```




```
# x0, x1 and y0, y1 are where you draw the start (0) and end (1) of the bars. x1 and y1 default to x0 a
```

You can also plot this directly as a line plot by setting the "type" argument to "l" for lines or "b" for both points and lines. It defaults to "p" for points.

```
plot(mean ~ conc,
     data = uptakeMeans,
     ylim = c(0, 45),
     type = "b")
arrows(x0 = uptakeMeans$conc,
       y0 = uptakeMeans$mean - uptakeSD$sd,
       y1 = uptakeMeans$mean + uptakeSD$sd,
       angle = 90,
       code = 3,
       length = 0.1)
```

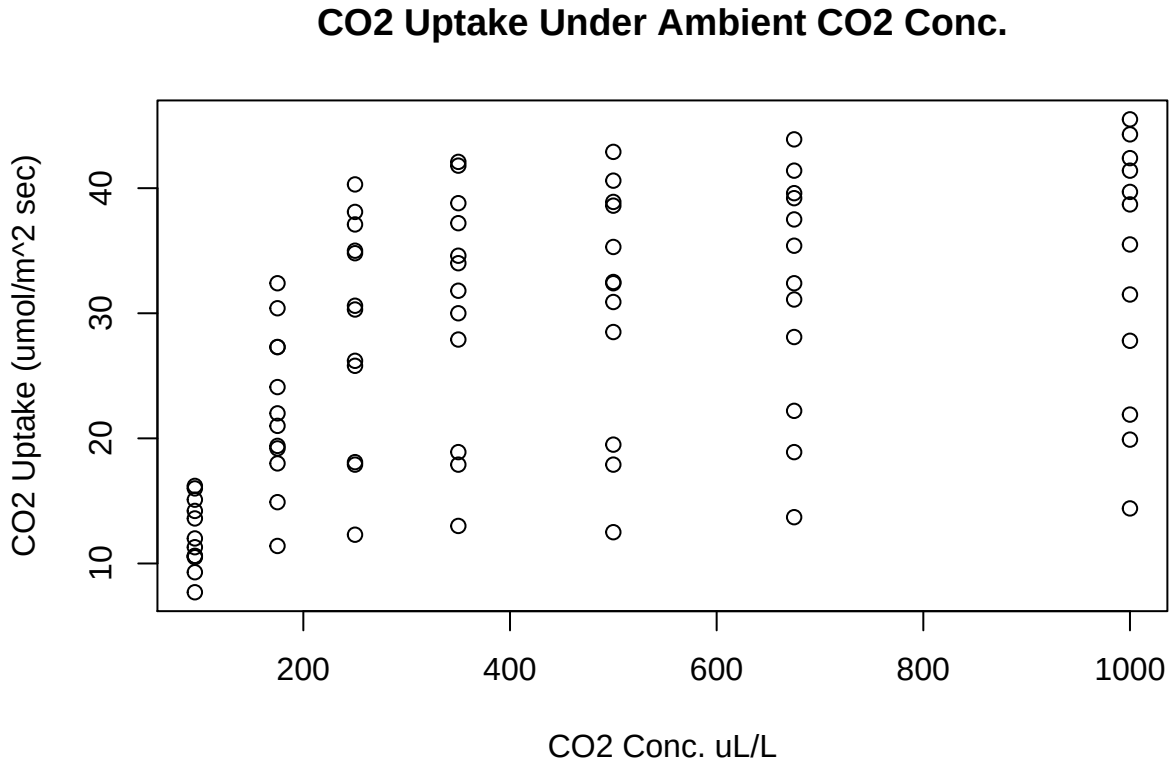


Commentary: There's an increase in uptake as the concentration gradient increases, but it doesn't really increase much after ~250 uL/L. It may imply that CO2 concentration is only a limiting factor at very low levels and does not drive uptake at higher concentrations.

Let's look at some ways we can make our plots look a little better and be more easily interpreted.

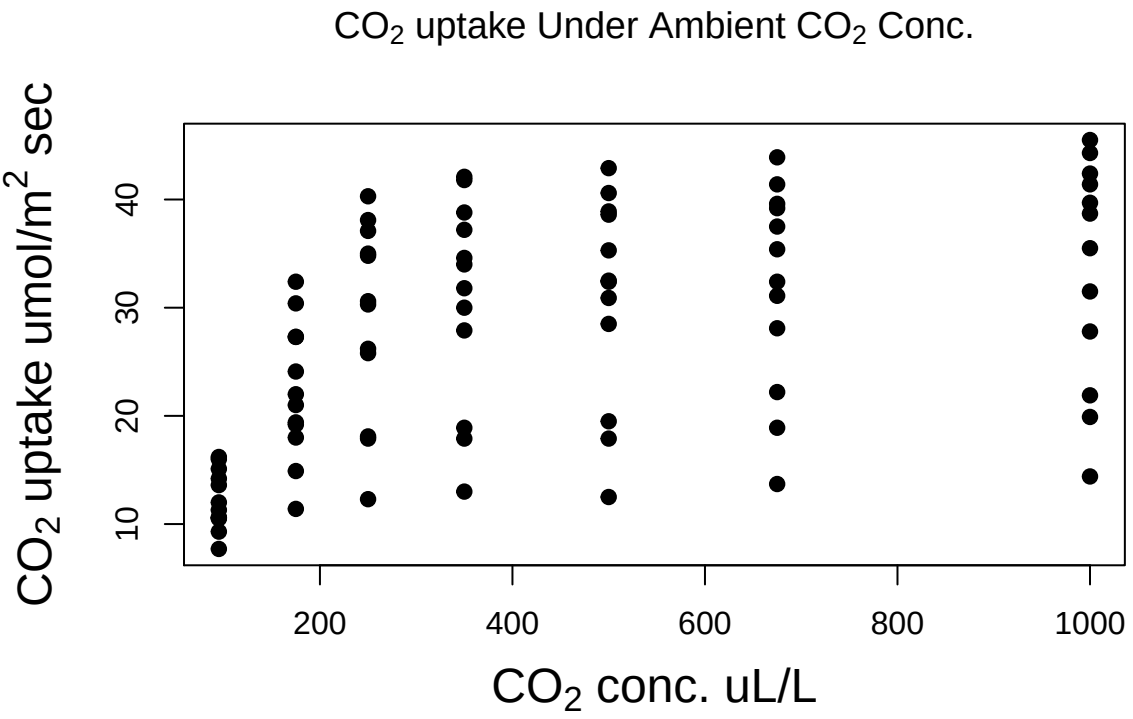
First, we can add some labels and a title:

```
plot(x = CO2$conc, y = CO2$uptake,
     xlab = 'CO2 Conc. uL/L',
     ylab = 'CO2 Uptake (umol/m^2 sec)',
     main = 'CO2 Uptake Under Ambient CO2 Conc.')
```



We can also improve the text using expression() to print special characters,increase the font size of our axis labels (cex.lab), and change the type of marker for our points (pch). We'll use par to increase font size and set marker type, as well as increase the margins around our plot window to accomodate the larger axis labels (mar).

```
?par
par(mar = c(5, 4, 4, 2) + 1,
    cex.lab = 1.5,
    pch = 19)
plot(x = CO2$conc, y = CO2$uptake,
     xlab = expression(paste(CO[2] ," conc. uL/L")),
     ylab = expression(paste(CO[2], " uptake umol/", m^2 ," sec")),
     main = expression(paste(CO[2], " uptake Under Ambient ", CO[2], " Conc.")))
```



Note, par() is useful if you want to set plot specifications for the plotting window, as well as any plots following, but many of the arguments could be used directly in the plot function too for a specific plot.

Using expression() with paste() allows us to print special characters in our plot text. You can do a google search to find examples online for the type of special character you need to print. Above, you can see that m^2 prints 2 as a superscript, while [2] prints 2 as a subscript. You don't actually need quotations around any of the text in expression(), but they allow you to add spaces where necessary.

Exploring data with biological questions in mind.

Biological processes are usually temperature mediated. We should expect higher respiration (CO2 uptake) with higher temperatures. Using levels() shows us the unique factors from a 'factor' vector.

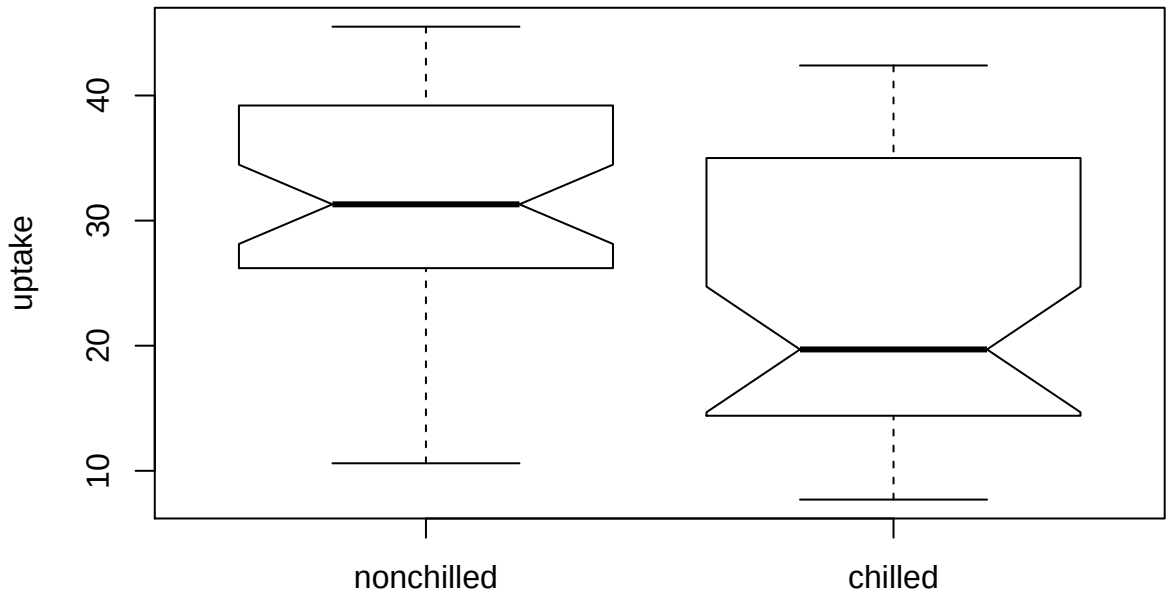
```
levels(CO2$Treatment)
```

```
## [1] "nonchilled" "chilled"
```

We're interested in the distribution of CO2 uptake as it relates to treatment. Different ways we can visualize this: + Box plot - grouping along a discrete variable (treatment), display distribution of data in each group + Density plot with line type determined by factor + Scatterplot colored by factor level

Box plot:

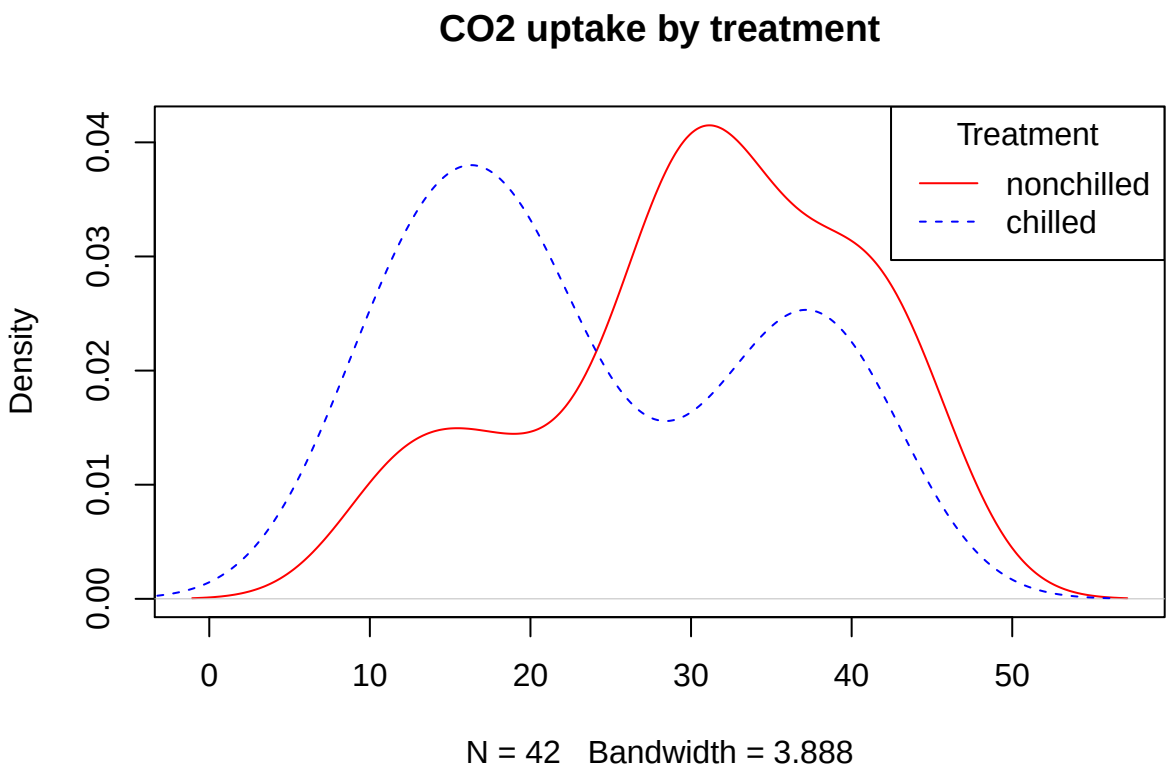
```
boxplot(uptake ~ Treatment, data = CO2, notch = TRUE) # including notch=TRUE provides evidence that the
```



Our boxplot indicates there is some effect of treatment. As we predicted, CO2 uptake is lower in the chilled conditions. But we have yet to do a statistical test!

Density plot

```
plot(density(CO2[CO2$Treatment == "nonchilled", "uptake"]),
     col = "red",
     type = "l",
     main = "CO2 uptake by treatment",
     lty = 1)

lines(density(CO2[CO2$Treatment == "chilled", "uptake"]),
      col = "blue",
      lty = 2)

legend(x = "topright",
       legend = c("nonchilled", "chilled"),
       col = c("red", "blue"),
       lty = c(1,2),
       title = "Treatment")
```

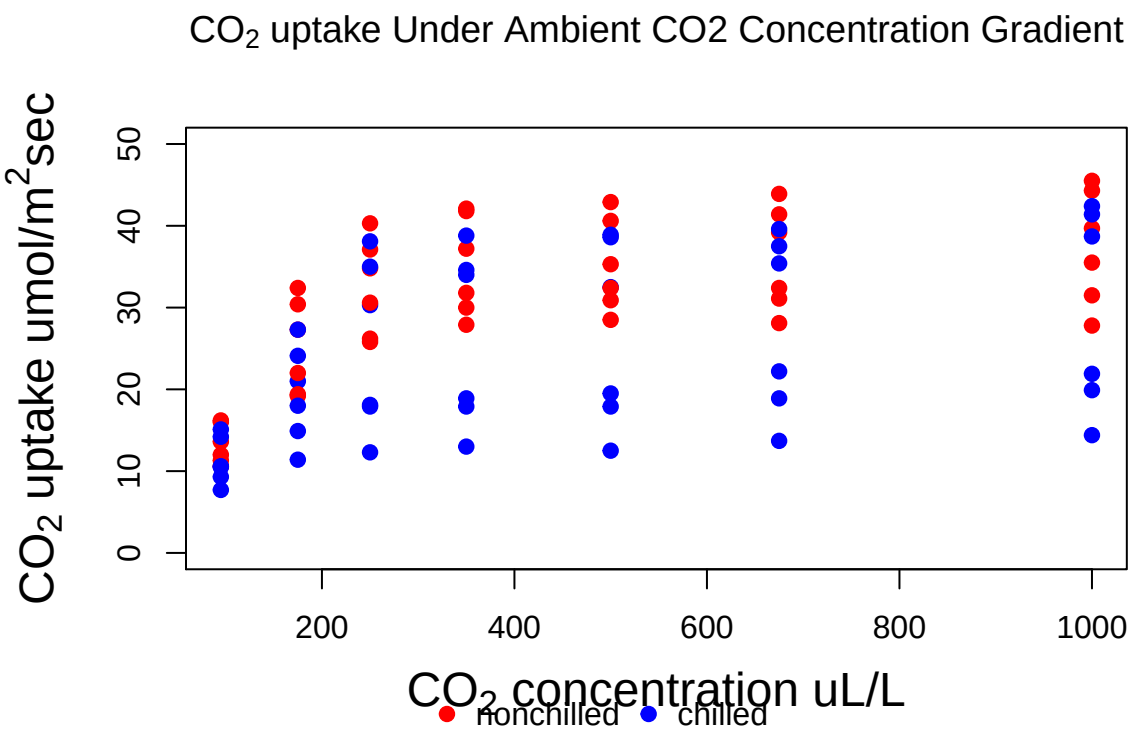


An interesting thing to note is there is a second density peak for chilled at higher CO2 uptake levels. What do you think is going on there?

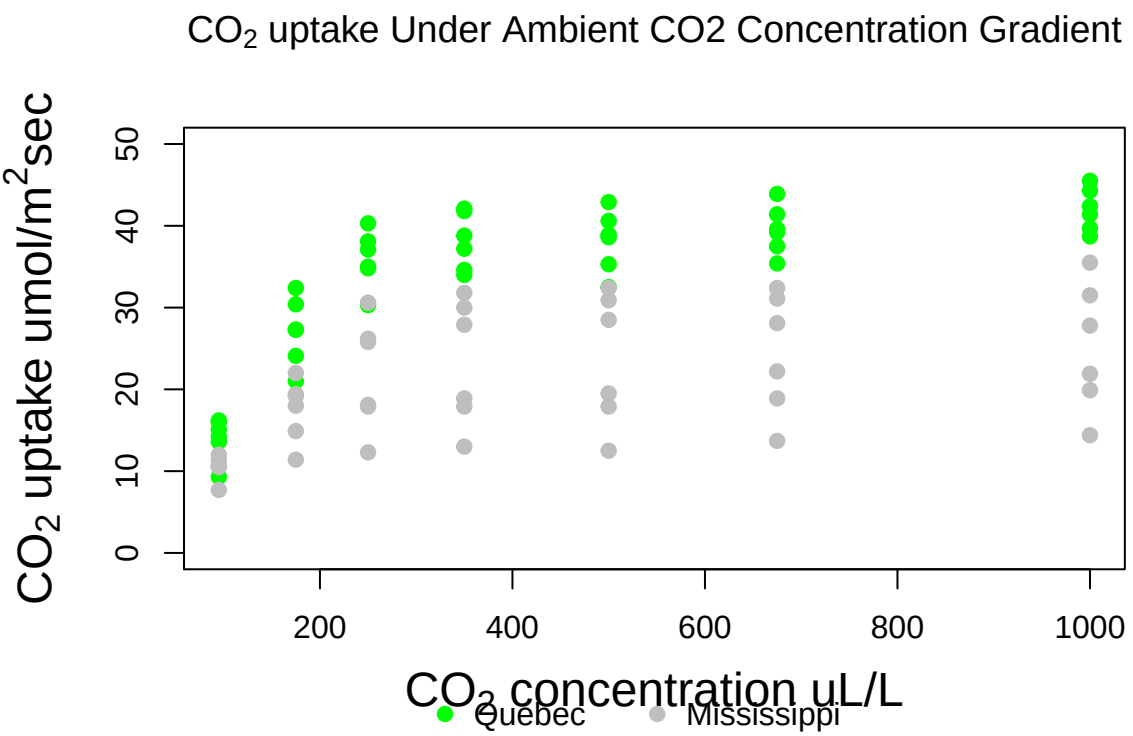
Scatterplot with color by factor level

Below, we will use the col argument to change the colors of the points that we plot by different factor levels, first by treatment, then by plant origin. We will also plot CO2 uptake against ambient CO2 once more.

```
par(mar = c(5, 4, 4, 2) + 1,
    cex.lab = 1.5,
    pch = 19,
    xpd = TRUE) # xpd set to TRUE allows us to plot the legend following outside of the plot box

palette(c("red", "blue")) # R has a default palette, which is the colors that automatically get pulled,

plot(x = CO2$conc, y = CO2$uptake,
     ylim = c(0, 50),
     xlab = expression(paste(CO[2] ," concentration uL/L")),
     ylab = expression(paste(CO[2], " uptake umol/", m^2 ,"sec")),
     main = expression(paste(CO[2], " uptake Under Ambient CO2 Concentration Gradient")),
     col = CO2$Treatment)
legend(x = 300,
       y = -15,
       legend = levels(CO2$Treatment),
       col = 1:2,
       pch = 19,
       bty = "n",
       horiz = T)
```

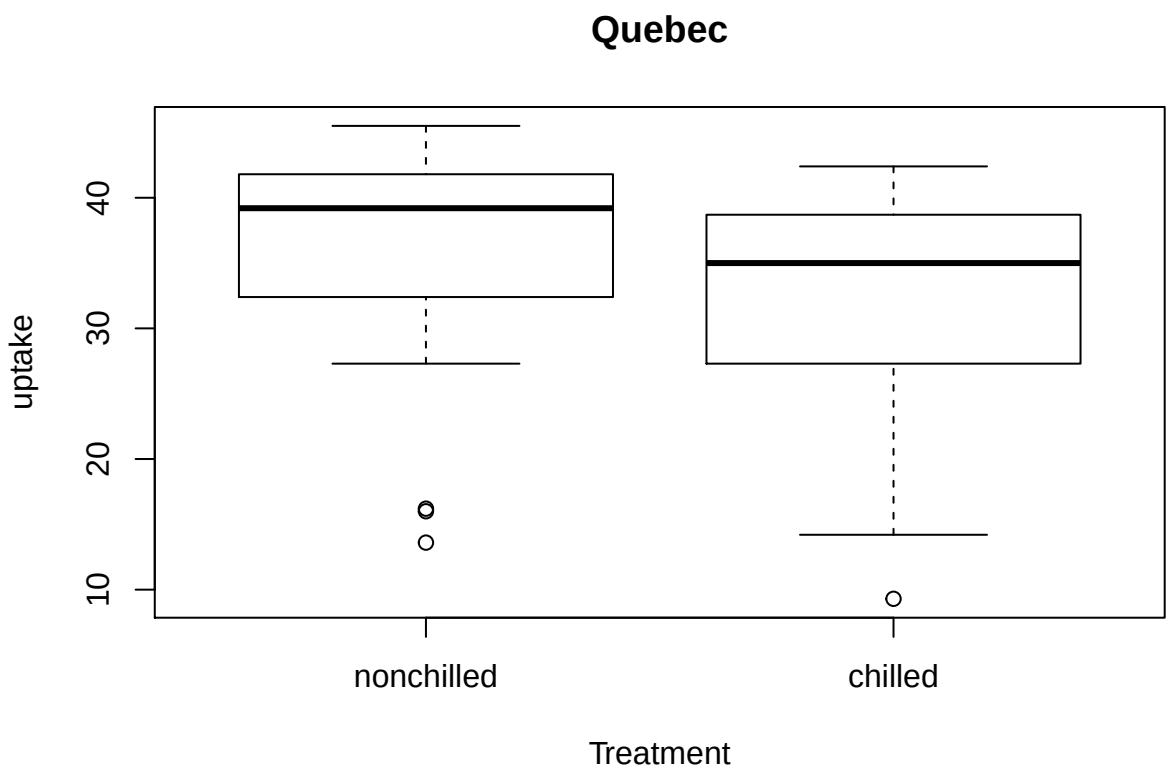



```
palette(c("green", "grey"))
plot(x = CO2$conc, y = CO2$uptake,
     ylim = c(0, 50),
     xlab = expression(paste(CO[2] ," concentration uL/L")),
     ylab = expression(paste(CO[2], " uptake umol/", m^2 ,"sec")),
     main = expression(paste(CO[2], " uptake Under Ambient CO2 Concentration Gradient")),
     col = CO2$Type)
legend(x = 300,
       y = -15,
       legend = levels(CO2$Type),
       col = 1:2,
       pch = 19,
       bty = "n",
       horiz = T)
```

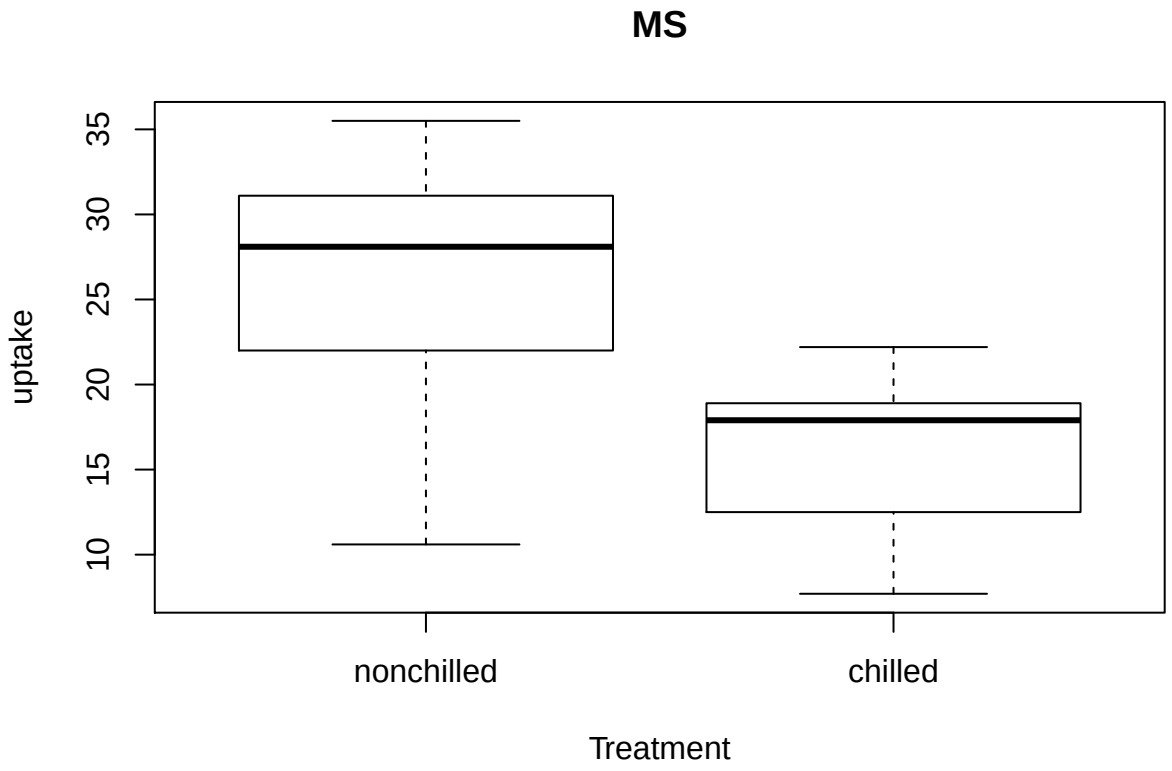


Now that we've looked at the uptake by both treatment and plant origin, let's take a closer look at the effect of plant origin on uptake and see if that might interact with treatment, as the plots above have hinted at.

```
boxplot(uptake ~ Treatment, data = CO2[CO2$Type == "Quebec",], main = "Quebec")
```



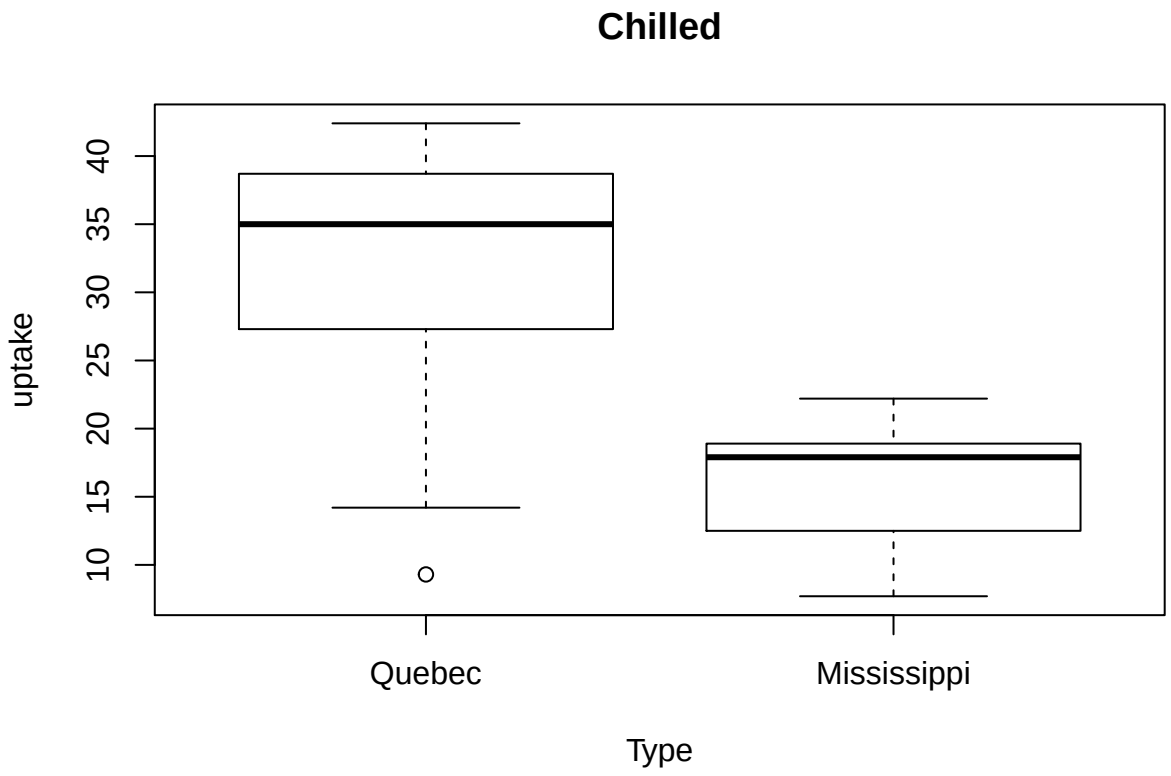
```
boxplot(uptake ~ Treatment, data = CO2[CO2$Type == "Mississippi",], main = "MS")
```



You can see above that the effect of the temperature treatment is much stronger, and perhaps only significant, in the plants that originated from Mississipi, as we might expect.

Let's also compare the distribution of uptake rates between Quebec and Mississippi under both temperature conditions.

```
boxplot(uptake ~ Type, data = CO2[CO2$Treatment == "chilled",], main = "Chilled")
```




```
boxplot(uptake ~ Type, data = CO2[CO2$Treatment == "nonchilled",], main = "Not chilled")
```

In fact, we can see that Quebec appears to have higher uptake rates in both chilled and non-chilled conditions, though the difference is stronger under cold conditions.

Exercises

1. Save the dataset "iris" to an object in your work environment.
2. Read through the help file on iris to understand what the data is about.
3. Use head and str to look at the data. How many variables are there? What kind of data type are they?

Univariate plots

Tip: before adding color to any plot to show species differences, make sure to change your current color palette, as it currently only contains two colors if you were following along above.

4. Let's just take a broad look first at a single variable. Create dot charts for petal length, first for all the data, then by species.
    - Do you see a difference in petal length by species?
5. The dots appear pretty clumped for one species, and more spread out for the other species. Let's look closer at the distribution of petal lengths in each species. How might you visualize this? There are a few ways. Go ahead and create an appropriate plot, complete with title and axis labels.
6. How might you visualize the data to see if there is a relationship between sepal length and petal length? Go ahead and create a plot that shows how petal length (we'll call it our "Y" variable) varies with sepal length (our "X" variable). Make sure to include a title and axis labels.
    - Do you notice a relationship between petal and sepal length?
7. Building on the plot from 6, change the color of the points to vary with species.
8. Add a legend to this plot, with appropriate text, marker type, and color corresponding to the factor levels of species.
    - Does there appear to be a difference in flower size between the three species? Which species would have the longest petals and sepals on average?
9. Create a plot of the average sepal lengths by species, with standard deviation error bars. Make sure to include a title and axis labels.