NC STATE UNIVERSITY

Summarizing Data: Graphical Displays via Base R

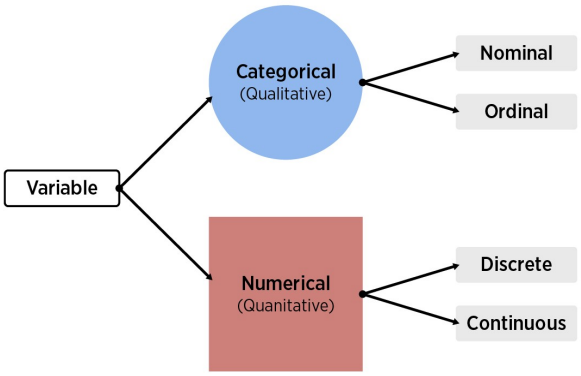
Justin Post

Where are we at?

- Understand types of data and their distributions
- Numerical summaries (across subgroups)
    - Contingency Tables
    - Mean/Median
    - Standard Deviation/Variance/IQR
    - Quantiles/Percentiles
- Graphical summaries (across subgroups)
    - Bar plots (categorical data)
    - Histograms
    - Box plots
    - Scatter plots

Understanding Data

- How to summarize data?
- Depends on data type:
  - Categorical (Qualitative) - variable whose entries are a label or attribute
  - Numeric (Quantitative) - variable whose entries are a numerical value where math can be performed



Understanding Data

Common goal: Describe the **distribution** of the variable

- Distribution = pattern and frequency with which you observe a variable
- Categorical variable - describe relative frequency (or count) in each category
- Numeric variable - describe the shape, center, and spread

Graphical Summaries

Three major systems for plotting:

- Base R (built-in functions)
    - Use `plot`, `barplot`, `hist`, etc. to start a plot
    - Annotate the plot using functions like `text`, `lines`, `points`, etc.

Graphical Summaries

Three major systems for plotting:

- Base R (built-in functions)
  - Use `plot`, `barplot`, `hist`, etc. to start a plot
  - Annotate the plot using functions like `text`, `lines`, `points`, etc.
- Lattice (not covered)
- ggplot2 (sort of part of the tidyverse - Cheat Sheet)
  - `ggplot(data = data_frame)` creates a plot instance
  - Add “layers” to the system (geoms or stats)

Great reference book here!

Base R Plotting: Categorical variables

Categorical variable - entries are a label or attribute

- Barplots via `barplot`

Base R Plotting: barplot

- Consider data on titanic passengers in `titanic.csv`

```
## # A tibble: 1,310 x 14
##   pclass survived name  sex      age sibsp parch ticket  fare cabin embarked
##    <dbl>    <dbl> <chr> <chr>  <dbl> <dbl> <dbl> <chr>  <dbl> <chr> <chr>
## 1      1        1 Alle~ fema~ 29         0     0 24160   211. B5    S
## 2      1        1 Alli~ male   0.917     1     2 113781  152. C22 ~ S
## 3      1        0 Alli~ fema~  2         1     2 113781  152. C22 ~ S
## 4      1        0 Alli~ male  30         1     2 113781  152. C22 ~ S
## 5      1        0 Alli~ fema~ 25         1     2 113781  152. C22 ~ S
## # ... with 1,305 more rows, and 3 more variables: boat <chr>, body <dbl>,
## #   home.dest <chr>
```

Base R Plotting: barplot

- Easiest to create a summary dataset with `table` for use with `barplot`
- One-way table easy to visualize

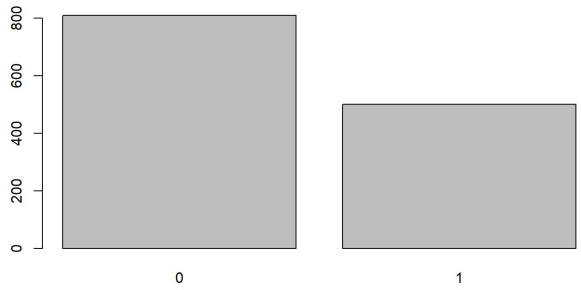
```
table(titanicData$survived)
```

```
##
##   0   1
## 809 500
```

Base R Plotting: barplot

- Easiest to create a summary dataset with `table` for use with `barplot`
- One-way table easy to visualize

```
barplot(table(titanicData$survived))
```

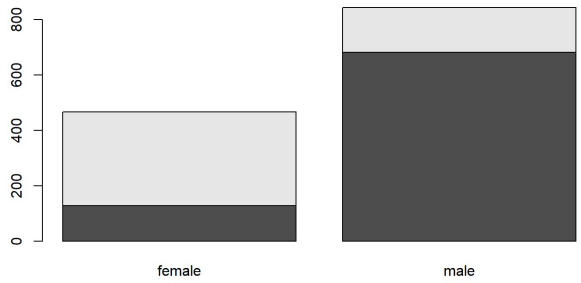


Base R Plotting: Stacked barplot

Categorical variable - entries are a label or attribute

- Easiest to create a summary dataset with `table` for use with `barplot`
- **Stacked barplot** created by passing a two-way table

```
twoTable <- table(titanicData$survived, titanicData$sex)
barplot(twoTable)
```

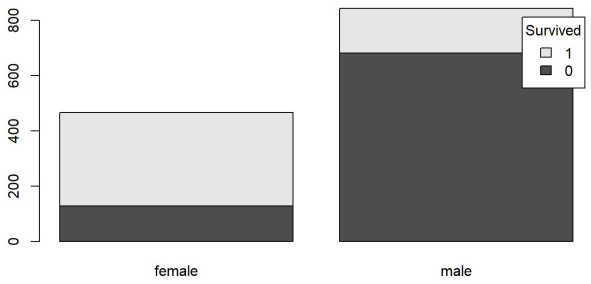


Base R Plotting: barplot legend

Categorical variable - entries are a label or attribute

- Common arguments to `barplot`
    - `legend = TRUE` & `args.legend = list(title = "...")`

```
twoTable <- table(titanicData$survived, titanicData$sex)
barplot(twoTable, legend = TRUE, args.legend = list(title="Survived"))
```

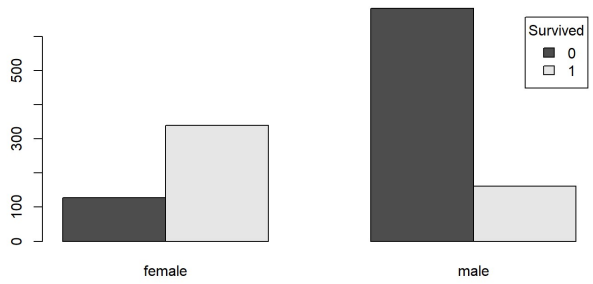


Base R Plotting: Side-by-side barplot

Categorical variable - entries are a label or attribute

- Easiest to create a summary dataset with `table` for use with `barplot`
- **Side-by-side barplot** created by passing a two-way table with `beside = TRUE`

```
twoTable <- table(titanicData$survived, titanicData$sex)
barplot(twoTable, beside = TRUE, legend = TRUE, args.legend=list(title="Survived"))
```

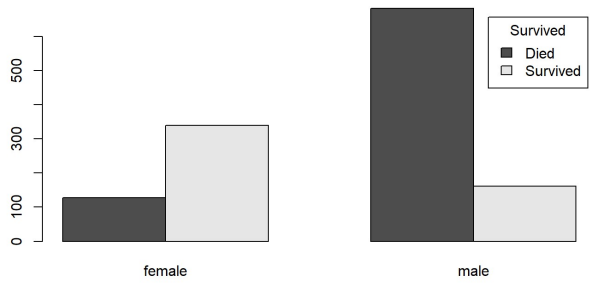


Base R Plotting:Side-by-side barplot

Categorical variable - entries are a label or attribute

- Easiest to create a summary dataset with `table` for use with `barplot`
- **Side-by-side barplot** created by passing a two-way table with `beside = TRUE`

```
barplot(twoTable, beside = TRUE, legend = c("Died", "Survived"),
        args.legend=list(title="Survived"))
```



Base R Plotting: Categorical variables

Recap!

Categorical variable - entries are a label or attribute

- Easiest to create a summary dataset with `table` for use with `barplot`
- Two-way table with `beside = TRUE` for side-by-side barplot
- Create same plot for each level of another variable?
    - Requires a bit of work with Base R… (`ggplot2` takes care of it for us!)

Next: Numeric variable plotting

Base R Plotting: Numeric Variables

Numeric variable - entries are a numerical value where math can be performed

Goal: describe the shape, center, and spread

- Generally, via a histogram or boxplot!

Process:

- Use `hist`, `boxpot`, `plot`, etc. to start a plot
- Annotate the plot using functions like `text`, `lines`, `points`, etc.

Base R Plotting: Numeric Variables

- Look at carbon dioxide (CO2) uptake data set
  - Response recorded: `uptake` CO2 uptake rates in grass plants
  - Environment manipulated: `Treatment` - chilled/nonchilled
  - Ambient CO2 specified and measured: `conc`

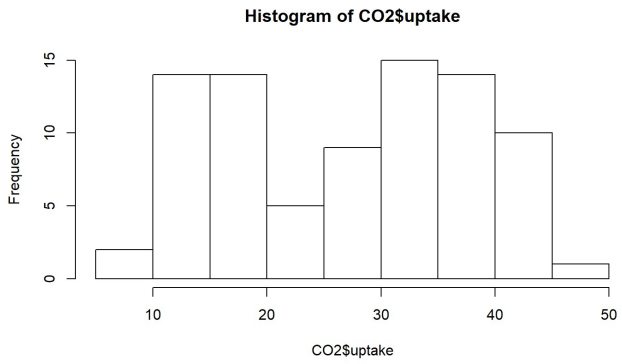
```
CO2 <- as_tibble(CO2)
CO2
```

```
## # A tibble: 84 x 5
##   Plant Type   Treatment   conc uptake
##   <ord> <fct>  <fct>      <dbl>  <dbl>
## 1 Qn1   Quebec nonchilled    95   16
## 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
## # ... with 79 more rows
```

Base R Plotting: Histogram

- **Histogram** - Bins data to show distribution of observations

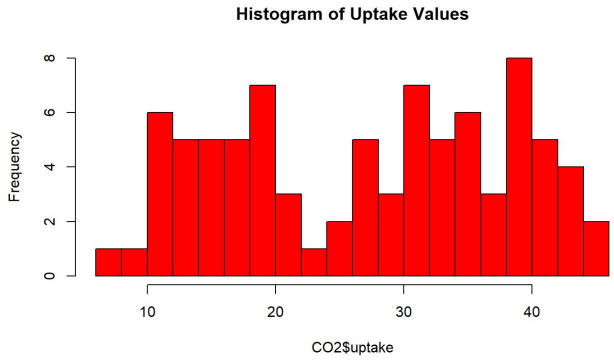
```
hist(CO2$uptake)
```



Base R Plotting: Histogram

- **Histogram** - Bins data to show distribution of observations
- Modify title with `main`, bins with `breaks`, color with `col`, ...

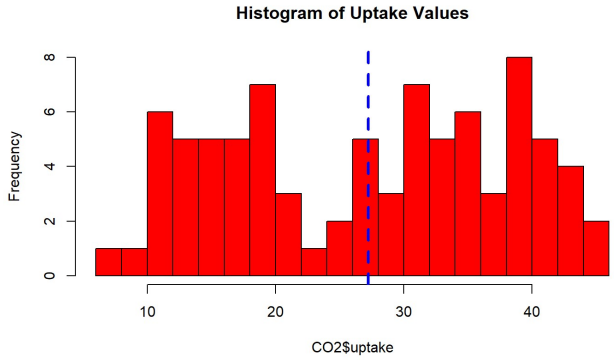
```
hist(CO2$uptake, main = "Histogram of Uptake Values", breaks = 15, col = "Red")
```



Base R Plotting: Adding a reference line

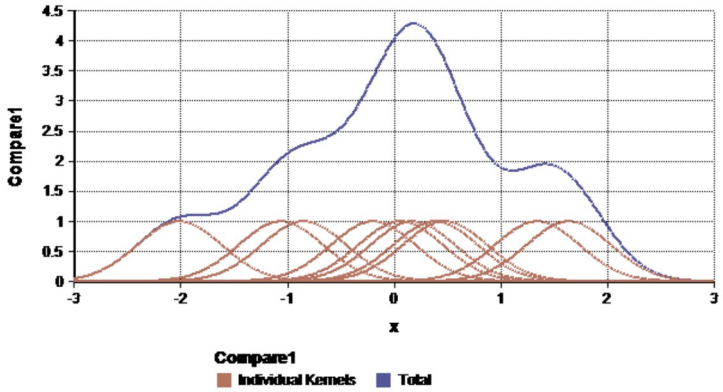
- **Histogram** - Bins data to show distribution of observations
- Add a line with `abline` (`lty` specifies line type, see `help(par)`)

```
hist(CO2$uptake, main = "Histogram of Uptake Values", breaks = 15, col = "Red")
abline(v = mean(CO2$uptake), lwd = 3, lty = 2, col = "Blue")
```



Base R Plotting: Kernel smoother

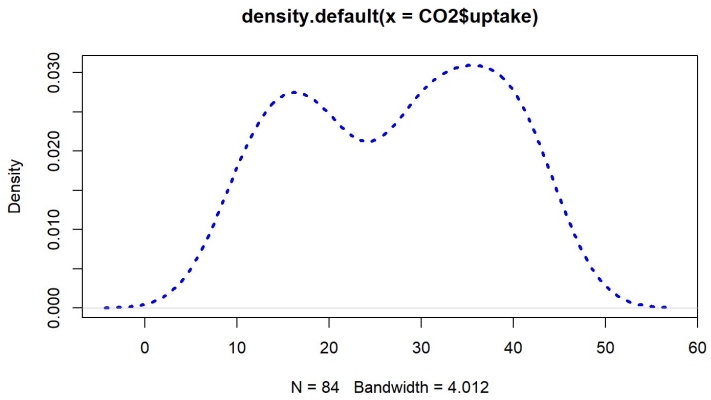
- **Kernel Smoother** - Smoothed version of a histogram
- 'Kernel' determines weight given to nearby points



Base R Plotting: Kernel smoother

- **Kernel Smoother** - Smoothed version of a histogram

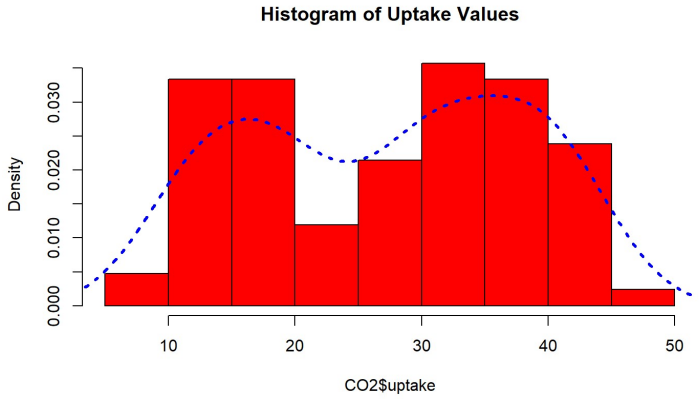
```
plot(density(CO2$uptake), lwd = 3, col = "Blue", lty = 3)
```



Base R Plotting: Overlaying plots

- **Histogram + Kernel Smoother** (Add to plot with `lines`!)
- Use `freq = FALSE` to put histogram on same scale

```
hist(CO2$uptake, main = "Histogram of Uptake Values", col = "Red", freq = FALSE)
lines(density(CO2$uptake), lwd = 3, col = "Blue", lty = 3)
```



Base R Plotting: Boxplot

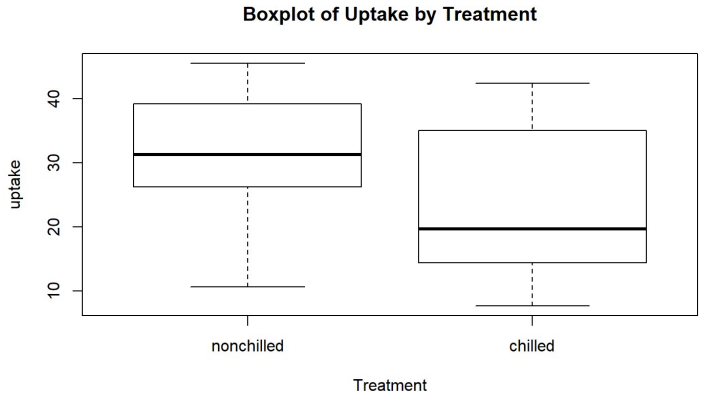
One numerical and one categorical variable

- **Boxplot** - Provides the five number summary in a graph
    - Min, Q1, Median, Q3, Max
    - Often show possible outliers as well
    - Use `boxplot` function

Base R Plotting: Boxplot

- **Boxplot** - Provides the five number summary in a graph

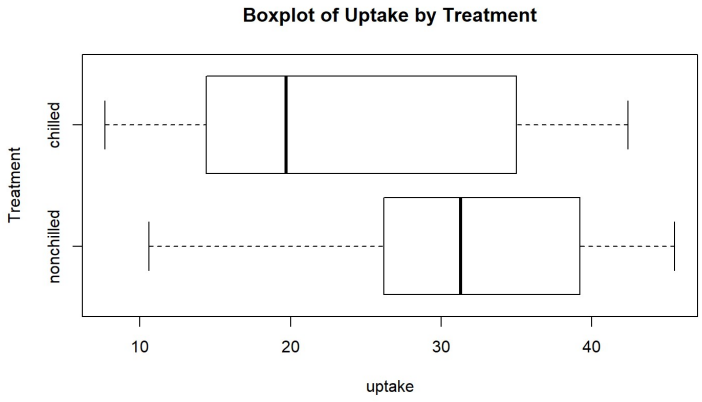
```
boxplot(uptake ~ Treatment, data = CO2, main = "Boxplot of Uptake by Treatment")
```



Base R Plotting: Boxplot

- Change orientation with `horizontal`

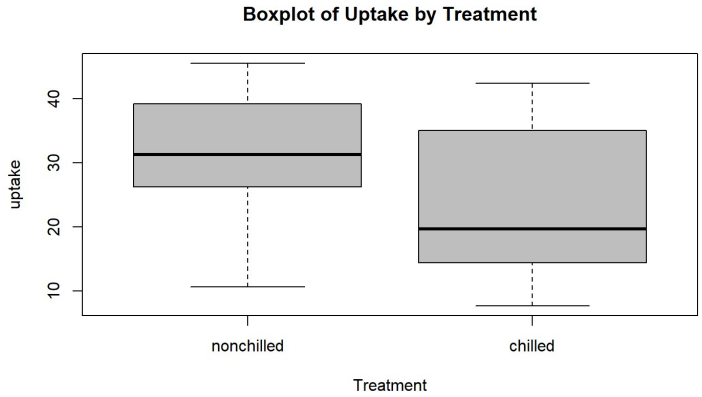
```
boxplot(uptake ~ Treatment, data = CO2,
        main = "Boxplot of Uptake by Treatment", horizontal = TRUE)
```



Base R Plotting: Graphical parameters

- Most graphical parameters can be changed!

```
boxplot(uptake ~ Treatment, data = CO2,
        main = "Boxplot of Uptake by Treatment", col = "Grey")
```



- Could overlay data values with `points` (probably just use `ggplot`!)

Base R Plotting: Scatter plot

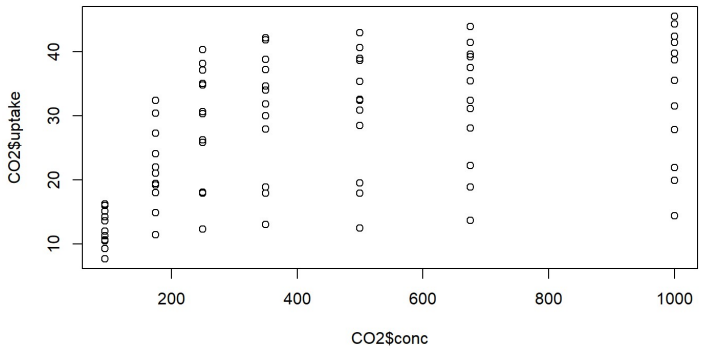
Two numerical variables

- **Scatter Plot** - graphs points corresponding to each observation
  - Use `plot`
  - Give coordinates for points to be plotted as `x` and `y` (usually)
  - Change plot type with `type`
  - Change line type `lty`, point type `pch`, color `color`, etc.

Base R Plotting: Scatter plot

- **Scatter Plot** - graphs points corresponding to each observation

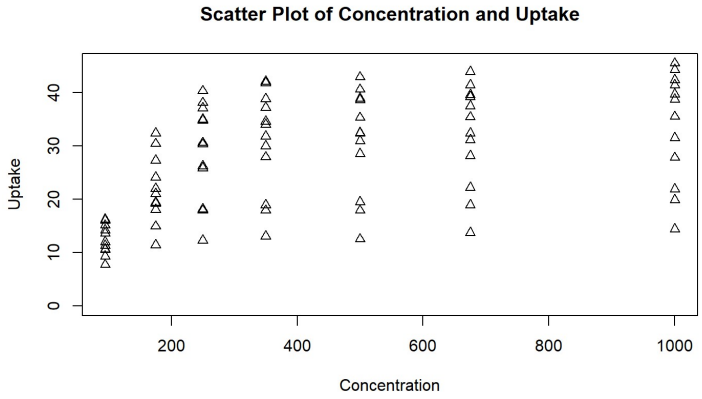
```
plot(x = CO2$conc, y = CO2$uptake)
```



Base R Plotting: `plot` arguments

- Change labels with `xlab`, `ylab`, title with `main`, plot character `pch`, etc.

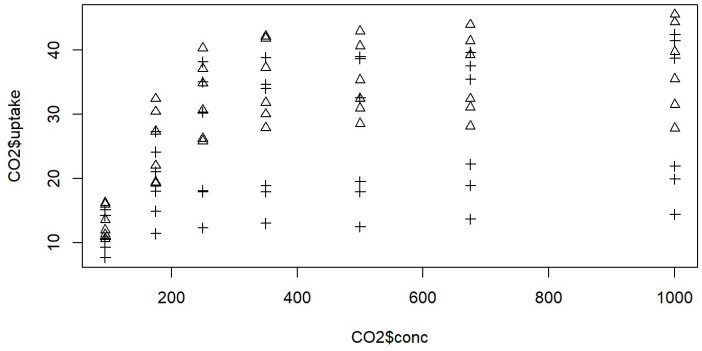
```
plot(x = CO2$conc, y = CO2$uptake, xlab = "Concentration", ylab = "Uptake",
     main = "Scatter Plot of Concentration and Uptake", pch = 2,
     ylim = c(0, max(CO2$uptake)))
```



Base R Plotting: Graphical parameters

- To change plot symbol (or color, etc.) based on another variable, create appropriate vectors to match data

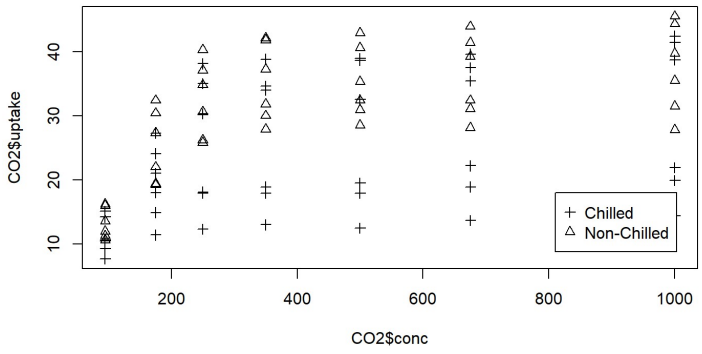
```
symbols <- ifelse(CO2$Treatment == "chilled", 3, 2)
plot(x = CO2$conc, y = CO2$uptake, pch = symbols)
```



Base R Plotting: Legends

- Manually create legend with `legend`

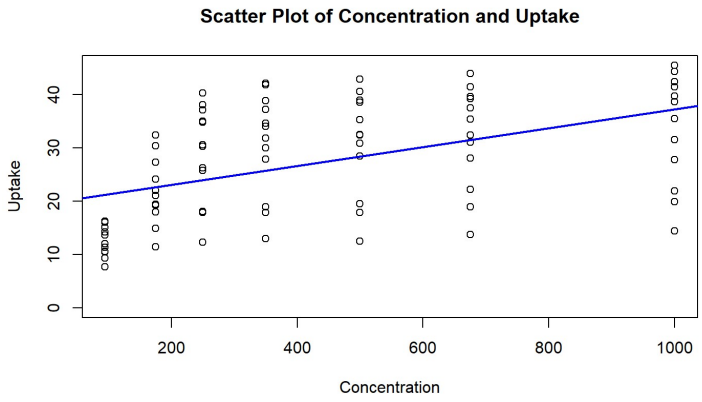
```
plot(x = CO2$conc, y = CO2$uptake, pch = symbols)
legend(x = 810, y = 18, legend = c("Chilled", "Non-Chilled"), pch = c(3, 2))
```



Base R Plotting: Adding a reference line

- Add trend line with `abline`

```
plot(x = CO2$conc, y = CO2$uptake, xlab = "Concentration", ylab = "Uptake",
     main = "Scatter Plot of Concentration and Uptake", ylim = c(0, max(CO2$uptake)))
abline(lm(uptake ~ conc, data = CO2), lwd = 2, col = "Blue")
```



Base R Plotting: Adding text

- May want to add value of correlation to plot
- `paste()` or `paste0()` handy

```
paste("Hi", "What", "Is", "Going", "On", "?", sep = " ")
```

```
## [1] "Hi What Is Going On ?"
```

```
paste("Hi", "What", "Is", "Going", "On", "?", sep = ".")
```

```
## [1] "Hi.What.Is.Going.On.?"
```

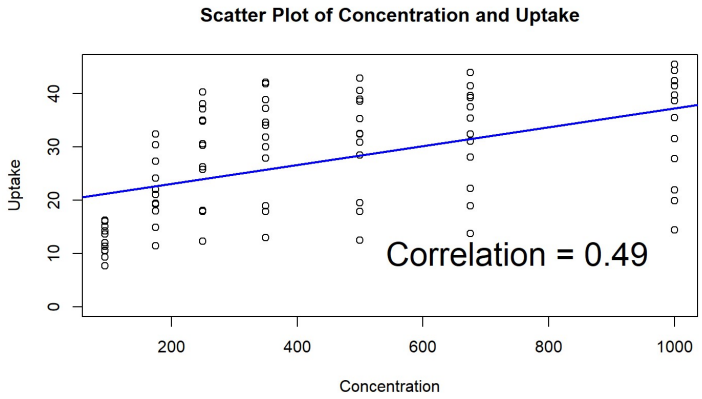
```
paste0("Hi", "What", "Is", "Going", "On", "?")
```

```
## [1] "HiWhatIsGoingOn?"
```

Base R Plotting: Adding text

- Use `text` to add text

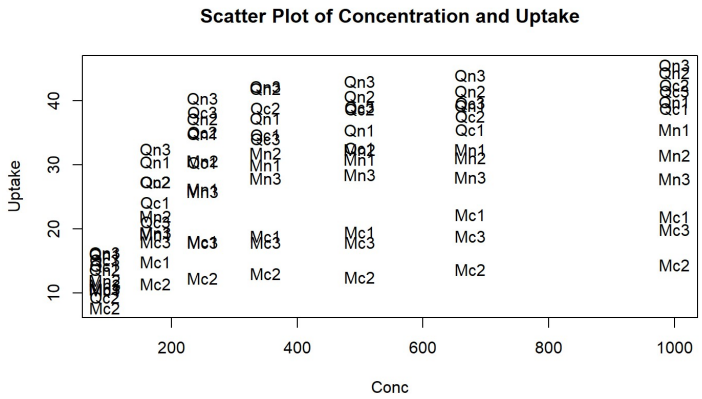
```
correlation <- cor(CO2$conc, CO2$uptake)
plot(x = CO2$conc, y = CO2$uptake, xlab = "Concentration", ylab = "Uptake",
     main = "Scatter Plot of Concentration and Uptake", ylim = c(0, max(CO2$uptake)))
abline(lm(uptake ~ conc, data = CO2), lwd = 2, col = "Blue")
text(x = 750, y = 10, cex = 2,
     label = paste0("Correlation = ", round(correlation, 2)))
```



Base R Plotting: Plotting text

- Text for points easy to do as well: Create an 'empty' plot

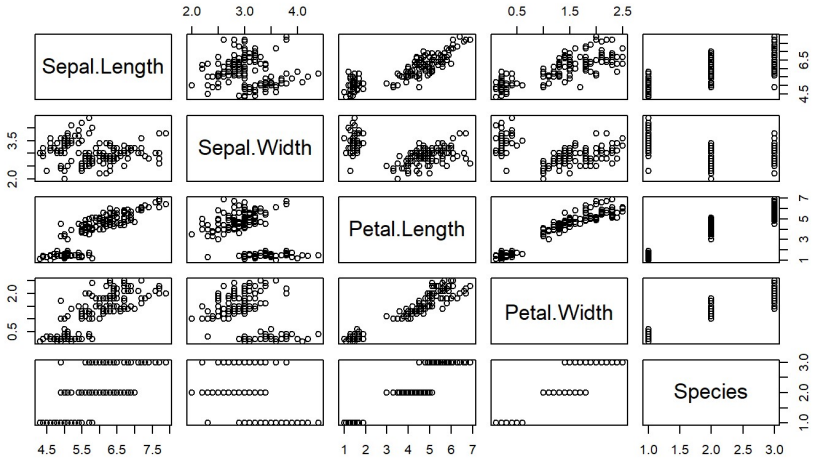
```
plot(NULL, xlim = c(min(CO2$conc), max(CO2$conc)), ylim = c(min(CO2$uptake), max(CO2$uptake)),
     main = "Scatter Plot of Concentration and Uptake", xlab = "Conc", ylab = "Uptake")
text(x = CO2$conc, y = CO2$uptake, label = CO2$Plant)
```



Base R Plotting: Multiple automated graphs

- Useful function: `pairs`

```
pairs(iris)
```



Base R Plotting: Numeric Variables

Recap!

Numeric variable - entries are a numerical value where math can be performed

Most common plots:

- Histogram (`hist`), Density (`plot(density)` or `lines(density)`)
- Boxplot (`boxplot`)
- Scatter plot (`plot`)
- Useful functions: `lines`, `abline`, `points`, `text`
- Graphical parameters: `lty`, `lwd`, `pch`, `cex`, `color`

Base R Plotting: Numeric Variables

Issues:

- Doing plots across another variable not super intuitive…
- Creating vectors for color, point type, etc. not fun…
- Creating legends stinks…

`ggplot2` automates these processes!