Introduction to R for Epidemiologists

Jenna Krall, PhD

Thursday, January 29, 2015

Final project

- Epidemiological analysis of real data
- Must include:
  - Summary statistics
  - T-tests or chi-squared tests
  - Regression
  - Figures
- Can use provided dataset **OR** you may provide your own

If using your own data, it:

- Must have at least 6 variables (with at least 2 continuous variables)
- Must have at least 100 observations
- Must be able to answer a relevant question (e.g. is air pollution associated with mortality?)
- **You must have your data approved by me by March 5**

Outline

1. Introduction to base plotting
2. Customizing plots
3. Multiple figures
4. Margins
5. Other plots
6. Saving plots
7. Rules for displaying data

Introduction to base plotting

Base R comes with excellent graphing capabilities

- Scatterplots
- Histograms
- Box plots

Graphical devices (how your computer represents graphical objects) available in R

- pdf
- postscript
- png
- jpeg

Introduction to base plotting

- base R plots: useful for creating quick plots
- Other graphics packages exist in R
  - Great for "faceting"
  - Great for multiple panels of plots

Other graphics packages

- ggplot2
  - Written by Hadley Wickham (RStudio)
  - We will cover ggplot2 later in the term
- lattice
  - Also useful for multiple panels of plots
  - We will not cover in this course

Introduction to base plotting

Recall Fisher's `iris` data

```
data(iris)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

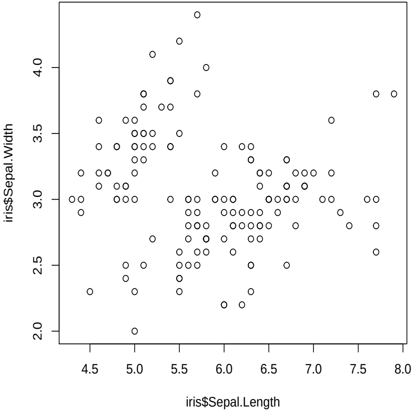
```
head(iris$Sepal.Length)
```

```
## [1] 5.1 4.9 4.7 4.6 5.0 5.4
```

Introduction to base plotting

To create a scatterplot, `plot(x, y)`

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
```

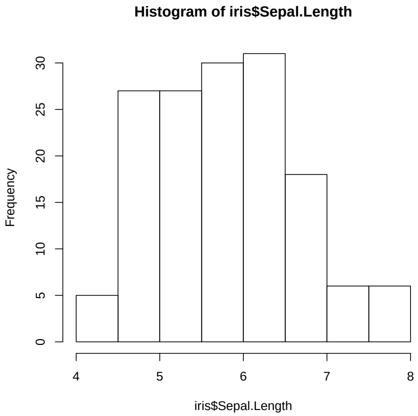


Introduction to base plotting

Histograms show the data distribution for a variable

- The data distribution is the frequency of different values

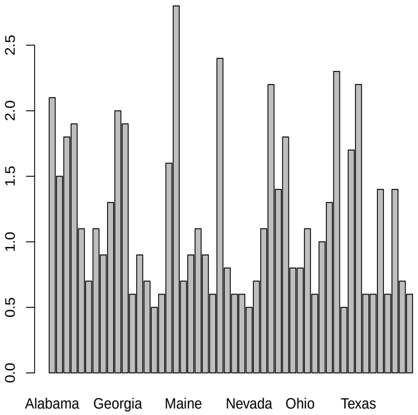
```
hist(iris$Sepal.Length)
```



Introduction to base plotting

Bar plots are used to show the relative frequency of different values of a categorial variable

```
barplot(state.x77[, "Illiteracy"])
```



Customizing plots

What can we change?

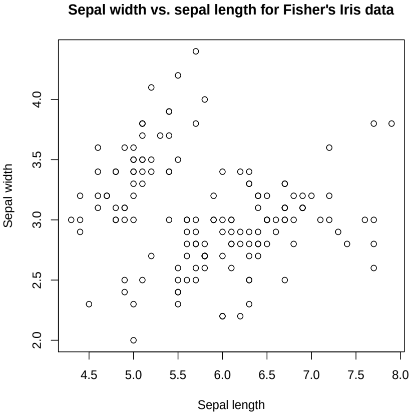
- Add labels
- Change colors
- Change plotting symbol
- Add multiple plots

```
?plot.default
?par
```

Customizing plots

Adding labels

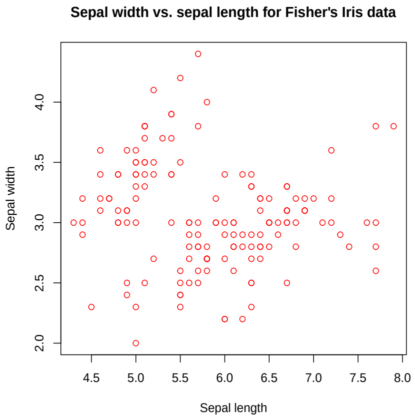
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



Colors

Use the `col` argument in the `plot` function to set color

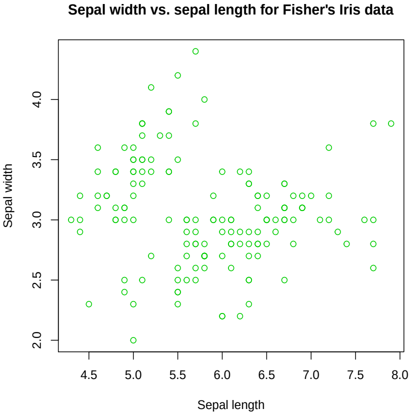
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width", col = "red",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



Colors

We can also use numbers to specify colors:

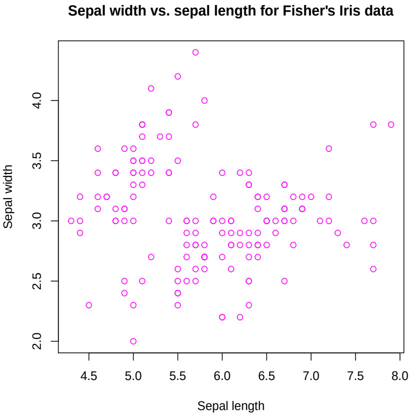
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width", col = 3,
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



Colors

We can also use hexadecimal notation (hex) for the combination of red, green, and blue to specify colors:

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width", col = "#FF00FF",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



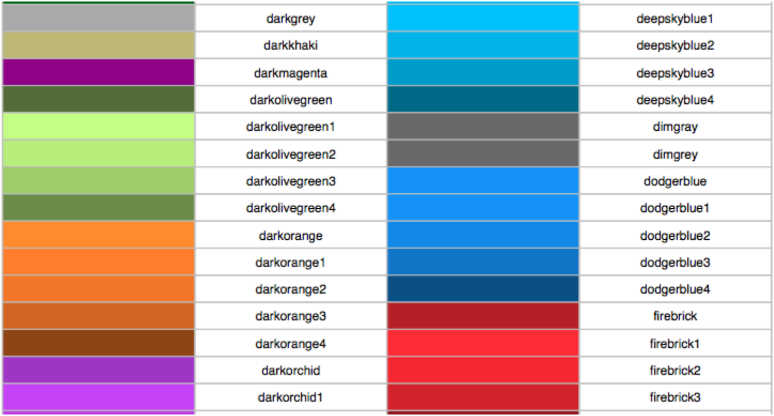
Colors

```r
colors()
```

```
##   [1] "white"                "aliceblue"            "antiquewhite"
##   [4] "antiquewhite1"        "antiquewhite2"        "antiquewhite3"
##   [7] "antiquewhite4"        "aquamarine"           "aquamarine1"
##  [10] "aquamarine2"          "aquamarine3"          "aquamarine4"
##  [13] "azure"                "azure1"               "azure2"
##  [16] "azure3"               "azure4"               "beige"
##  [19] "bisque"               "bisque1"              "bisque2"
##  [22] "bisque3"              "bisque4"              "black"
##  [25] "blanchedalmond"       "blue"                 "blue1"
##  [28] "blue2"                "blue3"                "blue4"
##  [31] "blueviolet"           "brown"                "brown1"
##  [34] "brown2"               "brown3"               "brown4"
##  [37] "burlywood"            "burlywood1"           "burlywood2"
##  [40] "burlywood3"           "burlywood4"           "cadetblue"
##  [43] "cadetblue1"           "cadetblue2"           "cadetblue3"
##  [46] "cadetblue4"           "chartreuse"           "chartreuse1"
##  [49] "chartreuse2"          "chartreuse3"          "chartreuse4"
##  [52] "chocolate"            "chocolate1"           "chocolate2"
##  [55] "chocolate3"           "chocolate4"           "coral"
##  [58] "coral1"               "coral2"               "coral3"
##  [61] "coral4"               "cornflowerblue"       "cornsilk"
##  [64] "cornsilk1"            "cornsilk2"            "cornsilk3"
##  [67] "cornsilk4"            "cyan"                 "cyan1"
##  [70] "cyan2"                "cyan3"                "cyan4"
```

Colors

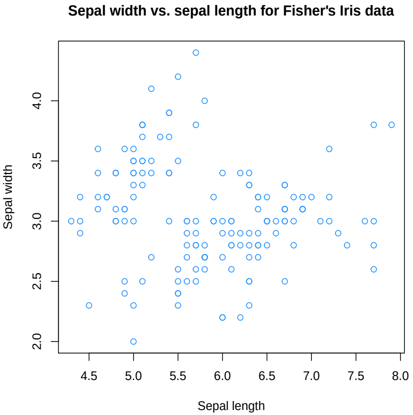
`http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf`



⬜	darkgrey	🟦	deepskyblue1
🟨	darkkhaki	🟦	deepskyblue2
🟪	darkmagenta	🟦	deepskyblue3
🟩	darkolivegreen	🟦	deepskyblue4
🟩	darkolivegreen1	⬛	dimgray
🟩	darkolivegreen2	⬛	dimgrey
🟩	darkolivegreen3	🟦	dodgerblue
🟩	darkolivegreen4	🟦	dodgerblue1
🟧	darkorange	🟦	dodgerblue2
🟧	darkorange1	🟦	dodgerblue3
🟧	darkorange2	🟦	dodgerblue4
🟧	darkorange3	🟥	firebrick
🟫	darkorange4	🟥	firebrick1
🟪	darkorchid	🟥	firebrick2
🟪	darkorchid1	🟥	firebrick3

Colors

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  main = "Sepal width vs. sepal length for Fisher's Iris data",
  ylab = "Sepal width", col = "dodgerblue")
```



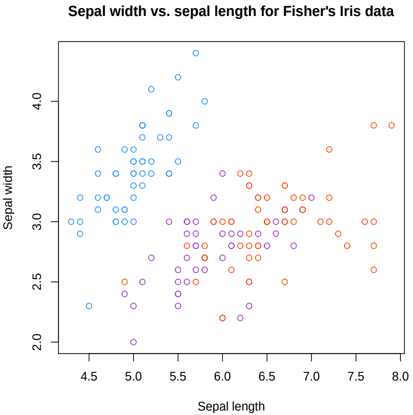
Colors

We can also give `col` a vector:

```r
col_species <- vector(length = length(iris$Species))
col_species[iris$Species == "setosa"] <- "dodgerblue"
col_species[iris$Species == "versicolor"] <- "darkorchid"
col_species[iris$Species == "virginica"] <- "orangered"
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  main = "Sepal width vs. sepal length for Fisher's Iris data",
  ylab = "Sepal width", col = col_species)
```

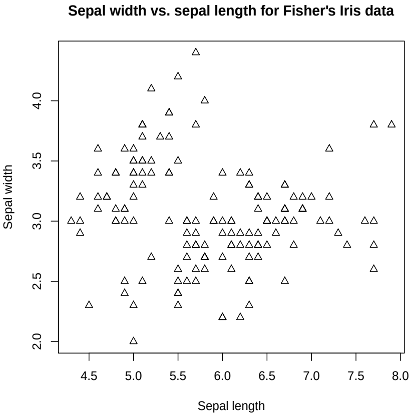
Colors

We can also give `col` a vector:



Plotting symbol

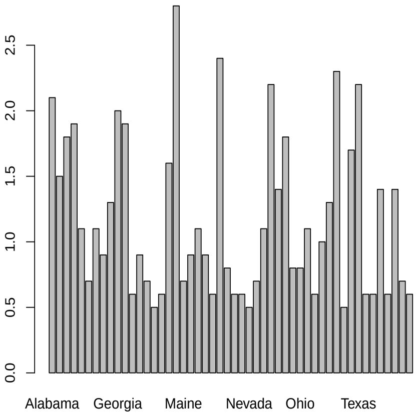
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width", pch = 2,
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



Customizing plots

Bar plots are used to show the relative frequency of different values of a categorial variable

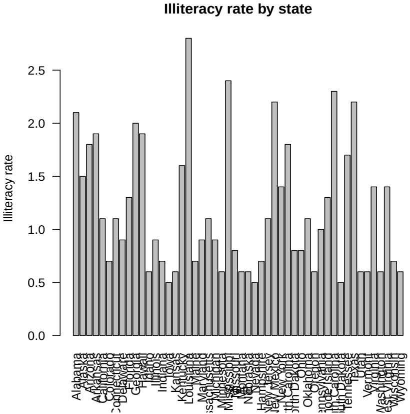
```
barplot(state.x77[, "Illiteracy"])
```



Customizing plots

We can use the las argument in the plot function to change the orientation of the axis

```
barplot(state.x77[, "Illiteracy"], xlab = "State", ylab = "Illiteracy rate",
  main = "Illiteracy rate by state", las = 2)
```



But now the x-axis labels are outside the plot!

Customizing plots

Use par to set global plot options

```
head(par())
```

```
## $xlog
## [1] FALSE
##
## $ylog
## [1] FALSE
##
## $adj
## [1] 0.5
##
## $ann
## [1] TRUE
##
## $ask
## [1] FALSE
##
## $bg
## [1] "transparent"
```

Margins

Change margins using `par(mar = c(bottom, left, top, right))`

Let's first look at the default

- We select the `mar` element from `par()`

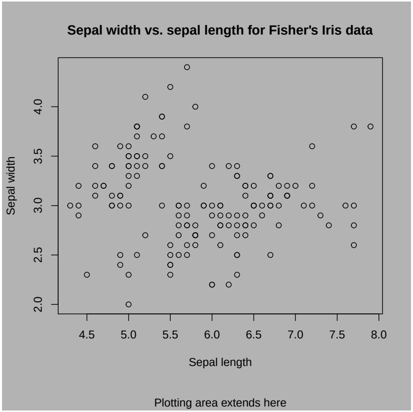
```r
mar.default <- par()$mar
mar.default
```

```
## [1] 5.1 4.1 4.1 2.1
```

Suppose we want to increase the bottom margin by 2

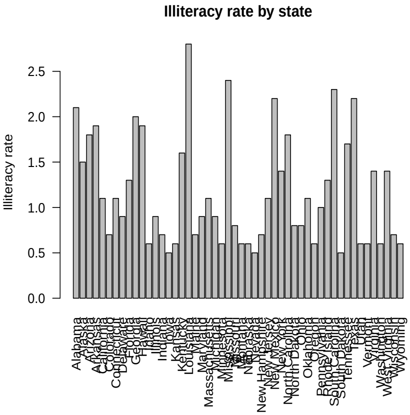
```r
par(mar = c(7.1, 4.1, 4.1, 2.1))
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```

Margins


Sepal width vs. sepal length for Fisher's Iris data
Sepal width
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
Sepal length
Plotting area extends here

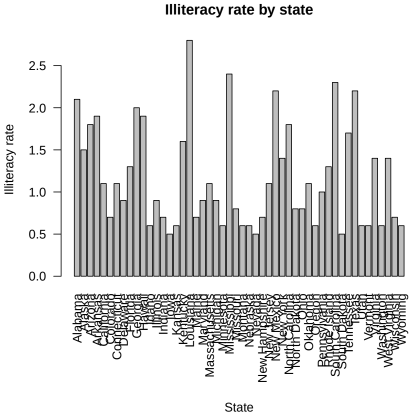
Customizing plots

```r
par(mar = mar.default + c(5, 0, 0, 0))
barplot(state.x77[, "Illiteracy"], xlab = "State", ylab = "Illiteracy rate",
  main = "Illiteracy rate by state", las = 2)
```



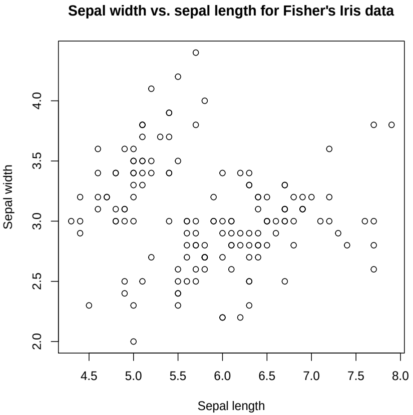
Customizing plots

```r
par(mar = mar.default + c(5, 0, 0, 0))
barplot(state.x77[, "Illiteracy"], xlab = "", ylab = "Illiteracy rate",
  main = "Illiteracy rate by state", las = 2)
mtext("State", side = 1, line = 8)
```



Sizing

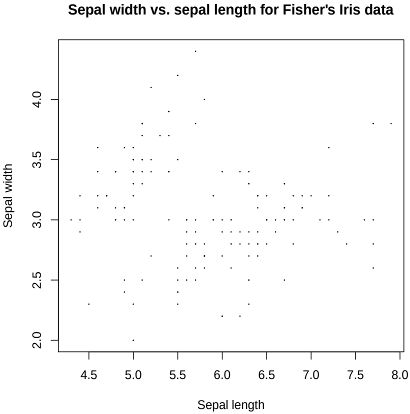
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



Sizing

Decrease size of plotting points

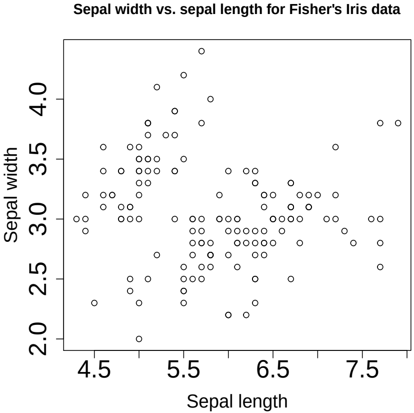
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width", cex = .1,
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



Sizing

Increase size of axis labels and axes:

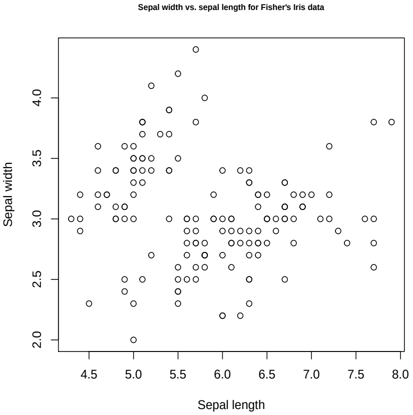
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width", cex.lab = 1.5, cex.axis = 2,
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



Sizing

Look under cex for ?par

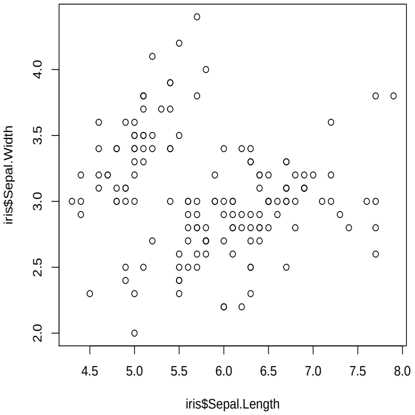
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width", cex.main = .7,
  main = "Sepal width vs. sepal length for Fisher's Iris data")
```



Layering plots

What if we want to add a horizontal line for the mean sepal width?

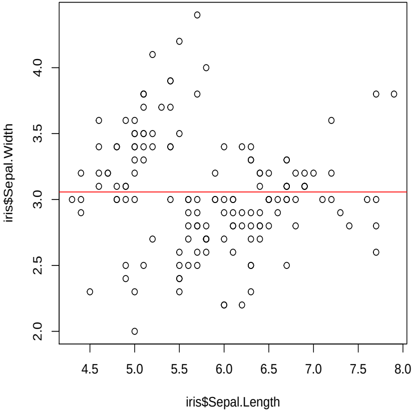
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
```



Layering plots

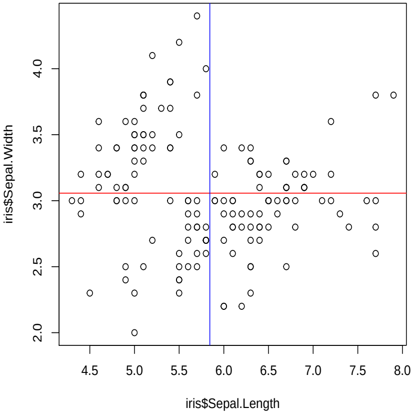
What if we want to add a horizontal line for the mean sepal width?

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
abline(h = mean(iris$Sepal.Width), col = "red")
```



Layering plots

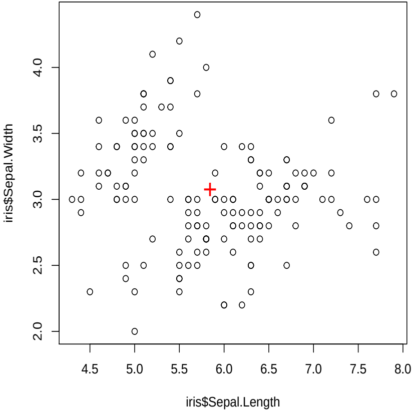
```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
abline(h = mean(iris$Sepal.Width), col = "red")
abline(v = mean(iris$Sepal.Length), col = "blue")
```



Layering plots

Adding a specific point

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
points(x = mean(iris$Sepal.Length), y = mean(iris$Sepal.Width), col = "red",
  pch = "+", cex = 2)
```

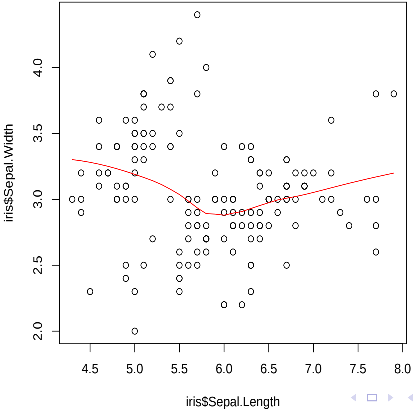


Layering plots

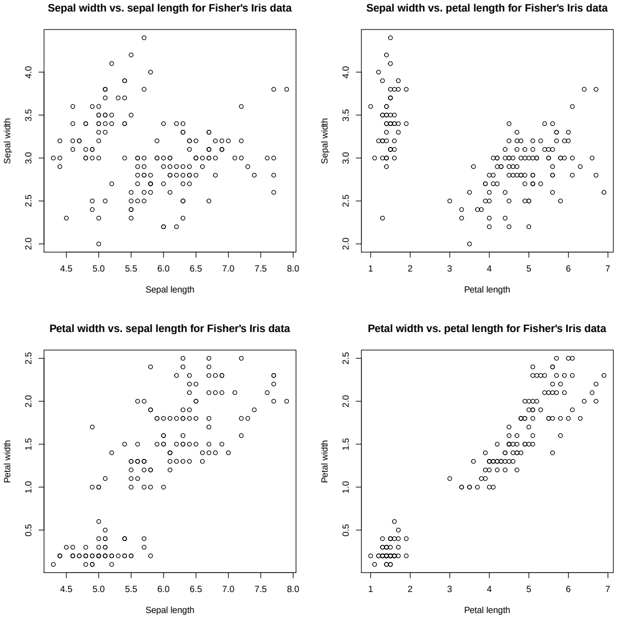
We can also add loess line to scatterplot

- Used to assess direction and magnitude associations or specify breakpoints for regression splines

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
lowess_iris <- lowess(x = iris$Sepal.Length, y = iris$Sepal.Width)
lines(x = lowess_iris$x, y = lowess_iris$y, col = "red")
```



Multiple figures



Multiple figures

The mfrow option in par allows us to plot multiple figures in one plot

- Takes the form `par(mfrow = c(number of rows, number of columns))`

```
# Change par to allow multiple figures
par(mfrow = c(2, 2))
# Create four plots
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
plot(x = iris$Petal.Length, y = iris$Sepal.Width, xlab = "Petal length",
  ylab = "Sepal width",
  main = "Sepal width vs. petal length for Fisher's Iris data")
plot(x = iris$Sepal.Length, y = iris$Petal.Width, xlab = "Sepal length",
  ylab = "Petal width",
  main = "Petal width vs. sepal length for Fisher's Iris data")
plot(x = iris$Petal.Length, y = iris$Petal.Width, xlab = "Petal length",
  ylab = "Petal width",
  main = "Petal width vs. petal length for Fisher's Iris data")
```

Saving plots

- Always set your working directory before saving your plots

Saving your plot as a png (portable network graphic):

- Height and width are in pixels (default is 480 by 480)

```r
png("Iris_scatterplot.png", height = 700, width = 480)
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
dev.off()
```

```
## pdf
##   2
```

Saving plots

Saving your plot as a pdf (portable document format):

- Height and width are in inches (default is 7 by 7)

```r
pdf("Iris_scatterplot.pdf", height = 11, width = 7)
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
dev.off()
```

```
## pdf
##   2
```

Saving plots

What if you don't run `dev.off()`?

- Graphics device does not close
- You will not have your desired output
- Can create multiple pages of plots

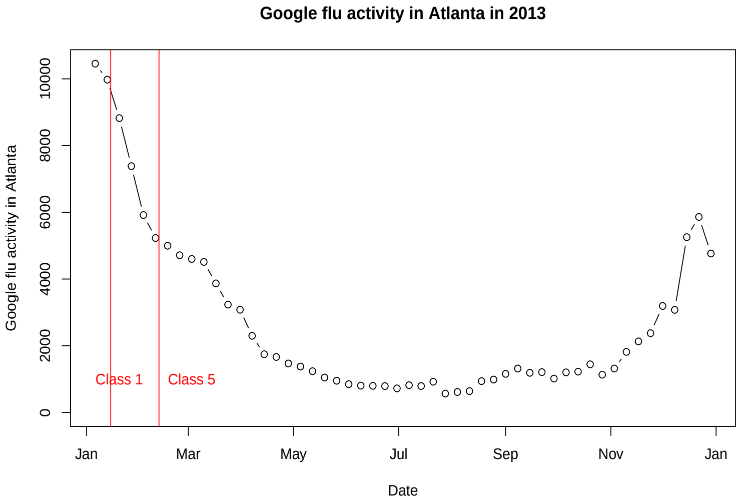
```r
pdf("Iris_scatterplot_2.pdf")
plot(x = iris$Sepal.Length, y = iris$Sepal.Width, xlab = "Sepal length",
  ylab = "Sepal width",
  main = "Sepal width vs. sepal length for Fisher's Iris data")
plot(x = iris$Petal.Length, y = iris$Petal.Width, xlab = "Petal length",
  ylab = "Petal width",
  main = "Petal width vs. petal length for Fisher's Iris data")
dev.off()
```

Last week's plot

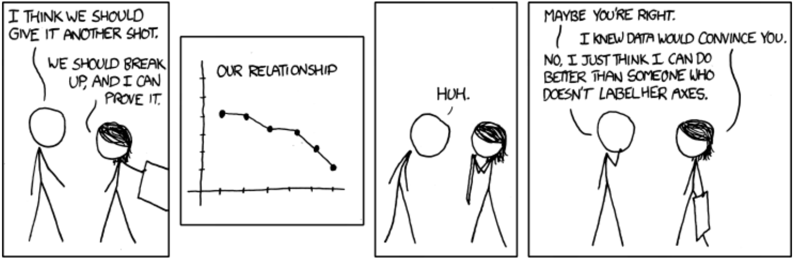
```r
# Load google flu data
load("googleflu.RData")
# Sort by date
flu <- flu[order(flu$Date), ]
# Find years for data
year <- substr(flu$Date, 1, 4)
# Subset to 2013
flu <- flu[year == "2013", ]

# Plot time series for 2013
plot(flu$Date, flu$Atlanta, type = "b", xlab = "Date",
    ylab = "Google flu activity in Atlanta",
    main = "Google flu activity in Atlanta in 2013")
abline(v = as.Date("2013-01-15"), col = "red")
text(labels = "Class 1", x = as.Date("2013-01-20"), y = 1000, col = "red")
abline(v = as.Date("2013-02-12"), col = "red")
text(labels = "Class 5", x = as.Date("2013-03-03"), y = 1000, col = "red")
```

Last week's plot



Rules for displaying data


I THINK WE SHOULD GIVE IT ANOTHER SHOT.
WE SHOULD BREAK UP, AND I CAN PROVE IT.
OUR RELATIONSHIP
HUH.
MAYBE YOU'RE RIGHT.
I KNEW DATA WOULD CONVINCE YOU.
NO, I JUST THINK I CAN DO BETTER THAN SOMEONE WHO DOESN'T LABEL HER AXES.