The Basics of Plotting in R
R has a built-in Datasets Package:

- iris
- mtcars
- precip
- faithful
- state.x77
- USArrests
- presidents
- ToothGrowth
- USJudgeRatings

You can call built-in functions like `hist()` or `plot()` on a built-in data set to quickly produce a chart.
Basic plotting in R

```r
hist(nhtemp)
plot(nhtemp)
```
Basic plotting in R

```r
plot(Petal.Length ~ Petal.Width, data = iris)
```
Basic plotting in R

```r
plot(Petal.Length ~ Petal.Width, col = Species, data = iris)
```
Basic plotting in R

```r
plot(Petal.Length ~ Petal.Width, col = Species, main = "Iris: Petal width vs. length", data = iris)
```
Legends

legend("topleft", legend = unique(iris$Species), fill = unique(iris$Species))
Basic plotting in R

```r
faithful$duration = faithful$eruptions

plot(waiting ~ duration, xlab = "Eruption duration in minutes",
     ylab = "Time waited in minutes",
     data = faithful)

plot(waiting ~ duration, xlab = "Eruption duration in minutes",
     ylab = "Time waited in minutes",
     col = "seagreen", data = faithful)
```
Histograms

```r
hist(faithful$waiting)
```

```r
hist(faithful$eruptions)
```
Histograms

\[
\text{hist(faithful$waiting, breaks = 5)}
\]

\[
\text{hist(faithful$waiting, breaks = 20)}
\]
Histograms

```r
hist(faithful$waiting,
    main = "Old Faithful",
    cex.main = 1.75,
    xlab = "Waiting Time in Minutes",
    ylab = "Frequency",
    cex.lab = 1.25)
```
Histograms

```r
hist(faithful$waiting,
main = "Old Faithful",
cex.main = 1.75,
xlab = "Waiting Time in Minutes",
ylab = "Frequency",
cex.lab = 1.25,
col = "blue",
border = "orange")
```
Stem and leaf plot

> stem(faithful$waiting)

The decimal point is 1 digit(s) to the right of the |

4 | 3
4 | 55566667777888899999
5 | 000011111122222333333444444444
5 | 555556666777888899999999
6 | 000000222233334444
6 | 555667899
7 | 0000111112333333344444
7 | 5555555566666666666667777777777888888888888888888899999999999
8 | 00000000111111111111222222222223333333333333333333333444444444
8 | 555555666666677888888888
9 | 00000012334
9 | 6
> stem(faithful$eruptions)

The decimal point is 1 digit(s) to the left of the |

16 | 070355555588
18 | 0000222333333557777778888822335777888
20 | 000022233788000035778
22 | 0002335578023578
24 | 00228
26 | 23
28 | 080
30 | 7
32 | 2337
34 | 250077
36 | 0000823577
38 | 2333335822225577
40 | 0000003355788888002233555577778
42 | 0333555778800233333555577778
44 | 022233555778000000002333357778888
46 | 000023335770000023578
48 | 00000022335800333
50 | 0370
Bar charts

> races_younger
[1] "Black" "Hispanic" "Other" "White"

> population_in_millions_younger
[1] 10.76 19.03 7.76 40.50

> barplot(population_in_millions_younger,
  names.arg = races_younger,
  main = "Younger than 18, 2014",
  ylab = "Population in Millions")
Bar charts

> races_all
[1] "Native" "Asian" "Black" "Hispanic" "Two" "White"

> population_in_millions_all
[1] 1.39 18.12 38.60 55.61 5.67 195.35

barplot(population_in_millions_all,
        names.arg = races_all,
        main = "All ages, 2014",
        ylab = "Population in Millions")
Stacked and grouped bar charts
Bar charts

> races_all_other
[1] "Black" "Hispanic" "White" "Other"

> population_in_millions_all_other
[1] 38.60 55.61 195.35 25.19

barplot(population_in_millions_all_other,
    names.arg = races_all_other,
    main = "All ages, 2014",
    ylab = "Population in Millions")
Pie charts

```r
pie_labels_younger <-
paste(races_younger,
population_in_millions_younger,
sep = "\n")

pie(population_in_millions_younger,
main = "Younger than 18\nPopulation in Millions",
labels = pie_labels_younger)
```
Pie charts

```r
pie_labels_all_other <-
paste(races_all_other,
population_in_millions_all_other,
sep = "\n")
pie(population_in_millions_all_other,
main = "All Ages\nPopulation in Millions",
labels = pie_labels_all_other)
```
Pie charts

**Younger than 18 Population in Millions**
- Hispanic: 19.03
- Black: 10.76
- White: 40.5
- Other: 7.76

**All Ages Population in Millions**
- Hispanic: 55.61
- Black: 38.6
- Other: 25.19
- White: 195.35
Dot plots

```r
states <- data.frame(state.x77)

sorted_states <- states[order(states$Income), ]

dotchart(sorted_states$Income,
         rownames(sorted_states),
         cex.lab = .25,
         main = "Income per capita")
```
Box plots

```r
boxplot(Petal.Length ~ Species, 
       data = iris, 
       main = "Iris Petal Length by Species", 
       xlab = "Species", 
       ylab = "Petal Length")
```
Notched Box plots

```r
boxplot(len ~ supp * dose,
        data = ToothGrowth,
        notch = TRUE,
        col = (c("gold", "darkgreen")),
        main = "Tooth Growth",
        xlab = "Supplement and Dose")
```

If the notches do not overlap, then the medians of the groups are different (because their confidence intervals do not overlap).
Simple Scatterplots

```r
plot(USJudgeRatings$RTEN ~ USJudgeRatings$FAMI,
     xlab = "Familiarity with Law",
     ylab = "Worthy of Retention",
     main = "Law Familiarity vs. Worthy of Retention",
     pch = "+")
```
Scatterplot Matrices

CONT Number of contacts of lawyer with judge.
INTG Judicial integrity.
DMNR Demeanor.
DILG Diligence.

```r
pairs(~USJudgeRatings$CONT + USJudgeRatings$INTG + USJudgeRatings$DMNR + USJudgeRatings$DILG, main = "US Judge Ratings")
```
features <- c("Contacts", "Integrity", "Demeanor", "Diligence")
pairs(~USJudgeRatings$CONT + USJudgeRatings$INTG + USJudgeRatings$DMNR + USJudgeRatings$DILG, labels = features, main = "US Judge Ratings")
3D Scatterplots

library(scatterplot3d)

scatterplot3d(mtcars$wt, mtcars$disp, mtcars$mpg, main = "3D Scatterplot")
Graphical parameters

- **Text and symbol size** *(cex: .axis, .lab, .main)*
- **Fonts** *(font: .axis, .lab, .main)*:
  - 1=plain, 2=bold, 3=italic, 4=bold italic, 5=symbol
- **Colors** *(col)*: colors, and more colors

http://www.statmethods.net/advgraphs/parameters.html
Graphical parameters (cont’d)

Plotting symbols (pch)

Line type (lty)
ggplot: A special purpose visualization package
Introduction to ggplot

- A package in R for creating data visualizations
- Very powerful and very flexible
- Similar capabilities to base R
- But ggplot’s visualizations are usually preferred
Syntax of ggplot2

gg stands for **grammar of graphics**: the idea that you build visualizations graph from standard components

- data
- a coordinate system
- visual marks that represent data

Image Source
Scatterplot

```r
ggplot(data = txhousing,
       aes(x = listings, y = sales)) +
  geom_point()
```

- **ggplot**: function call
- **data**: a data frame
- **aes**: `x` is the explanatory variable, and `y` is the response variable
- **geom_point**: a scatterplot
Add labels

• `+ labs()` property
• arguments:
  - `title = ""`
  - `subtitle = ""`
  - `x = ""`
  - `y = ""`
  - `caption = ""`

```
+ labs(title = "Texas Active House Listings in relation to Number of Sales", x = "Active House Listings", y = "Number of Sales")
```
Add color

- We can also change color by adjusting the `aes()` inside the geoms.
- Within the `geom_point()` function, add the argument `color` and have it equal the independent variable.

```r
geom_point(aes(color = listings))
```
Bar graph

```r
ggplot(data = midwest, aes(x = state)) + geom_bar()
```

- `ggplot`: function call
- `data`: a data frame
- `aes`: x is the variable
- `geom_bar`: a bar graph
Add labels

- `+ labs()` property
- arguments:
  - `title = ""`
  - `subtitle = ""`
  - `x = ""`
  - `y = ""`
  - `caption = ""`

```
+ labs(title = "Number of Counties in Midwestern States", x = "Midwestern States", y = "Number of Counties")
```
Add color

- We can also change color by adjusting the `aes()` inside the geoms
- Within the `geom_bar()` function, add the argument `fill` and have it equal the independent variable

```
geom_bar(aes(fill=state))
```
Checkoff

https://forms.gle/X63L7DngVoctZh4e7