Importing and visualizing data in R

Day 3

R data.frames

- Like pandas in python, R uses data frame (**data.frame**) object to support tabular data.

- These provide:
  - Data input
  - Row- and column-wise manipulation (e.g., getting, setting data)
  - Data output

Reading delimited files

- Most general:

```
read.table( filename, header=F, sep="" )
```

  - You must specify filename, whether to expect a header (`T/F`), and what the separator is
  - Tab: `"\t"`
  - Comma: `","`
  - Space: `" "`
- Preset defaults (header=T, format-specific delims)

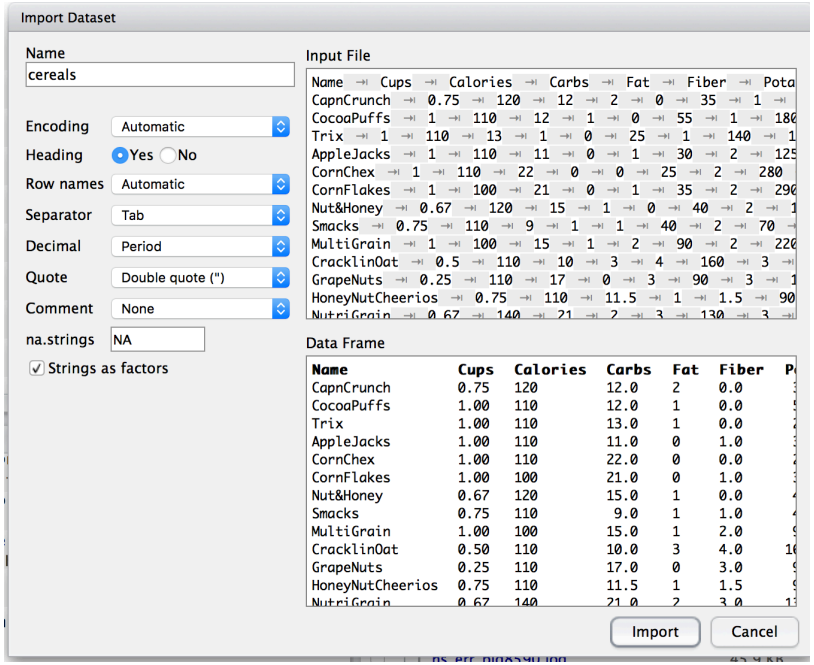
For TSV: `read.delim( filename, ... )`

For CSV: `read.csv( filename, ... )`

Docs: https://stat.ethz.ch/R-manual/R-devel/library/utils/html/read.table.html

Rstudio can automate this part for you...

- Load up Rstudio
- Select Tools → Import Dataset → From Local File....
- Point and click to select settings (column separator, row names, heading, etc).
- Click import, and Rstudio translates your settings into an R read command.




```
> cereals <- read.delim("~/Box Sync/teaching/2016 bioinformat
ics bootcamp/scratch-area/cereals.tsv")
```

Writing out data frames

- Load in cereal table
- Syntax is different than pandas
- Let's make a new column, calories_per_cup (= # calories per cup of cereal)

```
df$new_column = value
```

- Equivalently:

```
df['new_column'] = value
```

- To write out data frame as delimited file:

```
write.table(df, filename, sep="\t", row.names=F)
write.csv( … )
```

Two major plotting options in R

- Base graphics (built-in to R)
  - Prep your data ahead of time (e.g., summarize cereals by manufacturer)
  - Data doesn't need to be in data.frame
  - Run a command, make a plot
  - Run another command, add something to that plot
- ggplot2 (http://docs.ggplot2.org/current/)
  - Have all data points in a data.frame, one per line
  - Implements the grammar of graphics – separates data from plot with a series of abstractions
  - Upshot: it's easy quickly change aspects of the plot (e.g., scatter to histogram)
  - Great for exploratory plotting; final tweaks can be painful.

Base graphics – scatter plot

- Let's make x ~ N(0,1) and y = 2x + e, e ~ N(0, 0.1)

```
x=rnorm(100,0,1)
e=rnorm(100,0,0.1)
y=2*x+e
```

- Now, plot scatterplot of y vs x

```
plot(x,y)
```

- Tweak settings:

```
type="p", "l"
main="…", xlab="…", ylab="…"
cex=... (point magnification, normal=1)
col=... (point color)
pch=... (point type)
lwd=... (line width)
```

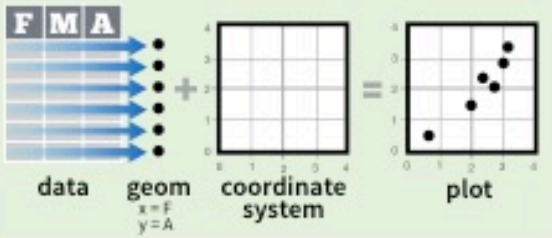
Base graphics – bar chart

- Make a bar plot
- For instance, we have raw data from a poll
- Questions:
  - Are you a choosy mom or dad?
  - Did you choose Jif?
- Have 100 responses, T/F to each question
- But, what we want is % chose jif | is choosy
- With base plots, first make the summary:
  `choosy_sums = table( choosy_data )`
- Then, barplot
  `barplot(choosy_sums)`

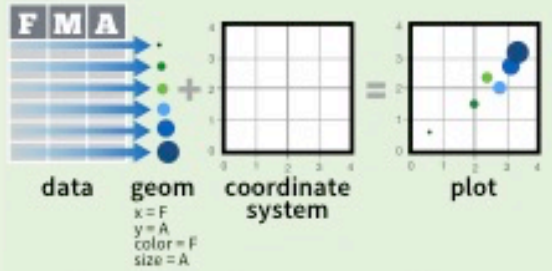
ggplot2

- Data frame with unsummarized data, one point per row
- geom = how to project those data onto a plot
- aes(thetics) = how to map data variables to x, y, color, point size, fill
- Transformations to bin, smooth, or scale data for display

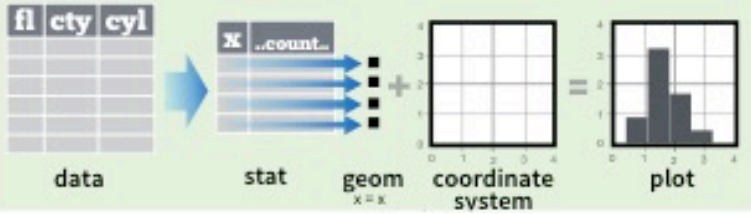
ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same few components: a **data** set, a set of **geoms**—visual marks that represent data points, and a **coordinate system.**



To display data values, map variables in the data set to aesthetic properties of the geom like **size**, **color**, and **x** and **y** locations.



Some plots visualize a **transformation** of the original data set. Use a **stat** to choose a common transformation to visualize, e.g. a + geom_bar(stat = "bin")



http://varianceexplained.org/r/why-I-use-ggplot2/

Opinions vary on which is better

David Robinson
@drob

Follow

Short version of why @jtleek uses base plotting instead of ggplot2:simplystatistics.org/2016/02/11/why… #rstats

2:02 PM - 11 Feb 2016

22 62

http://varianceexplained.org/r/why-I-use-ggplot2/

ggplot2 syntax – make a simple scatterplot

```
library(ggplot2)
```
First, import the library

```
dfxy = data.frame(xvals=x,
         yvals=y)
```
Make a new dataframe from our x,y

```
g = ggplot(data = dfxy)
```
Make a new ggplot using these data

```
g = g + geom_point(
   aes(x=xvals,y=yvals) )
```
Add a geom and map xvals to the x axis and yvals to the y axis

```
g
```
Show the plot!

Slightly more interesting dataset

```
library(ggplot2)

library(reshape2)

head(tips)

g = ggplot(data = tips)

g = g + geom_point(
   aes(x=total_bill,y=tip) )

g
```

First, import the library

This will load a dataframe called tips

Check it out

What if we wanted a histogram instead?

Slightly more interesting dataset

```
library(ggplot2)

library(reshape2)

head(tips)

gbase = ggplot(data = tips)

g = gbase + geom_point(
   aes(x=total_bill,y=tip) )


g
```

First, import the library

This will load a dataframe called tips

Check it out

Can we color points by sex?

Mapping with aes

```
library(ggplot2)

library(reshape2)

head(tips)

gbase = ggplot(data = tips)

g = gbase + geom_point(
   aes(x=total_bill,y=tip,
       colour=sex) )


g
```

First, import the library

This will load a dataframe called tips

Check it out

How about if we want a histogram instead?

Change to a histogram

```
library(ggplot2)

library(reshape2)

head(tips)

gbase = ggplot(data = tips)

g2 = gbase + geom_histogram(
   aes(x=total_bill) )


g2
```

First, import the library

This will load a dataframe called tips

Check it out

What happens here if you map sex to the aesthetic "colour"? Or "fill"?

Faceting for exploratory plotting

```
library(ggplot2)

library(reshape2)

head(tips)

gbase = ggplot(data = tips)

g = gbase + geom_point(
   aes(x=total_bill,y=tip) )

g3 = g + facet_grid( sex ~ .)

g3
```

What if we wanted to know whether men or women are stingier tippers?

Faceting for exploratory plotting

```
library(ggplot2)

library(reshape2)

head(tips)

gbase = ggplot(data = tips)

g = gbase + geom_point(
   aes(x=total_bill,y=tip) )

g3 = g + facet_grid( sex ~ time )

g3
```

What if we wanted to know whether men or women are stingier tippers?

Does meal time matter?

Layering multiple geoms on one plot

```
library(ggplot2)

library(reshape2)

head(tips)

gbase = ggplot(data = tips)

g = gbase + geom_point(
   aes(x=total_bill,y=tip) )

g4 = g + geom_smooth(
   aes(x=total_bill,y=tip),
method='lm' )

g5 = g4 + facet_grid( sex ~ time
)

g5
```