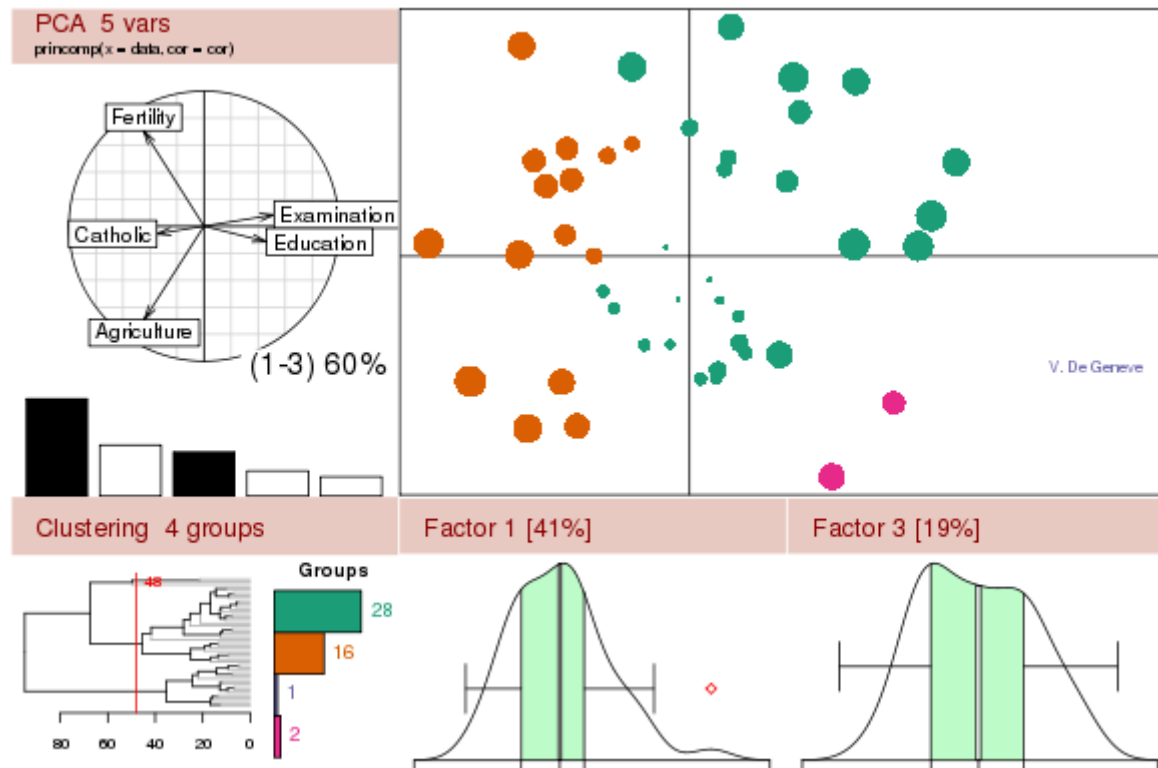
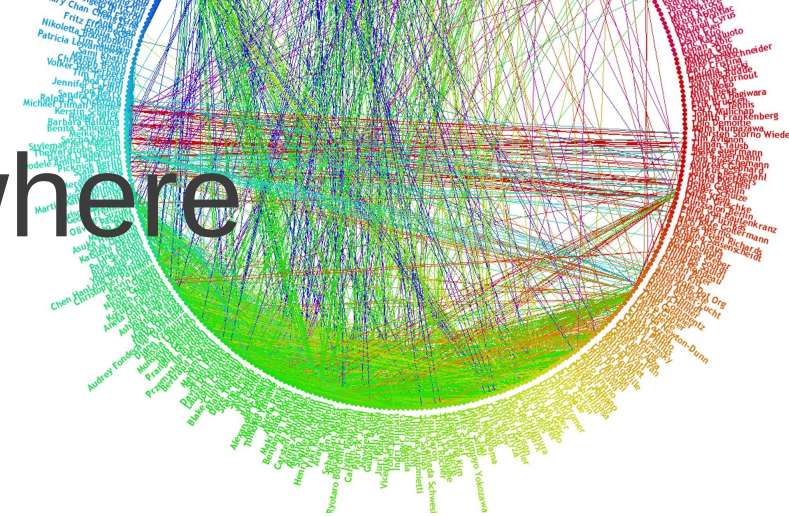


2013 Eric Pitman Summer Workshop in Computational Science



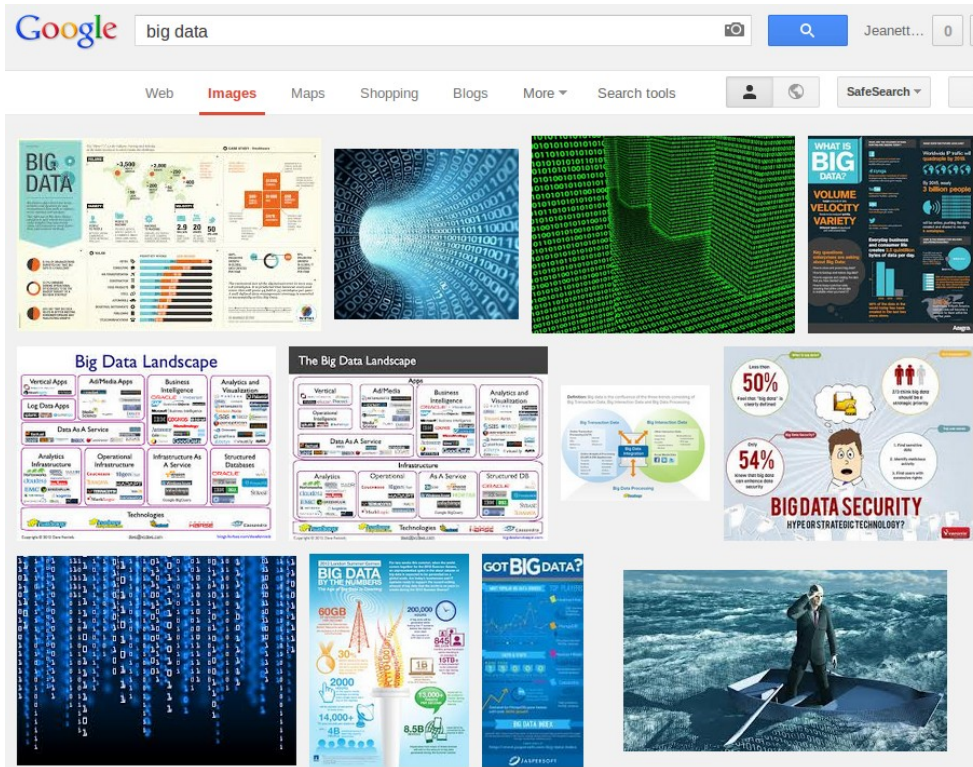
...an introduction to R, statistics, programming, and getting to know datasets

Data is Everywhere



- For example:
 - Science/engineering/medicine
 - Environmental science/Social science/Law enforcement
 - Finance and marketing
 - Social media
- How do we come to an understanding of what a dataset contains?
- Can we draw conclusions from a dataset?
- Let's taste the complexities for ourselves

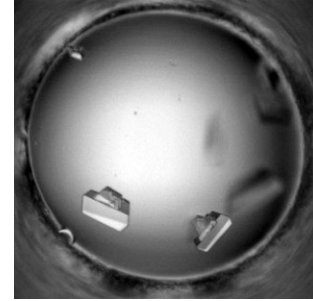
“Big Data” means three things



- *Volume*: lots of data
- *Velocity*: coming at you fast—Twitter, 7TB/day
- *Variety*: text, pictures, video, etc.

Our Plan for the Workshop

- Introduce the R language
- Do some programming
- Examine, model, and visualize datasets
- Project: explore and characterize protein crystallization data from HWI



Intro to

1. Using the command line;
variables;
and a test flight

hpc2 My Tools: R Studio Tool

The screenshot displays the hpc2 My Tools dashboard for user Jeanette Sperhac. The interface is organized into several sections:

- Header:** User profile icon, name "Jeanette Sperhac", and "Dashboard" link.
- My Sessions:** A section showing the current "Workspace" session with a "Storage (manage)" link and a progress bar indicating "0% of 0GB".
- My Groups:** A list of groups: "Center for Computational Research" (approved), "CCR Test Group" (manager), and "jmsperhac group" (manager). It includes buttons for "All My Groups", "All Groups", and "New Group".
- My Tools:** A central section with tabs for "Recent", "Favorites", and "All Tools". It lists several tools: "Interactive Quantum Espresso", "Jmol: 3D viewer for chemical structures in 3D", "Large-scale Atomic/Molecular Massively Parallel Simulator", "NAMD Scalable Molecular Dynamics", "Quantum ESPRESSO", and "R Studio Tool". The "R Studio Tool" is highlighted with a red arrow. Below the list, instructions state: "Add a tool to your favorites by clicking a heart. Click the heart again to remove it."
- My Contributions:** A section titled "Tools" showing contributions: "arith2" (Status: updated), "arith" (Status: published), and "rstudiotool" (Status: published). Each entry includes icons for help, share, and report. Below this is a section for "Other Contributions in Progress" with an entry "Another test document" (Type: Download).
- Left Sidebar:** A navigation menu with links: Dashboard, Profile, Account, Groups (3), Contributions (8), Points, Usage, Favorites, Messages (60), Resume, and Blog.

R Practical Matters



- R is case sensitive (R != r)
- Command line prompt is >
- To run R code: use command line, or save script and `source("script_name")`
- To separate commands, use ; or a newline
- The # character marks a non-executed *comment*
- To display help files:
`?<command-name>` or `??<command-name>`

R as a calculator



> 2 + 3 * 5 # Order of operations.

> (2 + 3) * 5

> 2+3*5 # Equivalent to the above!

Spaces are optional.

> (2+3)*5

On the command line...

R output



```
> 2 + 3 * 5      # In the console
```

```
[1] 17
```

Q: What's that [1] about?

A: R numbers outputs with [n]

Try this in the command line:

```
> 1:500
```

About Comments



> 2 + 3 * 5 # Order of operations.

A comment is:

Text useful to humans, ignored by computer

Helps you understand what code does, or why

Denoted by a pound sign in R

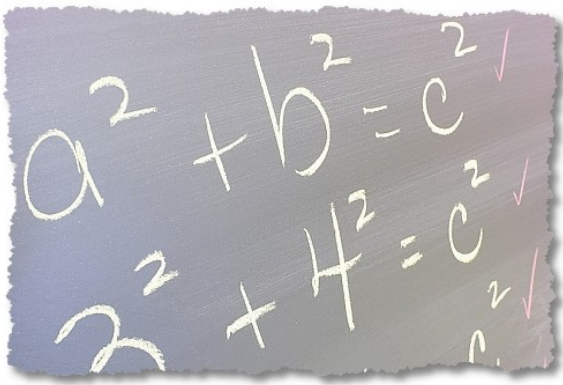
Use them!!

R as a calculator



Try these in your RStudio console:

<code>> 4^2</code>	<code># 4 raised to the second power</code>
<code>> 3/2</code>	<code># Division</code>
<code>> sqrt(16)</code>	<code># Square root</code>
<code>> 3 - 7</code>	<code># Subtraction</code>
<code>> log(10)</code>	<code># Natural logarithm</code>
	<code># with base e=2.718282</code>



Variables: save it

How do we keep a value for later use?

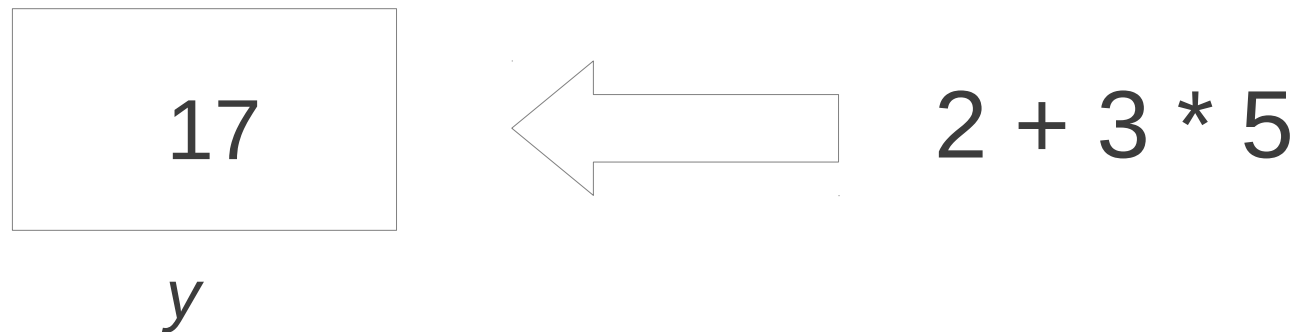
Variable assignment!

```
> y = 2 + 3 * 5      # Do some arithmetic  
> y                  # R stores this value as y  
[1] 17
```

y can be found under Values in the Workspace window

Variable Assignment

`> y = 2 + 3 * 5` # R stores this value as *y*



y can be found under Values in the
Workspace window

Naming Variables in R

A variable name may consist of letters, numbers and the dot or underline characters. It should start with a letter.

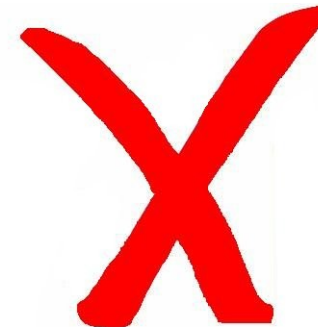
Good:

```
> y = 2  
> try.this = 33.3  
> oneMoreTime = "woohoo"
```

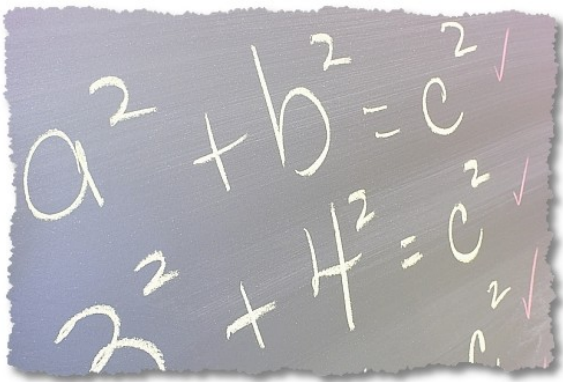


Bad:

```
> 2y = 2  
> _z = 33.3  
> function = "woohoo"
```



* *function* is a reserved word in R



Assign Variables

Try these in your RStudio console:

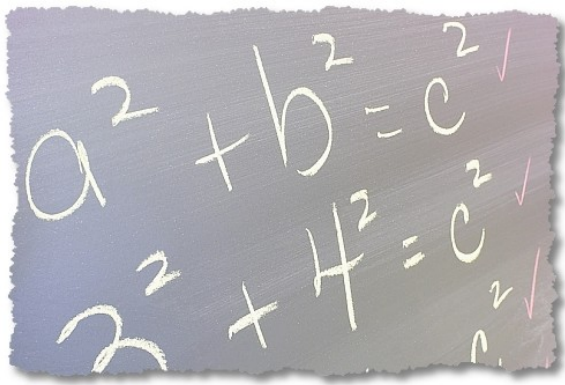
make variable assignments

```
> abc = 3
```

```
> Abc = log(2.8) * pi
```

```
> ABC = "fiddle"
```

Now, check Workspace: Values

A piece of chalkboard with a torn edge showing the Pythagorean theorem written twice in white chalk. The top line is $a^2 + b^2 = c^2$ and the bottom line is $2^2 + 4^2 = c^2$. Each equation has a small red checkmark to its right.

Variables: save it

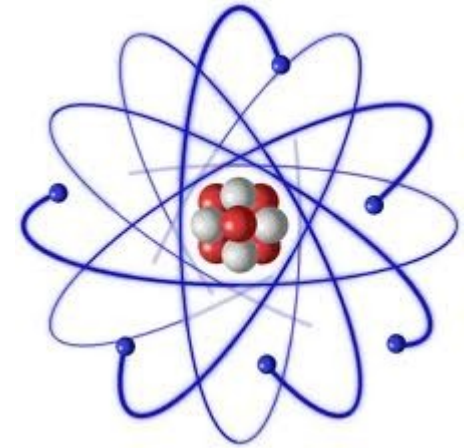
Alternate R syntax for assignment

```
> y = 2 + 3 * 5
```

```
> z <- 2 + 3 * 5      # Same thing as y
```

Variable assignment: Use = or <-

R's atomic data types



Let's take a look at some available data types:

- Numeric (includes integer)

3.14, 1, 2600

- Character (string)

"hey, I'm a string"

- Logical

TRUE or FALSE

- NA

No value known

Numeric data



Find the type of a variable using class()

```
> class(8)                                # numeric type  
[1] "numeric"
```

```
> class(6.02e+24)                         # numeric type  
[1] "numeric"
```

```
> class(pi)                              # numeric type (predefined in R)  
[1] "numeric"
```

Character and Logical data

Find the type of a variable using class()

```
> class("phooey") # character type:  
[1] "character"      # notice the quotes
```

```
> class(TRUE)     # logical type: no quotes  
[1] "logical"
```

```
> class(NA)        # NA (no quotes) means "no value known"  
[1] "logical"
```



RStudio test flight



To whet your appetite for RStudio, let's try:

- Using the editor
- Entering data
- Making a plot in R
- Sourcing a file

The M&M Exercise



On your workstation:

- Sign in to hpc2.org
- Start the RStudio tool
- Create/Access Project from GitHub

`git://github.com/jsperhac/workshop-dev.git`

- Files pane: click *examples*, then
`mm-single-example.R`

The M&M Exercise



Inside mm-single-example.R:

- Change the M&M color counts in the `mv` variable
- Edit `ptitle`, if you want

```
# EDIT HERE: ...
```

```
mv1 = c("red", "blue", "green", "yellow", "orange", "brown")
```

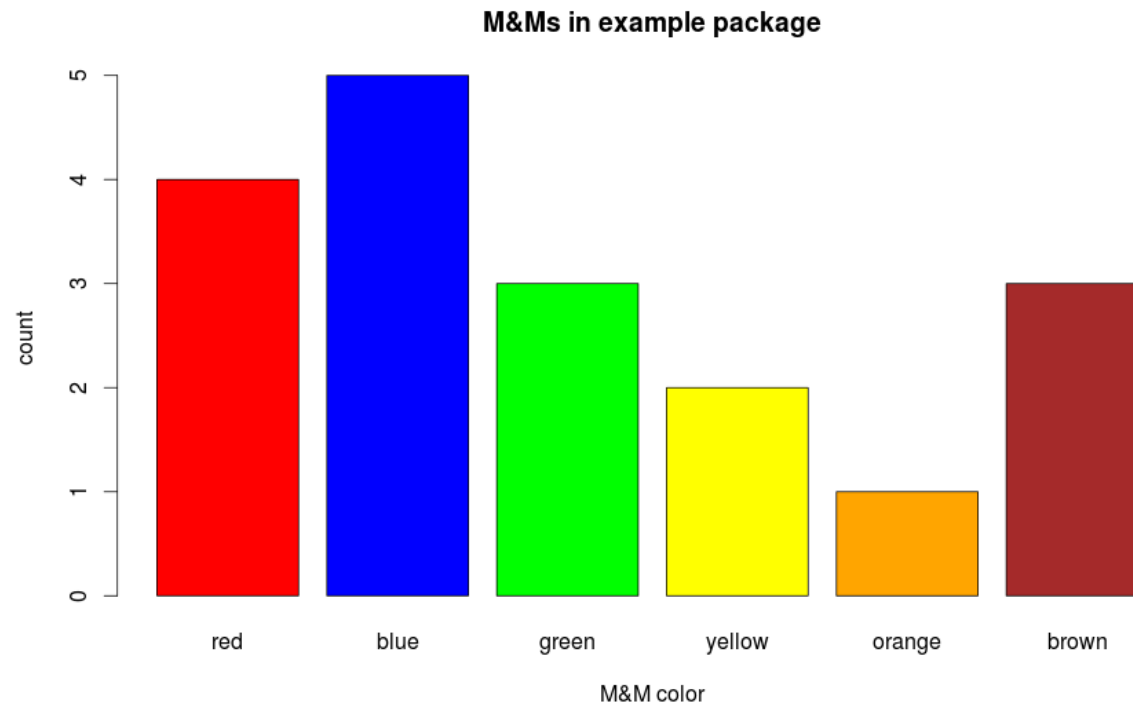
```
mv = c( 4,    5,    3,    2,    1,    3)
```

```
ptitle = "M&Ms in example package"
```


The M&M Exercise

Inside mm-single-example.R:

- Save the file (File:Save)
- Source the file (Source button)



The M&M Exercise



Questions:

- What have you plotted?
- What outputs does R provide in the console?
- What variables were created?
- What else happens inside this source file?

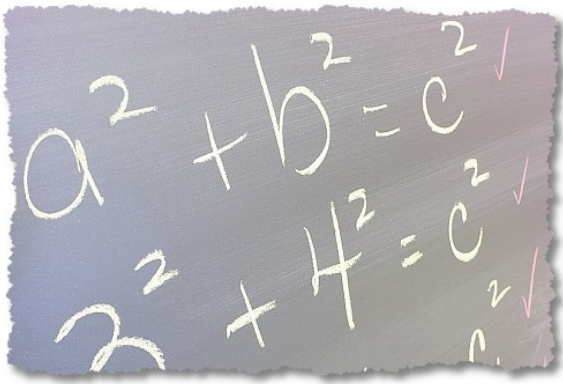
OK, now you can eat...

Using Logical Operators



<code>1==2</code>	<i># equivalence test: double equals</i>
<code>9 != 19</code>	<i># “not equal” test</i>
<code>3 < 204</code>	<i># less-than test</i>
<code>18 > 44</code>	<i># greater-than test</i>
<code>“tree”==89</code>	<i># comparing mixed data types</i>

What should the results of these tests be?



A logical test

Compare R syntax for assignment

```
> y = 2 + 3 * 5
```

```
> z <- 2 + 3 * 5    # Same thing as y
```

```
> y==z              # Here's the test...
```

```
[1] TRUE
```

Logical data



A logical value is often created from a comparison between variables.

$u \ \& \ v$ # Are u AND v both true?

$u \ | \ v$ # Is at least one of u OR v true?

$!u$ # “NOT u ” flips the logical value of
 # variable u

Learning about Object x



R stores everything, variables included, in Objects.

Object x



```
> x <- 2.71
```

```
> print(x)           # print the value of the object
```

```
[1] 2.71
```

```
> class(x)           # what data type or object type?
```

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

```
> is.na(x)           # is.na() tests whether a value has a  
                      # known value
```

```
[1] FALSE
```


Interlude

Complete variable/atomic datatype exercises.



Open in the RStudio source editor:

`<workshop>/exercises/exercises-variables-atomic-datatypes.R`