Introduction to Programming in R

Matthew K. Lau

Cottonwood Ecology Group
Department of Biological Sciences
Northern Arizona University

July 12, 2011

Introductions

Why learn R?

On a recent flight from Tokyo to Beijing, at around the time my lunch tray was taken away, I remembered that I needed to learn Mandarin. "...dammit," I whispered, "I knew I forgot something."
– David Sedaris (New Yorker, July 2011)

Three Reasons

- Software does not *have* to limit analysis.

Three Reasons

- Software does not *have* to limit analysis.
- Scripting can save time and effort.

Three Reasons

- Software does not *have* to limit analysis.
- Scripting can save time and effort.
- Free and Free

Three Reasons

- Software does not *have* to limit analysis.
- Scripting can save time and effort.
- Free = No Cost and Free = Open Source

Three Reasons

- Software does not *have* to limit analysis.
- Scripting can save time and effort.
- Free = No Cost and Free = Open Source

What can R do?

- Math (e.g. linear algebra)
- Basic Statistics
- Publication quality plots
- Simulations
- Database interfacing
- GIS
- Phylogenetics
- Multivariate statistics
- Network analysis
- Bayesian statistics
- Animations
- Make julienned fries
- and much MUCH MORE...

Requested Topics

- The basics (What the heck!)
- Data input and management
- Bayesian statistics (interfacing with Win-Bugs)
- Growth curve analyses
- Summary statistics from large datasets
- GLM, GAM and Mixed Models

Lasciate ogne speranza, voi ch'intrate.
– Dante Aligheri (The Inferno)

Abandon all hope, ye who enter here.
– Dante Aligheri (The Inferno)

Getting Started

- Open R.
- Say hello to the command line.

Think Intuitively

Remember, no one's listening but **R**.

Math in R

What do you get when you add two and two?

?

Math in R

What do you get when you add two and two?

```
> 2 + 2
[1] 4
```

Getting Help.

- `?`, `??`, `help`
- Googling
- Manuals and Books

?

How do you use ? to learn about `help`?

?

How do you use ? to learn about `help`?

```
> ?  help
```

?

What other math commands are there?

?

What other math commands are there?

```
> ?   '+'
```

Objects

How do you create an object?

```
> x = 2 + 2
```

Objects

What is the difference between x and 'x'?

```
> x
[1] 4
> "x"
[1] "x"
```

Objects

Can you do operations with objects?

Objects

Can you do operations with objects?

```
> x = 2 + 2
> y = 2 + 2
> x + y
```

```
[1] 8
```

```
> x * y
```

```
[1] 16
```

```
> x/y
```

```
[1] 1
```

Objects

What can I name objects?

- Letters
- Symbols (e.g. "." and "_")
- Numbers (following a letter)
- Keep them short (<=7 characters)
- Avoid function names (e.g. data, factor, sqrt)

Objects

How do I know if I have created an object already? How do you I get rid of them?

```
> ls()
> rm()
```

*NOTE: `rm(ls())` will remove all objects in your workspace.

Functions

`function`(*arguments*,...)

Functions - Cognates

What do you think the functions are for the mean and standard deviation?
Apply them to our object `x`.

Functions - Cognates

What do you think the functions are for the mean and standard deviation?
Apply them to our object `x`.

```
> mean(x)
[1] 4
> mean(mean(x))
[1] 4
> sd(x)
[1] NA
```

Comprehensive R Archive Network (CRAN)

- How is CRAN organized?
- CRAN.r-project.org to download R.
- Also a resource for help files, manuals and other resources.

Work Flow Outline

1. Create a project working directory.
2. Create a data folder.
3. Place data in data folder.
4. Open and save a new script in the working directory.
5. Set the working directory.
6. Call files by 'filename' or data by data/'filename'.

Scripting

1. Long tasks are awkward in the command line.
2. Command line entries are not saved.
3. Scripted code can be annotated.

Scripting

1. Open a new script window.
2. Save it to your working directory.
3. What happens when you run "# 2+2" using `CTRL R`?

Scripting

1. Open a new script window.
2. Save it to your working directory.
3. What happens when you run "# 2+2" using CTRL R?

```
>#  2+2
>
```

Scripting

Scripting Tips

1. Start each script with meta-data.
2. Annotate each section and individual lines if possible.
3. Be descriptive but succinct, like a lab journal.

```
#Matthew K. Lau 10July2011
#Script from Introduction to R
class at UNCW.

#How to do basic math operations.
2+2 #addition
2-2 #subtraction
2*2 #multiplication
```

Scripting

Scripting Tips

1. Start each script with meta-data.
2. Annotate each section and individual lines if possible.
3. Be descriptive but succinct, like a lab journal.

```
#Matthew K. Lau 10July2011
#Script from Introduction to R
class at UNCW.

#How to do basic math operations.
2+2 #addition
2-2 #subtraction
2*2 #multiplication
```

***You'll thank yourself later when you easily remember what you were doing when you wrote your script.**

Importing

- Entering by hand (`:`, `c`)
- Importing from a file (`read.csv`)

Entering data by hand (:, c

How do you create a vector of integers from 1:5?

```
> 1:5
[1] 1 2 3 4 5
```

Entering data by hand (:, c)

How do you create a vector of integers from 1:5?

```
> c(1, 2, 3, 4, 5)
[1] 1 2 3 4 5
```

Entering data by hand (:, c)

How do you create a vector of integers from 1:5?
(Hint: Create an object)

Entering data by hand (:, c)

How do you create a vector of integers from 1:5?
(Hint: Create an object)
NOTE: Our object `x` was an integer, but now it's a vector.

```
> x = 1:5
> x

[1] 1 2 3 4 5
```

Data Types

- (Scalars)
- Vectors (`numeric`, `character`)
- Matrices and Arrays (`matrix`, `array`)
- Data Frames (`data.frame`)
- Lists (`list`)
- ...

Vectors (numeric, character)

How do you know what type of data you have?

```
> class(x)
[1] "integer"
> mode(x)
[1] "numeric"
> y = c(1, 2, 2.5, 3, 5)
> class(y)
[1] "numeric"
> mode(y)
[1] "numeric"
```

Vectors (numeric, character)

How do change data types?

```
> x = as.character(x)
> class(x)
[1] "character"
> mode(x)
[1] "character"
```

Matrices and Arrays

How do you create a matrix?

```
> M = matrix(data = 1:9, nrow = 3, ncol = 3)
> M
```

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Matrices and Arrays

How do you create an array?

Matrices and Arrays

How do you create an array?

```
> A = array(1:9, dim = c(3, 3))
> A
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> class(A)
[1] "matrix"
> mode(A)
[1] "numeric"
```

Matrices and Arrays

What about connecting two vectors together into columns?

```
> x = 1:5
> y = 1:5
> cbind(x, y)
```

```
     x y
[1,] 1 1
[2,] 2 2
[3,] 3 3
[4,] 4 4
[5,] 5 5
```

Matrices and Arrays

What about connecting two vectors together into rows?

```
> x = 1:5
> y = 1:5
> rbind(x, y)
  [,1] [,2] [,3] [,4] [,5]
x    1    2    3    4    5
y    1    2    3    4    5
```

Data Frames

How do you create a data frame from numeric and character vectors?

Data Frames

How do you create a data frame with a numeric and a character vector?

```
> x = 1:3
> y = c("a", "b", "c")
> data.frame(x, y)

  x y
1 1 a
2 2 b
3 3 c
```

Lists

How do you create a list with `m` and `a`?

Lists

How do you create a list with `m` and a?

```
> list(M, A)
[[1]]
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

[[2]]
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Importing a Matrix

How do you import a matrix?

Data can be found at `http://perceval.bio.nau.edu/downloads/igert/IntroR-Course_Notes/CommData.csv`

```
> com = read.csv("data/CommData.csv")
```

Quick Summaries of Data

```
> head(com)
```

	env	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V
1	1	321	179	179	143	36	179	179	179	250	179	0	0	71	1
2	1	250	143	36	214	214	214	250	214	0	107	0	36	107	
3	1	71	250	107	179	143	107	179	250	143	179	0	0	36	
4	1	36	250	71	107	71	179	143	250	36	357	0	71	0	
5	1	179	107	143	214	0	143	107	179	143	250	36	36	36	
6	1	357	107	143	286	179	179	179	179	250	286	0	36	143	

	V19	V20	V21	V22	V23	V24	V25	V26	V27	V28	V29	V30
1	0	71	0	71	36	0	36	36	0	0	107	36
2	36	0	36	36	36	71	71	36	0	143	71	36
3	36	36	36	71	36	143	36	0	36	71	0	0
4	107	71	36	36	0	36	36	36	0	0	36	36
5	36	71	0	36	36	36	0	71	36	71	36	143

Quick Summaries of Data

```
> summary(com)
      env                V1                V2                V3
 Min.   :1.0   Min.   :  0.0   Min.   :  0.0   Min.   :  0.
 1st Qu.:1.0   1st Qu.:  0.0   1st Qu.: 36.0   1st Qu.: 36.
 Median :1.5   Median : 71.0   Median :107.0   Median : 89.
 Mean   :1.5   Mean   :107.1   Mean   :109.0   Mean   : 92.
 3rd Qu.:2.0   3rd Qu.:187.8   3rd Qu.:152.0   3rd Qu.:152.
 Max.   :2.0   Max.   :357.0   Max.   :250.0   Max.   :250.
       V5                V6                V7                V8
 Min.   :  0.00   Min.   :  0.00   Min.   :  0.0   Min.   :
 1st Qu.: 36.00   1st Qu.: 36.00   1st Qu.: 36.0   1st Qu.:
 Median :107.00   Median :107.00   Median :107.0   Median :
 Mean   : 96.45   Mean   : 98.35   Mean   :103.8   Mean   :
 3rd Qu.:143.00   3rd Qu.:152.00   3rd Qu.:152.0   3rd Qu.:
```

Manipulating

How do you pull out a single value from a vactor?

```
> x = 1:5
> x[3]

[1] 3
```

Manipulating

How do you isolate multiple values?

```
> x = 1:5
> c(1, 2, 3)
[1] 1 2 3
> x[c(1, 2, 3)]
[1] 1 2 3
```

Manipulating

How do you isolate multiple values? Is there a simpler way?

Manipulating

How do you isolate multiple values? Is there a simpler way?

```
> x = 1:5
> 1:3
[1] 1 2 3
> x[1:3]
[1] 1 2 3
```

Manipulating

How do you subset a vector?

```
> x = 1:5
> c(FALSE, FALSE, FALSE, TRUE, FALSE)
[1] FALSE FALSE FALSE  TRUE FALSE
> x[c(FALSE, FALSE, FALSE, TRUE, FALSE)]
[1] 4
```

Manipulating

How do you subset a vector?

```
> x == 4
[1] FALSE FALSE FALSE  TRUE FALSE
> x[x == 4]
[1] 4
> x != 4
[1]  TRUE  TRUE  TRUE FALSE  TRUE
```

Manipulating

How do you isolate all of the values of `x` greater than or equal to 2?

Manipulating

How do you isolate all of the values of `x` greater than or equal to 2?

```
> x[x >= 2]
[1] 2 3 4 5
```

Manipulating

How do you isolate one number from a matrix?

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
> M[2, 3]
[1] 8
```

Manipulating

How do you isolate a whole row or column from a matrix?

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
> M[1, ]
[1] 1 4 7
> M[, 1]
[1] 1 2 3
```

Manipulating

How can you get all of the values of `M[1,]` that are greater than 2?

Manipulating

How can you get all of the values of `M[1,]` that are greater than 2?

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
> M[1, ]
[1] 1 4 7
> M[1, ] > 2
[1] FALSE  TRUE  TRUE
> M[1, M[1, ] > 2]
[1] 4 7
```

Manipulating

How do you sort a matrix by the first column?

```
> order(M[, 1])
[1] 1 2 3
> order(M[, 1], decreasing = TRUE)
[1] 3 2 1
```

Manipulating

How do you sort a matrix by the first column?

```
> M[order(M[, 1]), ]
```

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
> M[order(M[, 1], decreasing = TRUE), ]
```

```
     [,1] [,2] [,3]
[1,]    3    6    9
[2,]    2    5    8
[3,]    1    4    7
```

Manipulating

How do you sort a matrix by the first row?

Manipulating

How do you sort a matrix by the first row?

```
> M[1, ]
[1] 1 4 7
> order(M[1, ])
[1] 1 2 3
> M[, order(M[1, ])]
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Manipulating

How would you sort our matrix (`com`) by the column names?

Manipulating

```
> colnames(com)
 [1] "env" "V1"  "V2"  "V3"  "V4"  "V5"  "V6"  "V7"  "V8"
[13] "V12" "V13" "V14" "V15" "V16" "V17" "V18" "V19" "V20"
[25] "V24" "V25" "V26" "V27" "V28" "V29" "V30"
```

Manipulating

How would you sort our matrix (`com`) by the column names?

```
> com[, order(colnames(com))]
```

Manipulating

Can I refer to the columns by name?

```
> colnames(com)[1:3]

[1] "env" "V1"  "V2"

> attach(com)
> com[order(V1), ]
> detach(com)
> com$V1
> com$V2
```

Watch Out!

1. `x and 'x'`
2. `x == x and x=x`
3. `T` and `F` are `TRUE` and `FALSE` by default, unless you change them.

Exporting

1. Pick a file format (I recommend .csv).
2. Decide on whether you want to include row names and column names.
3. Write your object using `write.csv`, customizing output using the `row.names=FALSE` argument to exclude row names.

```
> write.csv(com,'data/mymatrix',row.names=FALSE)
```

INTERMISSION

Summary Statistics

How do you calculate summary statistics from vectors?

```
> x = 1:3
> y = 2:4
```

```
> length(x)
[1] 3
> mean(x)
[1] 2
> sqrt(x)
[1] 1.000000 1.414214 1.732051
```

Summary Statistics

How do you calculate summary statistics from vectors?

```
> x = 1:3
> y = 2:4
```

```
> sd(x)
[1] 1
> var(x)
[1] 1
> cor(x, y)
[1] 1
```

Summary Statistics

How do you calculate summary statistics from matrices, such as the mean for all observations (i.e. for each row)?

```
> M
```

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
> apply(M, MARGIN = 1, FUN = mean)
```

```
[1] 4 5 6
```

Summary Statistics

How about for all columns?

```
> M
```

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
> apply(M, MARGIN = 2, FUN = mean)
```

```
[1] 2 5 8
```

Summary Statistics

What if I want the mean for a given variable (y) for each level of a factor (x)?

```
> y = com$V1
> x = com$env
> tapply(y, INDEX = x, fun = mean)
```

```
 [1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
```

Summary Statistics

What about getting the standard deviation for `y` given the levels of `x`?

```
> y = com$V1
> x = com$env
```

Summary Statistics

What about getting the standard deviation for y given the levels of x?

```
> y = com$V1
> x = com$env
```

```
> tapply(y, INDEX = x, fun = s
 [1] 1 1 1 1 1 1 1 1 1 1 2 2 2
```

Summary Statistics

How about getting the sum of all of the values in the matrices in a list?

```
> AM.list = list(A, M)
```

Summary Statistics

How about getting the sum of all of the values in the matrices in a list?

```
> AM.list = list(A, M)
```

```
> lapply(AM.list, FUN = sum)

[[1]]
[1] 45

[[2]]
[1] 45
```

t-test

How do you conduct one- and two-sample t-tests?

```
> x = com$V1
> y = com$V2
```

```
> t.test(x)
```

t-test (one-sample)

```
> t.test(x)

        One Sample t-test

data:  x
t = 4.1358, df = 19, p-value = 0.0005619
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
  52.89918 161.30082
sample estimates:
mean of x
    107.1
```

t-test (one-sample)

```
> t.test(x, alternative = "two.sided")

        One Sample t-test

data:  x
t = 4.1358, df = 19, p-value = 0.0005619
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
  52.89918 161.30082
sample estimates:
mean of x
    107.1
```

t-test (one-sample)

```
> t.test(x, alternative = "greater")

        One Sample t-test

data:  x
t = 4.1358, df = 19, p-value = 0.0002810
alternative hypothesis: true mean is greater than 0
95 percent confidence interval:
 62.32249      Inf
sample estimates:
mean of x
    107.1
```

t-test (one-sample)

```
> t.test(x, alternative = "less")

        One Sample t-test

data:  x
t = 4.1358, df = 19, p-value = 0.9997
alternative hypothesis: true mean is less than 0
95 percent confidence interval:
     -Inf 151.8775
sample estimates:
mean of x
    107.1
```

t-test (two-sample)

```
> t.test(x, y, alternative = "less")

        Welch Two Sample t-test

data:  x and y
t = -0.0588, df = 33.738, p-value = 0.4767
alternative hypothesis: true difference in means is less th
95 percent confidence interval:
     -Inf 51.35174
sample estimates:
mean of x mean of y
   107.10    108.95
```

Regression

How do you conduct a regression?

```
> x = com$V1
> y = com$V2
```

```
> xy.fit = lm(y ~ x)
> summary(xy.fit)
```

Regression

```
> xy.fit = lm(y ~ x)
> summary(xy.fit)

Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-100.81  -51.17  -13.80   25.17  157.08

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  84.8026    23.9013   3.548   0.0023 **
x             0.2255     0.1536   1.468   0.1594
---
```

ANOVA

How do you conduct an ANOVA?

```
> x = factor(com$env)
> y = com$V2
```

```
> anova.fit = aov(y ~ x)
> summary(anova.fit)
```

ANOVA

```
> anova.fit = aov(y ~ x)
> summary(anova.fit)

            Df Sum Sq Mean Sq F value    Pr(>F)
x            1  69502   69502  24.208 0.0001104 ***
Residuals   18  51679    2871
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.0
```

ANOVA

```
> unlist(anova.fit)
$`coefficients.(Intercept)`
[1] 167.9

$coefficients.x2
[1] -117.9

$residuals.1
[1] 11.1

$residuals.2
[1] -24.9

$residuals.3
```

ANOVA

```
> names(anova.fit)
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.re
 [9] "contrasts"     "xlevels"       "call"          "terms
[13] "model"
```

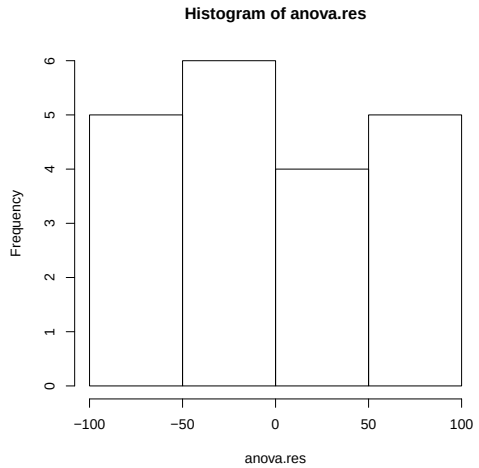
ANOVA

```
> anova.fit$residuals
    1     2     3     4     5     6     7     8     9    10
 11.1 -24.9  82.1  82.1 -60.9 -60.9 -24.9  82.1 -96.9  11.1
   14    15    16    17    18    19    20
-50.0  57.0 -50.0 -14.0 -14.0  57.0  21.0
```

Histogram

How would you plot a histogram of the residuals?

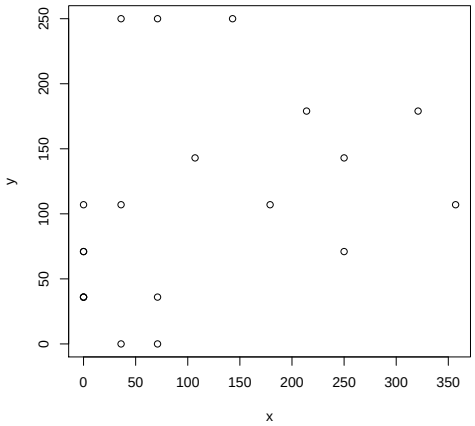
```
> anova.res = anova.fit$residuals
> hist(anova.res)
```



Scatterplot

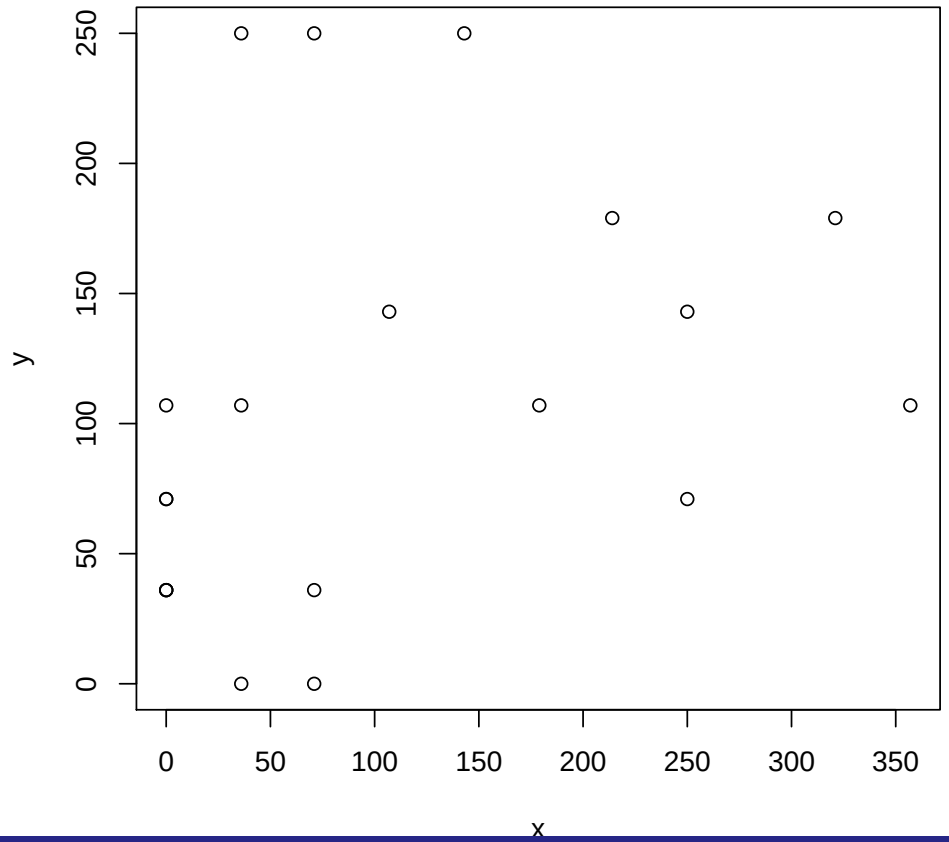
How would you make an x-y scatterplot?

```
> x = com$V1
> y = com$V2
> plot(x, y)
```



Scatterplot

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



Barplots

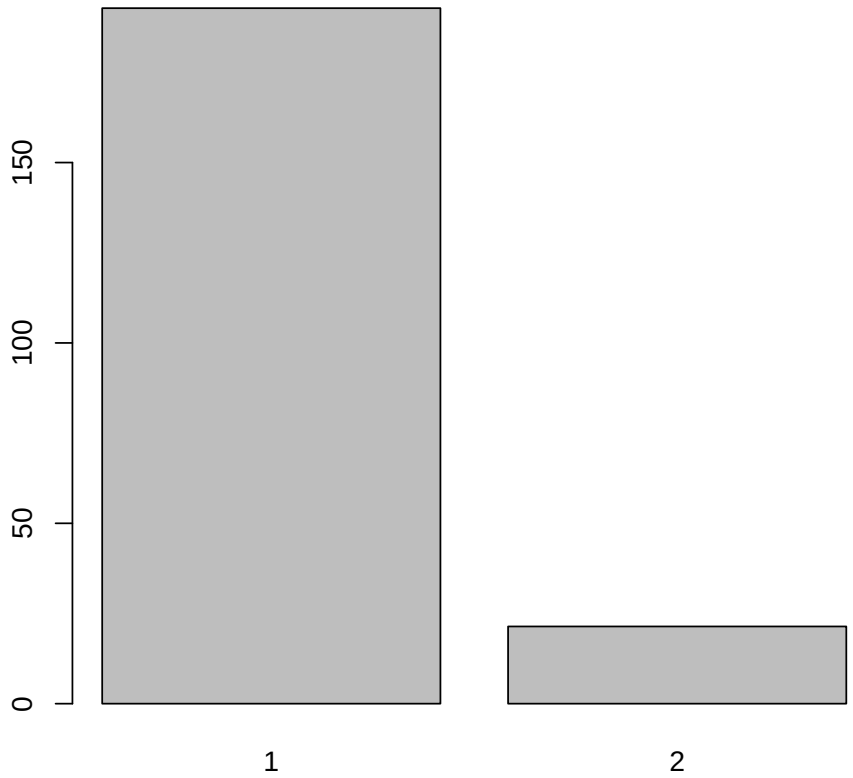
How would you make a barplot of the means of `V1` at each level of `env` in the `com` dataset?

Barplots

How would you make a barplot of the means of `V1` at each level of `env` in the `com` dataset?

```
> x = com$env
> y = com$V1
> mu = tapply(y, INDEX = x, fun = mean)
> barplot(mu)
```

Barplots



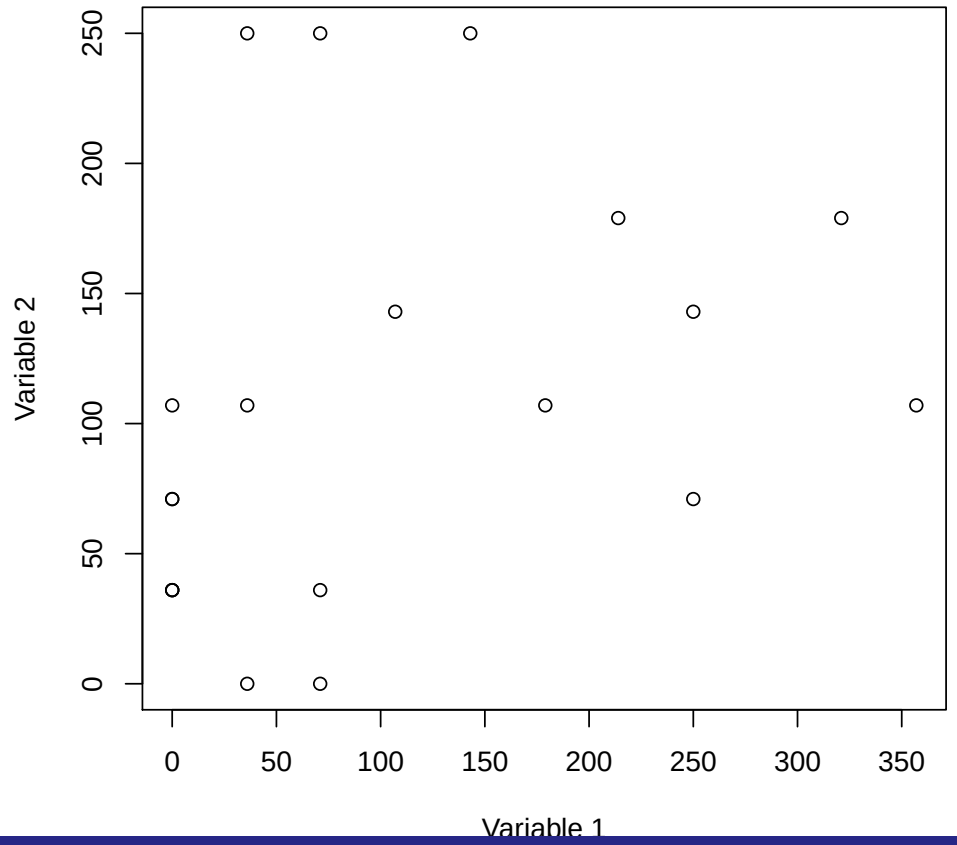
Customizing Plots

How do you change the axis names?

```
> x = com$V1
> y = com$V2
> plot(x, y)
> plot(x, y, xlab = "Variable 1", ylab = "Variable 2")
```

Customizing Plots

```
> plot(x, y, xlab = "Variable 1", ylab = "Variable 2")
```



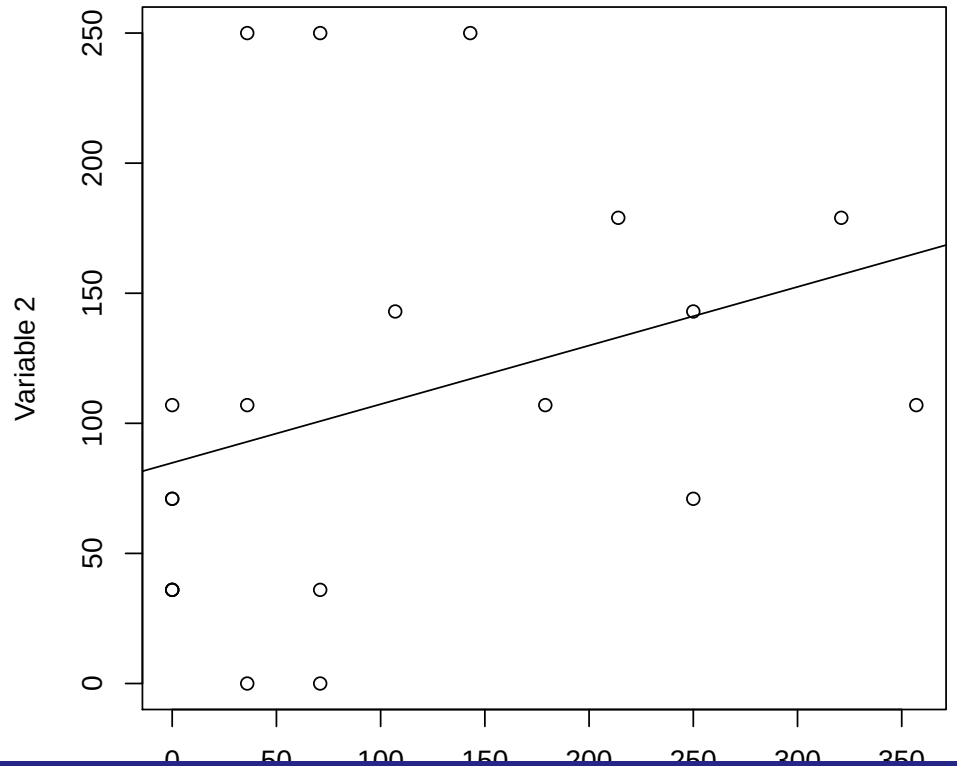
Customizing Plots

How do you add a regression line?

```
> x = com$V1
> y = com$V2
> plot(x, y, xlab = "Variable 1", ylab = "Variable 2")
```

Customizing Plots

```
> plot(x, y, xlab = "Variable 1", ylab = "Variable 2")
> abline(lm(y ~ x))
```



Multi-plot Windows

1. Setup the plot window using the `par` function.
2. Change `par(mfrow=c(1,2))`.
3. Build each plot in succession.

Locator - In case you get lost in a plot...

1. Build your plot.
2. Run `locator(1)` in the command line.
3. Click on the plot, **R** will output the coordinates.

Packages: Installination and Use

1. Find the name of the package you want.
2. Make sure you're connected to the internet.
3. Use the command, `install.packages('package name')`
4. Use the command, `library('package name')`
5. If there are conflicting functions in two packages, detach the one you're not using with the command, `detach(package:'package name')`

Packages: Installination and Use

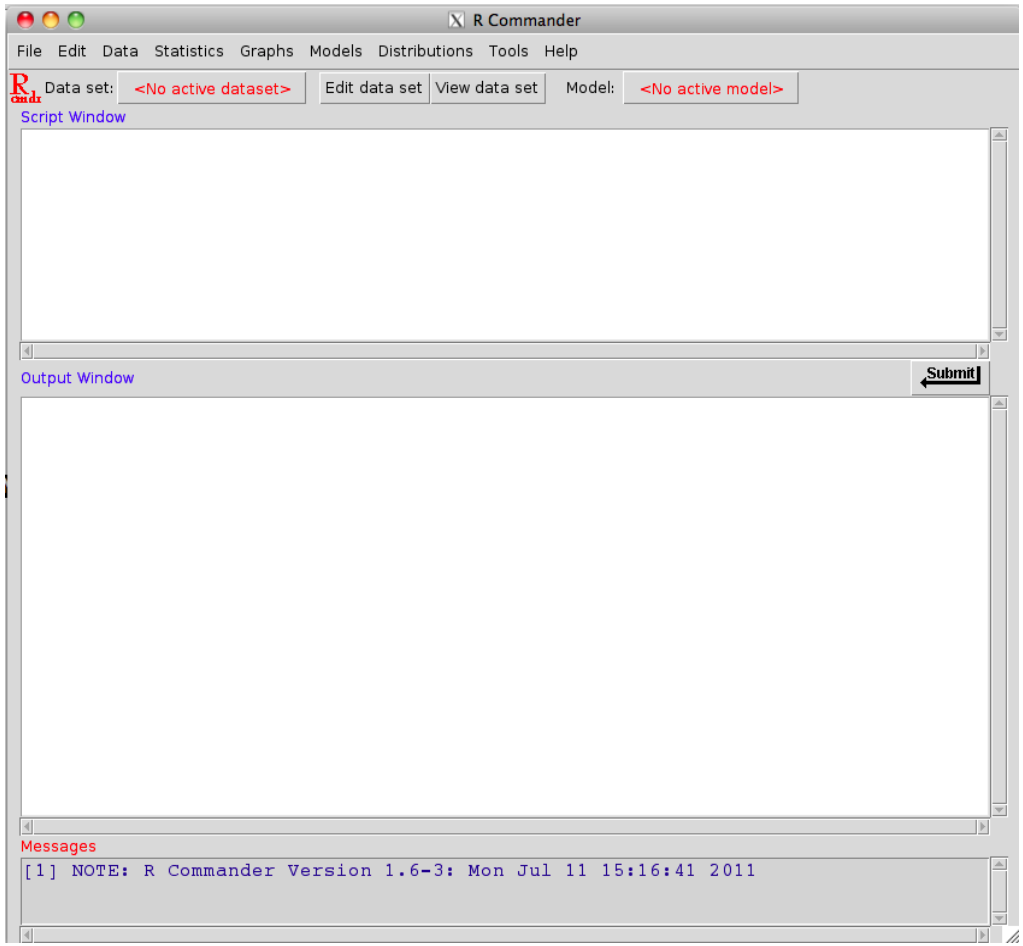
Ummm, everyone and their mother uses Excel, can R?

```
> install.packages('gdata')
> library("gdata")
> read.xls("data/CommData.xls")
```

R Commander

1. **R**'s version of *JMP* (designed by John Fox)
2. Fully integrated script editor.
3. `install.packages('Rcmdr')`
4. `library('Rcmdr')`
5. Documentation can be found here:
   http://socserv.mcmaster.ca/jfox/Misc/Rcmdr/

R Commander


R Commander
File Edit Data Statistics Graphs Models Distributions Tools Help
Data set: <No active dataset>
Edit data set
View data set
Model: <No active model>
Script Window
Output Window
Submit
Messages
[1] NOTE: R Commander Version 1.6-3: Mon Jul 11 15:16:41 2011

Resources

- *Cheat Sheet*: http://cran.r-project.org/doc/contrib/Short-refcard.pdf
- *Scripting*: http://google-styleguide.googlecode.com/svn/trunk/google-r-style.html
- *SimpleR*: http://www.calvin.edu/~stob/courses/m241/S11/Verzani-SimpleR.pdf
- *Quick-R*: http://www.statmethods.net/index.html
- *Plotting*: http://www.stat.auckland.ac.nz/~paul/RGraphics/rgraphics.html
- *Regression and ANOVA*: http://cran.r-project.org/doc/contrib/Faraway-PRA.pdf
- *Ecological Analyses*:
  - http://ecology.msu.montana.edu/labdsv/R/
  - http://www.mpcer.nau.edu/igert/eco_analysis_r.html
- *Network Analyses*: http://erzuli.ss.uci.edu/R.stuff/

Writing Functions

1. Create functions as you would objects using `function`.
2. The fundamental design is:
   `my.func=function(x,...)'insert things you want done'`
3. Arguments and objects created within the function are not passed out of the function.

Looping

1. Used to repeat a task (can be very powerful, but computationally inefficient).
2. Takes arguments that determine the number of repititions.
3. Uses the `for` or `while` commands.