Just a simple analysis

Gregory S Gilbert ENVS291R
11/4/2016

Let's do a simple linear regression

- Create, copy, or download a data file in Excel
- Save as a .csv data file
- Set the working directory to where the file is
- Read the data into a data frame
- Make a quick plot of the data
- Run a regression
- Interpret the statistics
- Plot the data with a regression line
- Save your work

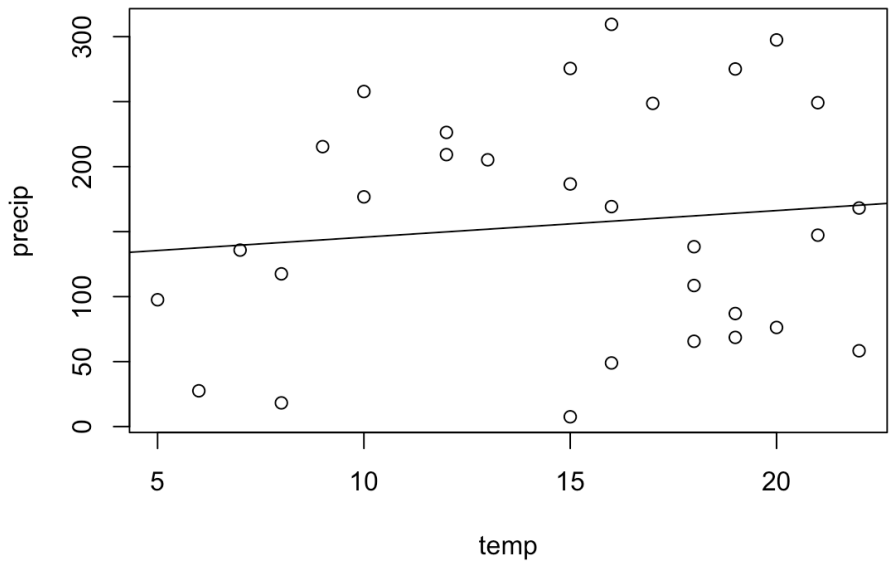
This is where we are going

```
a<-read.csv("/Users/greg/Desktop/classdata/RegressionDataset.csv")
summary(lmaout<-lm(precip~temp,data=a))


Call:
lm(formula = precip ~ temp, data = a)

Residuals:
    Min      1Q  Median      3Q     Max
-148.43  -86.78   -2.99   75.30  151.42

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  125.227     51.396   2.437   0.0214 *
temp           2.051      3.270   0.627   0.5356
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 89.7 on 28 degrees of freedom
Multiple R-squared:  0.01385,   Adjusted R-squared:  -0.02137
F-statistic: 0.3934 on 1 and 28 DF,  p-value: 0.5356
```

This is where we are going

Scatterplot with regression line overlay

```
plot(precip~temp,data=a); abline(lmaout)
```

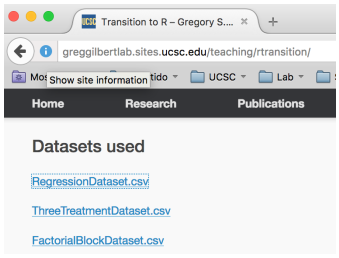


4/29

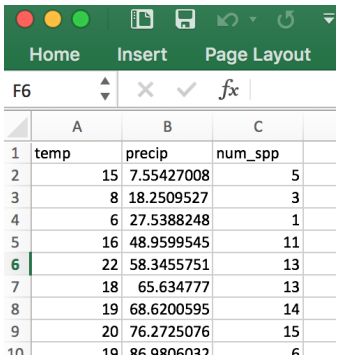
Get some data ready

http://greggilbertlab.sites.ucsc.edu/teaching/rtransition/

Control/Right-click to download data; open in Excel



File Save As… Comma Separated Values (.csv)
Place in folder where you want to work

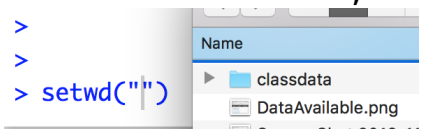


Set the working directory

- **By menu**
  Session/Set Working Directory/Choose Directory
- **Through R code**

```
setwd("/Users/greg/Desktop/classdata")
```

- **Drag-and-drop R code**
  Type setwd("") in the Console window
  Then from Finder, drag folder to between the quotes



Name with path is automatically copied

Check your working directory

```
getwd() #get working directory function
```

```
[1] "/Users/greg/Dropbox/classes/ENVS291Rtransition/IntroToR_Fall2016
```

Because `getwd` is a function it MUST be followed by (), even if there is no object in it.

```
dir("/Users/greg/Desktop/classdata") # what files are in directory
```

```
[1] "lmaoutplot.pdf"        "RegressionDataset.csv"
```

You can nest functions.
What if you did with `dir(getwd())`?

Using a working directory

```
# define the full path of the file
a<-read.csv("/Users/greg/Desktop/classdata/RegressionDataset.csv")
# "~" represents common part of path
a<-read.csv("~/Desktop/classdata/RegressionDataset.csv")
#Set the working directory first
setwd("/Users/greg/Desktop/classdata")
#read in the file straight from the working directory
a<-read.csv("RegressionDataset.csv")
```

Now that this is your working directory you only need to use the name of a file within that directory.

Check on the data you imported

```
str(a) # check out the structure of a
```

```
'data.frame':   30 obs. of  3 variables:
 $ temp   : int  15 8 6 16 22 18 19 20 19 5 ...
 $ precip : num  7.55 18.25 27.54 48.96 58.35 ...
 $ num_spp: int  5 3 1 11 13 13 14 15 6 19 ...
```

```
head(a,3) #look at the first 3 lines of a
```

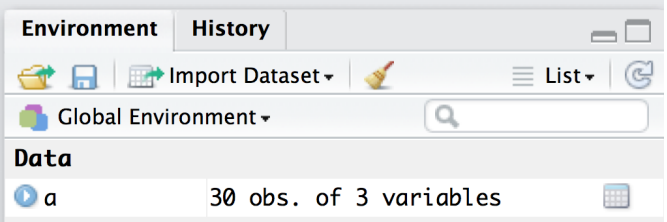
```
  temp   precip num_spp
1   15  7.55427       5
2    8 18.25095       3
3    6 27.53882       1
```

```
tail(a,2) #look at last 2 lines of a
```

```
   temp   precip num_spp
29   20 297.4792      38
30   16 309.4613      30
```

RStudio Check on the data you imported

RStudio Environment panel for info on objects



Click on the blue arrow for str(a)

Click on the spreadsheet symbol for head(a)

Let's explore the data a little

```
summary(a$temp) #basic moments of one variable
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    5.0    10.5    16.0    14.9    19.0    22.0 
```

```
mean(a$precip) #mean of a variable
```

```
[1] 155.7819
```

```
length(a$num_spp) #number of observations of a variable
```

```
[1] 30
```

```
dim(a) #dimensions of the data frame
```

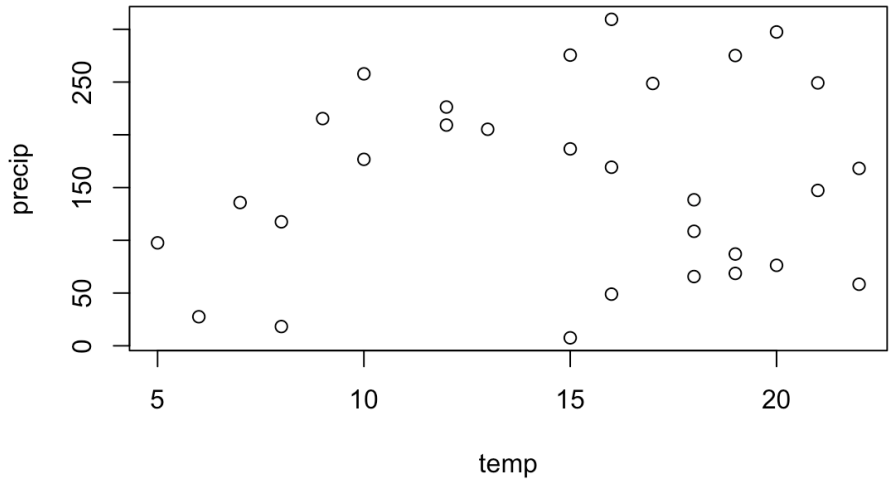
```
[1] 30  3
```

What do you get with these?

```
summary(a)
length(a)
mean(a)
dim(a$precip)
```

How are precipitation and temperature related?

```
plot(precip~temp,data=a)
```



13/29

Linear regression

Use the `lm()` function (linear model)

```
lm(precip~temp,data=a)
```

```
Call:
lm(formula = precip ~ temp, data = a)

Coefficients:
(Intercept)         temp
    125.227        2.051
```

Doesn't tell you much, huh?

Linear regression - assign to object

Assign the output of `lm()` to an object 'lmaout'

```
lmaout<-lm(precip~temp,data=a)
lmaout
```

```
Call:
lm(formula = precip ~ temp, data = a)

Coefficients:
(Intercept)         temp
    125.227        2.051
```

Still not much?

Linear regression - output is a list

```
str(lmaout)
```

```
List of 12
 $ coefficients : Named num [1:2] 125.23 2.05
  ..- attr(*, "names")= chr [1:2] "(Intercept)" "temp"
 $ residuals    : Named num [1:30] -148 -123 -110 -109 -112 ...
  ..- attr(*, "names")= chr [1:30] "1" "2" "3" "4" ...
 $ effects      : Named num [1:30] -853.3 56.3 -61 -89.3 -109.8 ...
  ..- attr(*, "names")= chr [1:30] "(Intercept)" "temp" "" "" ...
 $ rank         : int 2
 $ fitted.values: Named num [1:30] 156 142 138 158 170 ...
  ..- attr(*, "names")= chr [1:30] "1" "2" "3" "4" ...
 $ assign       : int [1:2] 0 1
 $ qr           :List of 5
  ..$ qr   : num [1:30, 1:2] -5.477 0.183 0.183 0.183 0.183 ...
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : chr [1:30] "1" "2" "3" "4" ...
  .. .. ..$ : chr [1:2] "(Intercept)" "temp"
  .. ..- attr(*, "assign")= int [1:2] 0 1
  ..$ qraux: num [1:2] 1.18 1.25
  ..$ pivot: int [1:2] 1 2
  ..$ tol  : num 1e-07
  ..$ rank : int 2
  ..- attr(*, "class")= chr "qr"
 $ df.residual  : int 28
 $ xlevels      : Named list()
 $ call         : language lm(formula = precip ~ temp, data = a)
 $ terms        :Classes 'terms', 'formula'  language precip ~ temp
  .. ..- attr(*, "variables")= language list(precip, temp)
  .. ..- attr(*, "factors")= int [1:2, 1] 0 1
  .. .. ..- attr(*, "dimnames")=List of 2
  .. .. .. ..$ : chr [1:2] "precip" "temp"
  .. .. .. ..$ : chr "temp"
       - attr(*, "term.labels")= chr "temp"
```

Linear regression - extractor functions

```
summary(lmaout)


Call:
lm(formula = precip ~ temp, data = a)

Residuals:
    Min      1Q  Median      3Q     Max
-148.43  -86.78   -2.99   75.30  151.42

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  125.227     51.396   2.437   0.0214 *
temp           2.051      3.270   0.627   0.5356
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 89.7 on 28 degrees of freedom
Multiple R-squared:  0.01385,   Adjusted R-squared:  -0.02137
F-statistic: 0.3934 on 1 and 28 DF,  p-value: 0.5356
```

Linear regression - extractor functions

```
anova(lmaout)
```

```
Analysis of Variance Table

Response: precip
          Df Sum Sq Mean Sq F value Pr(>F)
temp       1   3165  3165.2  0.3934 0.5356
Residuals 28 225303  8046.5
```

Linear regression - break into the list!

```
lmaout$coefficients
```

```
(Intercept)        temp
 125.227425    2.050635
```

```
lmaout$coefficients[1] #intercept
```

```
(Intercept)
   125.2274
```

```
lmaout$coefficients[2] #slope
```

```
    temp
2.050635
```

Linear regression - break into the list!

```
lmaout$call
```

```
lm(formula = precip ~ temp, data = a)
```

```
lmaout$model$precip[1:5]
```

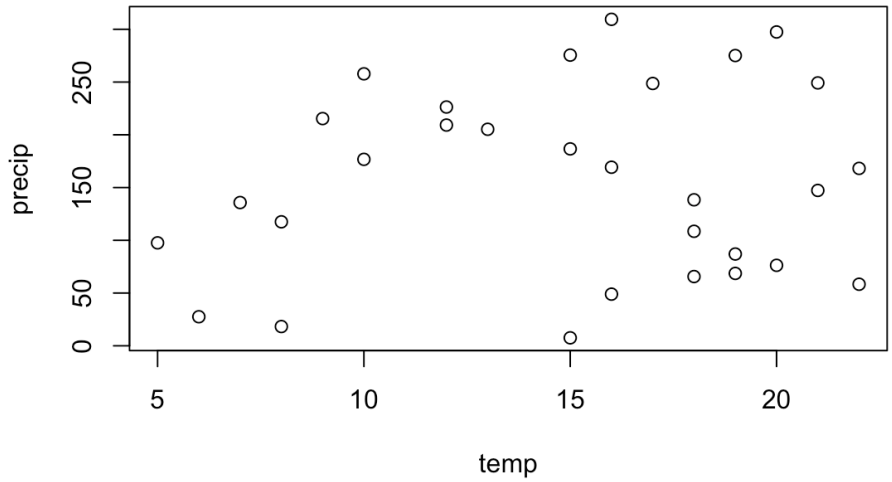
```
[1]  7.55427 18.25095 27.53882 48.95995 58.34558
```

```
lmaout$model$temp[1:5]
```

```
[1] 15  8  6 16 22
```

Linear regression - plot from the lm() object

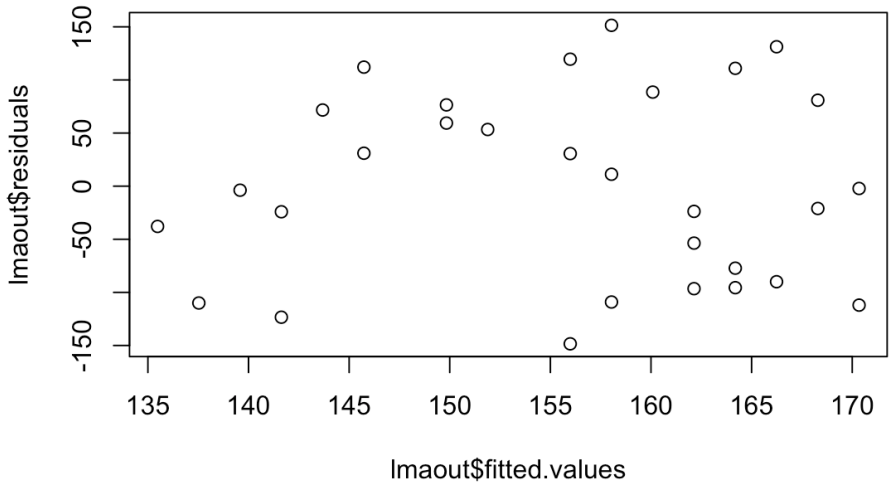
```
plot(precip~temp,data=lmaout$model)
```



21/29

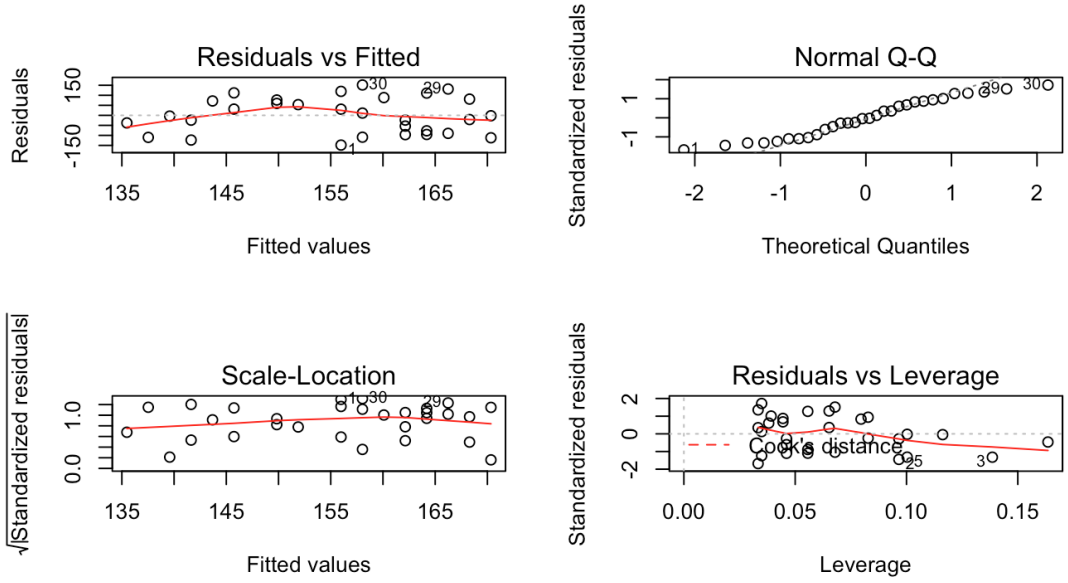
Linear regression - hand-made residual plot

```
plot(lmaout$residuals~lmaout$fitted.values)
```



Linear regression - but R makes it easy

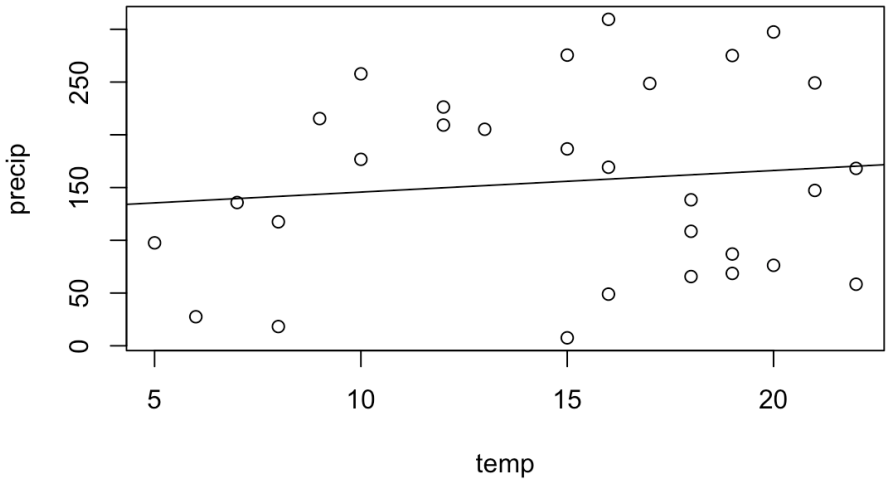
```
par(mfrow=c(2,2)) #puts graphs on a 2x2 grid
plot(lmaout) #spits out 4 diagnostic plots
```




```
par(mfrow=c(1,1))
```

Linear regression - plot overlay

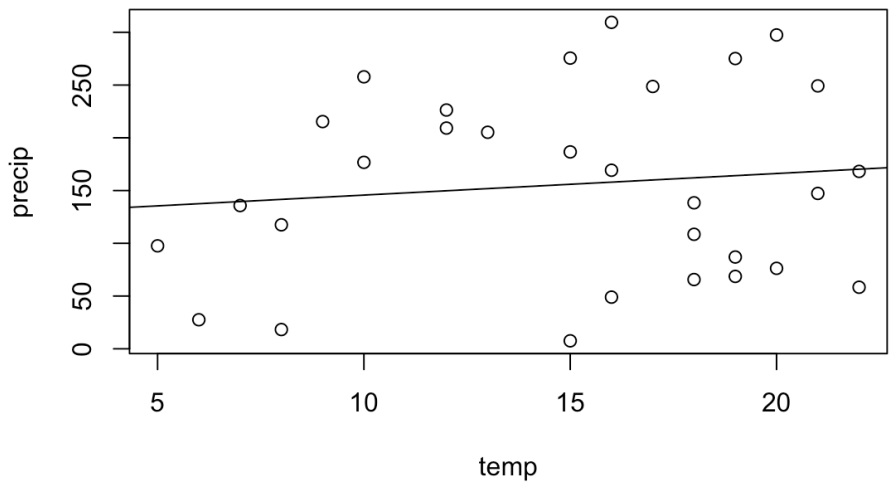
```
plot(precip~temp,data=a) #plot of raw data
abline(lmaout) #overlay
```



24/29

Linear regression - plot overlay direct!

```
plot(precip~temp,data=a) #plot of raw data
abline(lm(precip~temp,data=a)) #overlay regression
```



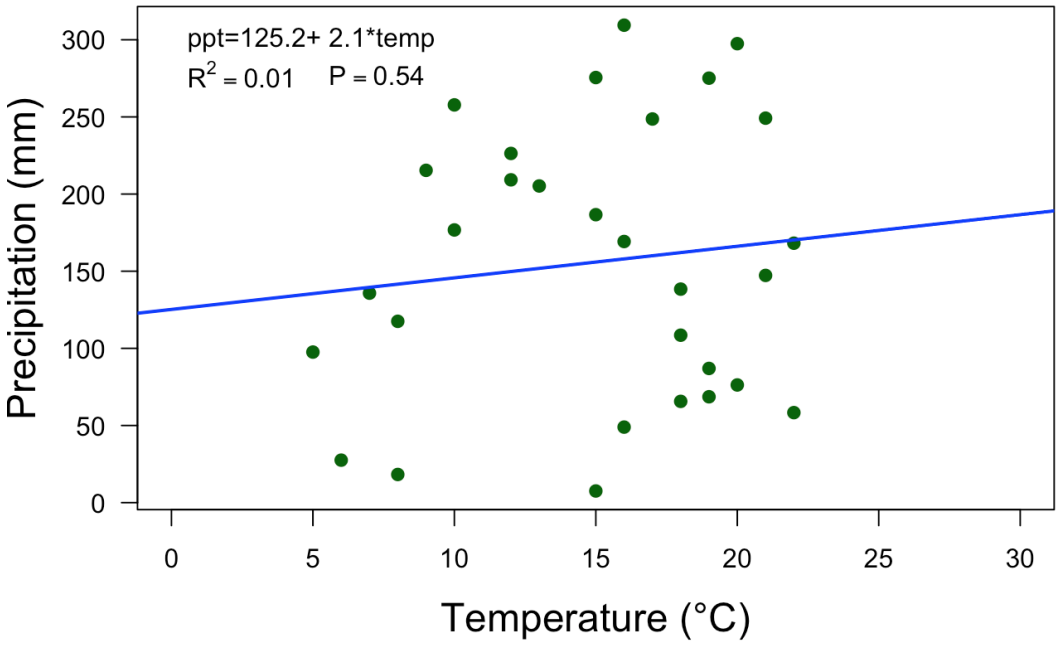
25/29

Linear regression - plot overlay

```
plot(precip~temp,data=a,xlim=c(0,30),
xlab="Temperature (°C)",ylab="Precipitation (mm)",
pch=19,col="darkgreen",cex.lab=1.5, las=1)

abline(lmaout,col="blue",lwd=2) #overlay regression

formula<-paste("ppt=",round(lmaout$coefficients[1],1),
"+ ",round(lmaout$coefficients[2],1),"*temp",sep="")

text(0,300,formula,pos=4)

text(0,275,bquote(R^2 ==.(round(summary(lmaout)$r.squared,2))),pos=4)

text(5,275,
bquote(P==.(round(summary(lmaout)$coefficients[2,4],2))),pos=4)
```

26/29

Linear regression - plot overlay



27/29

Save the plot

- RStudio Plots panel, Export
- Direct through R code

```
pdf(file="/Users/greg/Desktop/classdata/lmaoutplot.pdf")
plot(precip~temp,data=a)
abline(lmaout,col="blue",lwd=2)
dev.off()
```

```
## quartz_off_screen
##                 2
```

Where did it go?

Do I have to remember all this?

No. Get help.
- RStudio Help panel
- ?lm
- Stackoverflow.com

29/29