

Week 5 - General Linear Model - Regression 2

Andrew Stewart

Andrew.Stewart@manchester.ac.uk



@ajstewart_lang



<https://github.com/ajstewartlang>

Week	Topic
1	Introduction, Open Science, and Power
2	Introduction to R
3	Data Wrangling and Visualisation
4	General Linear Model - Regression
5	General Linear Model - Regression
6	No Timetabled Lecture - Reading Week
7	Consolidation Lab
8	General Linear Model - ANOVA
9	General Linear Model - ANOVA
10	Tidy Thursday Data Wrangling & Visualisation Challenge
11	Reproducing your Computational Environment using Binder
12	Dynamic, Reproducible Presentations Using xaringan

Semester 1 Assignments

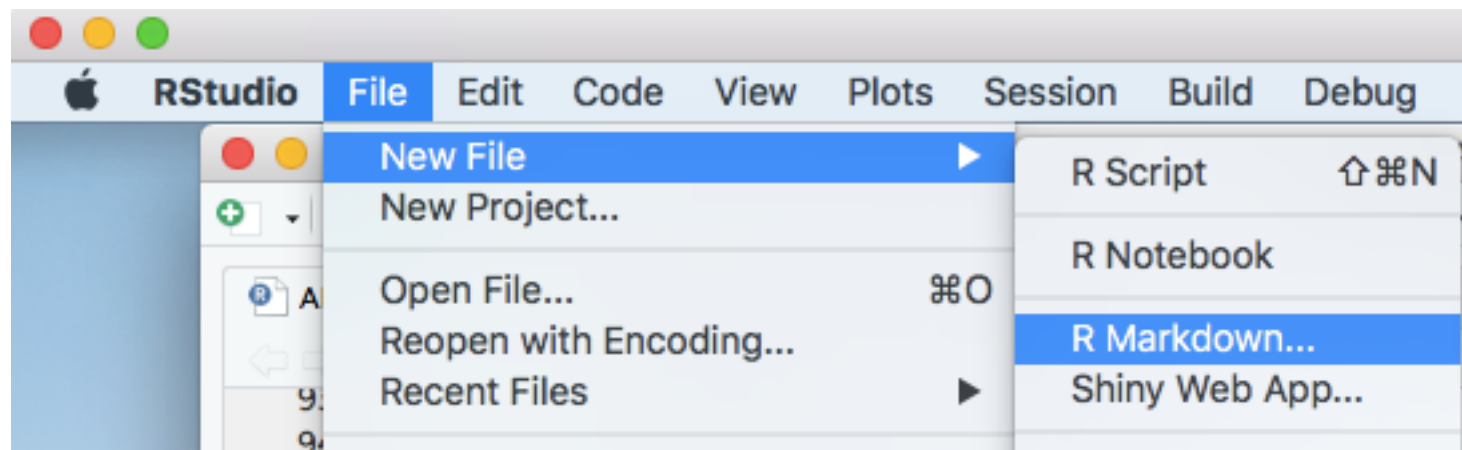
Data wrangling and visualisation – Due December 5th

ANOVA/ANCOVA – Due January 17th

R Markdown

- R Markdown is a language in and of itself that allows you to produce documents in many formats (e.g., .html, .pdf, .doc) that contain your R code, the output of that code, and narrative that you write describing what you are doing (and why).
- R Markdown documents can be produced from within RStudio.
- R Markdown cheat sheet is in the R documentation folder on Blackboard.

- First we need to create a new R Markdown file:



Type the title of your document here.

New R Markdown

Document
Presentation
Shiny
From Template

Title: ANOVA example

Author: Andrew Stewart

Default Output Format:

☒ HTML
Recommended format for authoring (you can switch to PDF or Word output anytime).

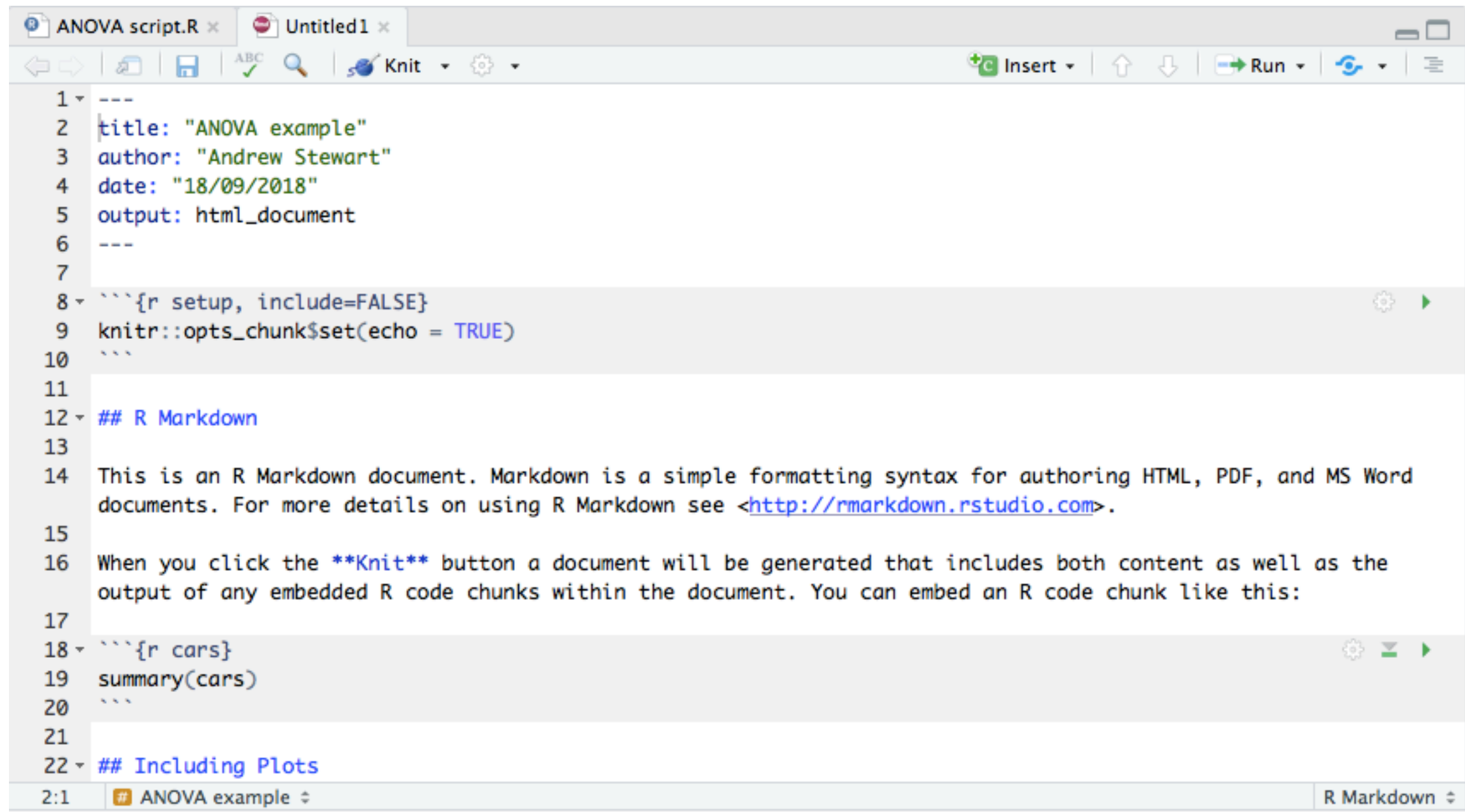
☐ PDF
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

☐ Word
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

OK Cancel

Important - replace your name here with your student ID number.


Select the kind of file you want to be generated by your Markdown - you can change this later btw.



The screenshot shows the RStudio interface with a file named 'ANOVA script.R' open. The editor displays an R Markdown document with the following content:

```
1 ---
2 title: "ANOVA example"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word
15 documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
16
17 When you click the Knit button a document will be generated that includes both content as well as the
18 output of any embedded R code chunks within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
```

The status bar at the bottom indicates the current line is 2:1 and the document type is 'R Markdown'.

You will now see a document like the above - it contains lots of example narrative (with a white background) and R code (with a grey background). We could actually 'knit' this document by clicking on  Knit to see what is produced...

What you will get is an html file (because that's the type of document we asked to be produced) that contains the R code, the associated output, plus the narrative. Notice how the ## symbols increasing the font size to allow us to generate headings in our narrative.

ANOVA example

Andrew Stewart

18/09/2018

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

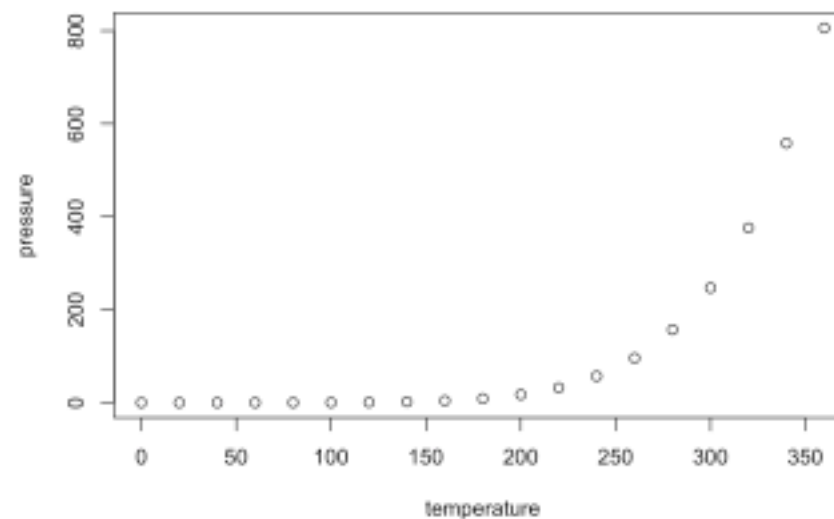
When you click the Knit button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed      dist
## Min.   : 4.0    Min.   :  2.00
## 1st Qu.:12.0    1st Qu.: 26.00
## Median :15.0    Median : 36.00
## Mean   :15.4    Mean   : 42.98
## 3rd Qu.:19.0    3rd Qu.: 56.00
## Max.   :25.0    Max.   :120.00
```

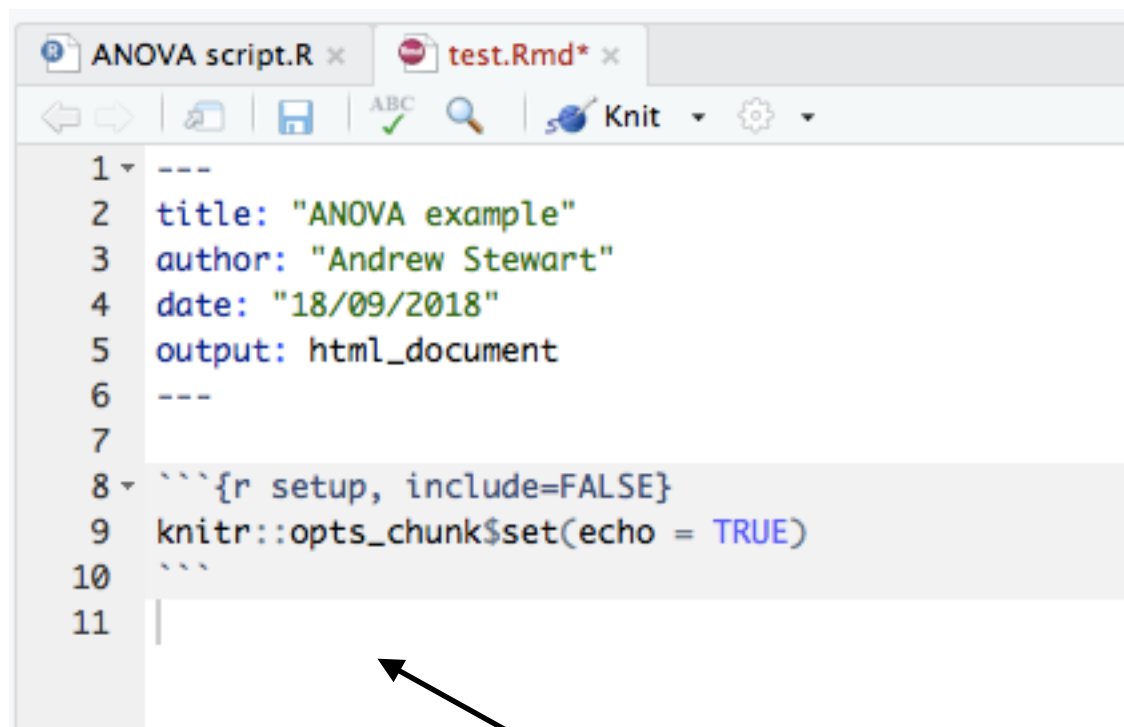
Including Plots

You can also embed plots, for example:



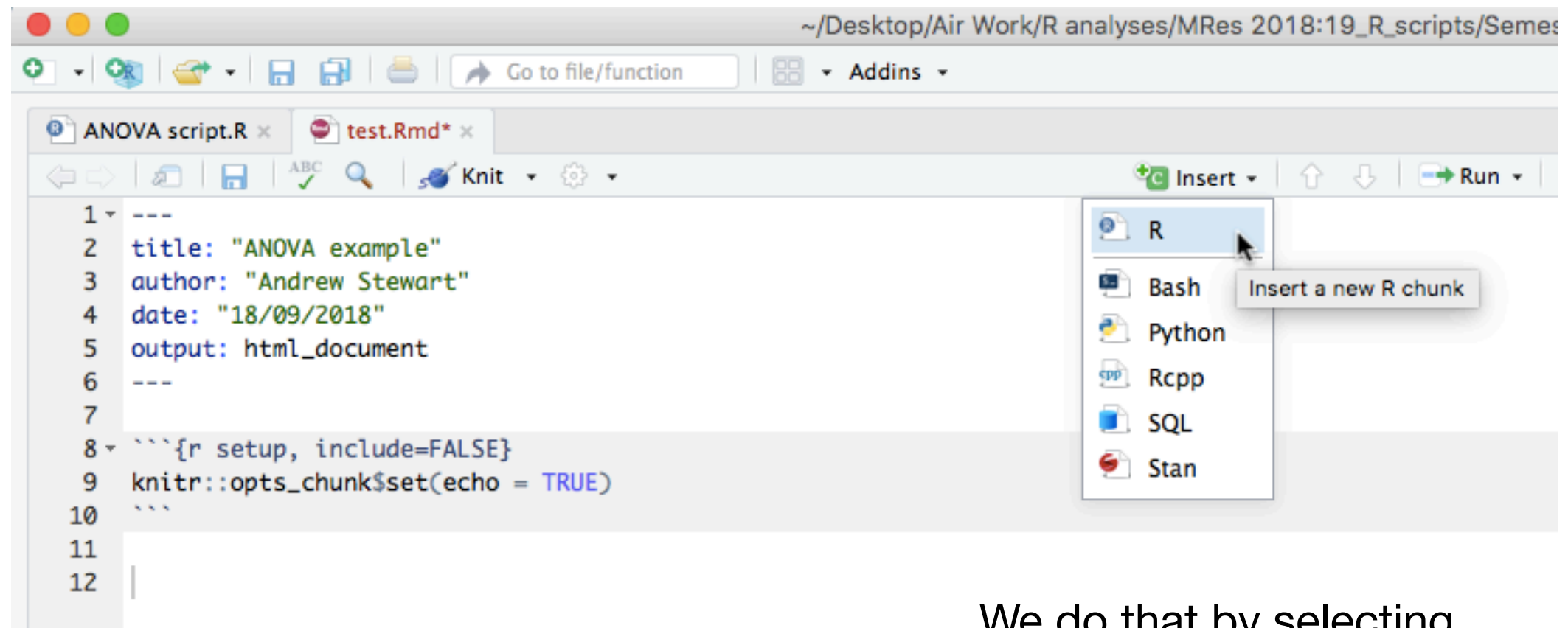
Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

When you come to create a document using R Markdown, you'll first want to delete a lot of the example code that appears when you first start a new Markdown file:

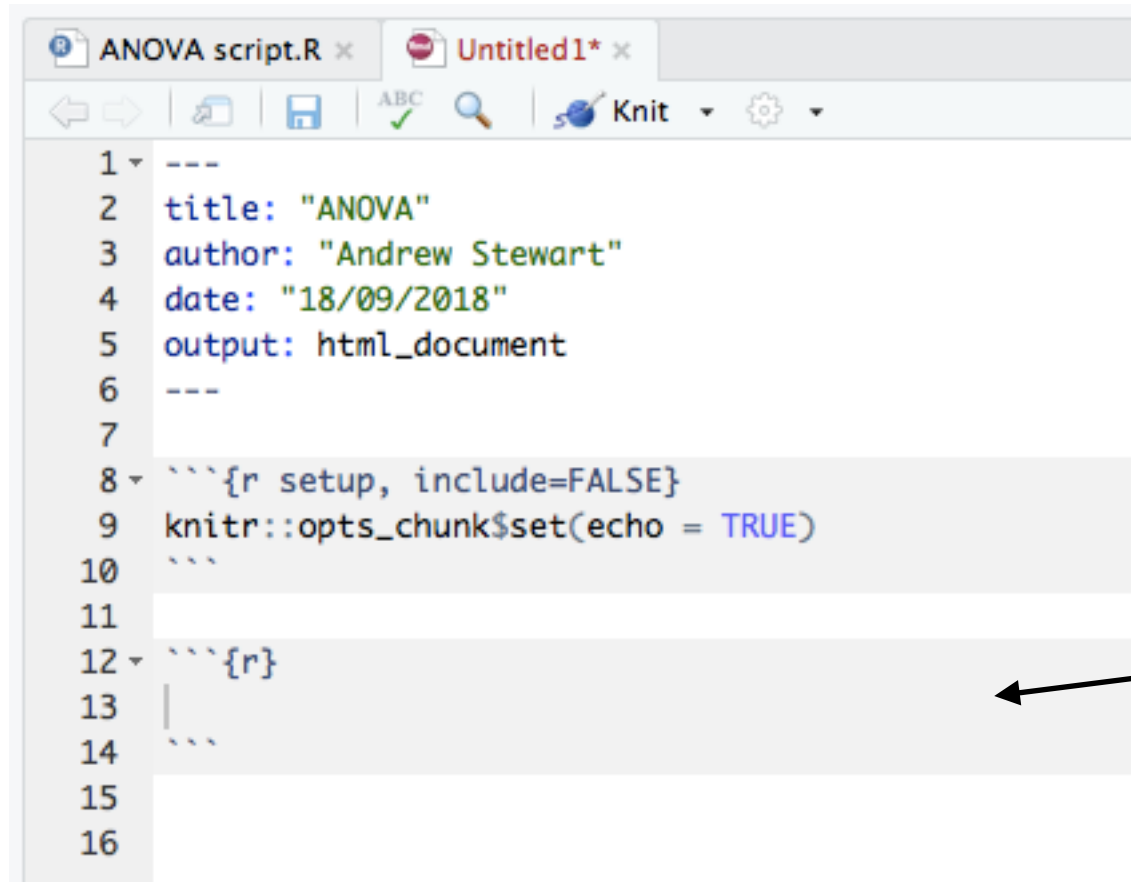


```
1 ---
2 title: "ANOVA example"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11 |
```

We now want to paste in chunks of our R code here



We do that by selecting “Insert” and then “R” - this will allow us to add some new R code.



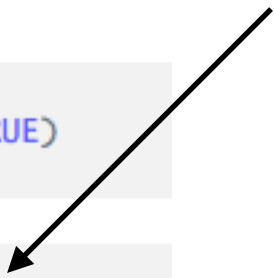
We can now insert our new R code here.

```

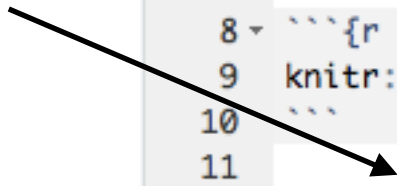
1 ---
2 title: "ANOVA"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ```{r}
13 library(tidyverse)
14 ```
15
16

```

We load the tidyverse packages first.



But maybe we should precede that with some narrative (and a heading) explaining what we're doing.



```

1 ---
2 title: "ANOVA"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## This is my first R Markdown document
13
14 First we load the tidyverse.
15 ```{r}
16 library(tidyverse)
17 ```
18

```

What happens if we now knit this short document?

ANOVA

Andrew Stewart

18/09/2018

This is my first R Markdown document

First we load the tidyverse.

```
library(tidyverse)
```


```
## -- Attaching packages ---- tidyverse 1.2.1 --  
--
```

```
## ✓ ggplot2 3.0.0      ✓ purrr  0.2.5  
## ✓ tibble  1.4.2      ✓ dplyr  0.7.6  
## ✓ tidyr   0.8.1      ✓ stringr 1.3.1  
## ✓ readr   1.1.1      ✓ forcats 0.3.0
```

```
## -- Conflicts ---- tidyverse_conflicts() --  
--  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag()     masks stats::lag()
```

Hmm, so we get some messages and warnings in our document - how can we get rid of these?

```
14 First we load the tidyverse.  
15 ```{r, message=FALSE}  
16 library(tidyverse)  
17 ```  
18
```



We can set the option for this chunk of R code not to display any messages by setting `messages=FALSE` within the first curly bracket. There are lots of other options available to use (e.g., `warnings=FALSE`).

Everything you need (and probably lots you don't) can be found in this cheatsheet here:

<http://www.rstudio.com/wp-content/uploads/2016/03/rmarkdown-cheatsheet-2.0.pdf>

and here is the 'R Markdown: The Definitive Guide':

<https://bookdown.org/yihui/rmarkdown/>

I've even put together a video tutorial talking you through how to make an R Markdown document...

<https://youtu.be/CBJjxS-UopA>

Assessments

- The assessments this semester and the mixed models one next semester both need to be produced using R Markdown.
- Top tips:

Have lots of narrative explaining what you're doing (and why).

Use small chunks of code bookended by narrative - the ideal code chunk length is where you can have one comment that describes what that bit of code does.

If you find you're commenting about two things that one chunk of R code does, that chunk should probably be split into two...

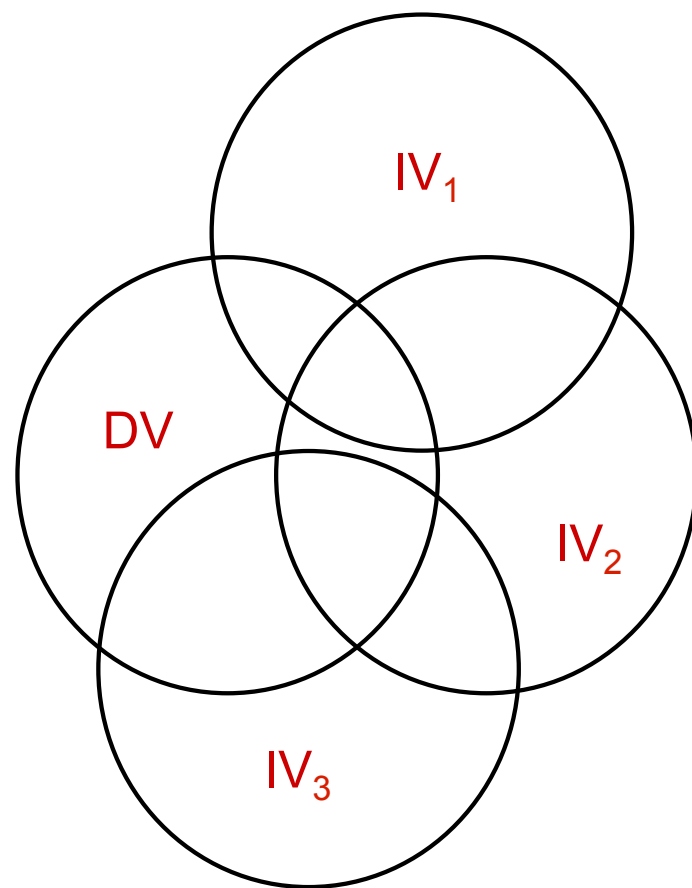
Last Week

- We went from variance to covariance to correlation to simple regression.
- We discussed how regression is really just prediction.
- We can build different models of data and test how good they are via measuring the extent to which they minimise the sum of squares of the difference between the observed data and our linear model.

Multiple Regression

- Multiple regression is just an extension of simple linear regression - but rather than having one predictor we have several.
- Our goal is the same though - to end up with an equation for a straight line that is the best fit to our data - in other words, we want to minimise the residual error.

Defining the Problem



To what extent do each of our IVs predict our DV (outcome)? Do we need to worry about our IVs being (too) correlated with each other?

Data Considerations

- Sample size (do we have enough power?)
- Multicollinearity and Singularity.
- Errors of prediction are normally distributed: Skewness, Nonlinearity, Homoscedasticity.
- The influence of outliers and influential cases.

How big should our N be?

- For testing the regression $N \geq 50 + 8k$ (where k = no. of predictors)
- For testing individual predictors $N \geq 104 + k$
- Usually we want to test both so calculate both equations and choose the one that produces the largest number.
- For example, with $k=6$ we require $N=98$ to test the regression and $N=110$ to test the individual predictors. We select 110.

Multicollinearity and Singularity

- Multicollinearity: when two or more variables are highly correlated (tested by examining VIF value for each variable).
- Singularity: redundancy (e.g., one of the variables is a combination of two or more of the other IVs).
- We can use collinearity diagnostics to see if we have a possible problem...

Assumptions: no multicollinearity among predictors

- VIF stands for Variance Inflation Factor. Essentially, it tells us about when we have to worry about (multi)collinearity. We can ask for VIF to any model in R by using the function `vif()` in the `car` package.
- So when do you worry?
- As a rule of thumb VIF greater than 10 suggests a multicollinearity issue (although greater than 5 has been suggested too - more conservative).

Assumptions: normal distribution of errors

- Errors of prediction are normally distributed around each every predicted (Y') score.
- That is, for every predicted score, the observed scores around that prediction should be normally distributed (i.e., normally distributed error).

Assumptions: Linearity

- The relationship to be modelled must be linear.
- That is, an increase in the scores of a predictor should be followed by a increase in the outcome and vice versa.
- There must be a uniform relationship between the fitted values and the residual error that can be captured by a straight line.
- Violation of this assumption is not dire – it weakens the power of the analysis to capture the relationship between the variables, but the analysis can proceed.

Assumptions: homoscedasticity

- Standard deviations of errors of prediction should be approximately equal for all predicted scores.
- That is, the amount of error is the same at different predicted values of the outcome.
- Violation is not terminal, but does weaken the analysis.
- Can be corrected by using weighted (generalised) least squares regression instead of ols regression.

Outliers

- Outliers are cases that do not fit well with the regression model (cases with large residuals).
- Like correlation, regression models are sensitive to outliers so it makes sense to consider removing/replacing them to improve the model.
- Bear in mind that even if you remove them they may be theoretically interesting.
- Influential cases can be detected easily via Cook's distance (part of one of our diagnostic plots).

The Linear Model in R

We are interested in whether house prices in 250 regions of the UK can be predicted by:

- (a) Population size
- (b) Crime rate (per 10,000 people)
- (c) Average age of people in the region
- (d) Average household income in the region.

Including our predictor and a column identifying our Regions, our datasets consists of 6 variables.

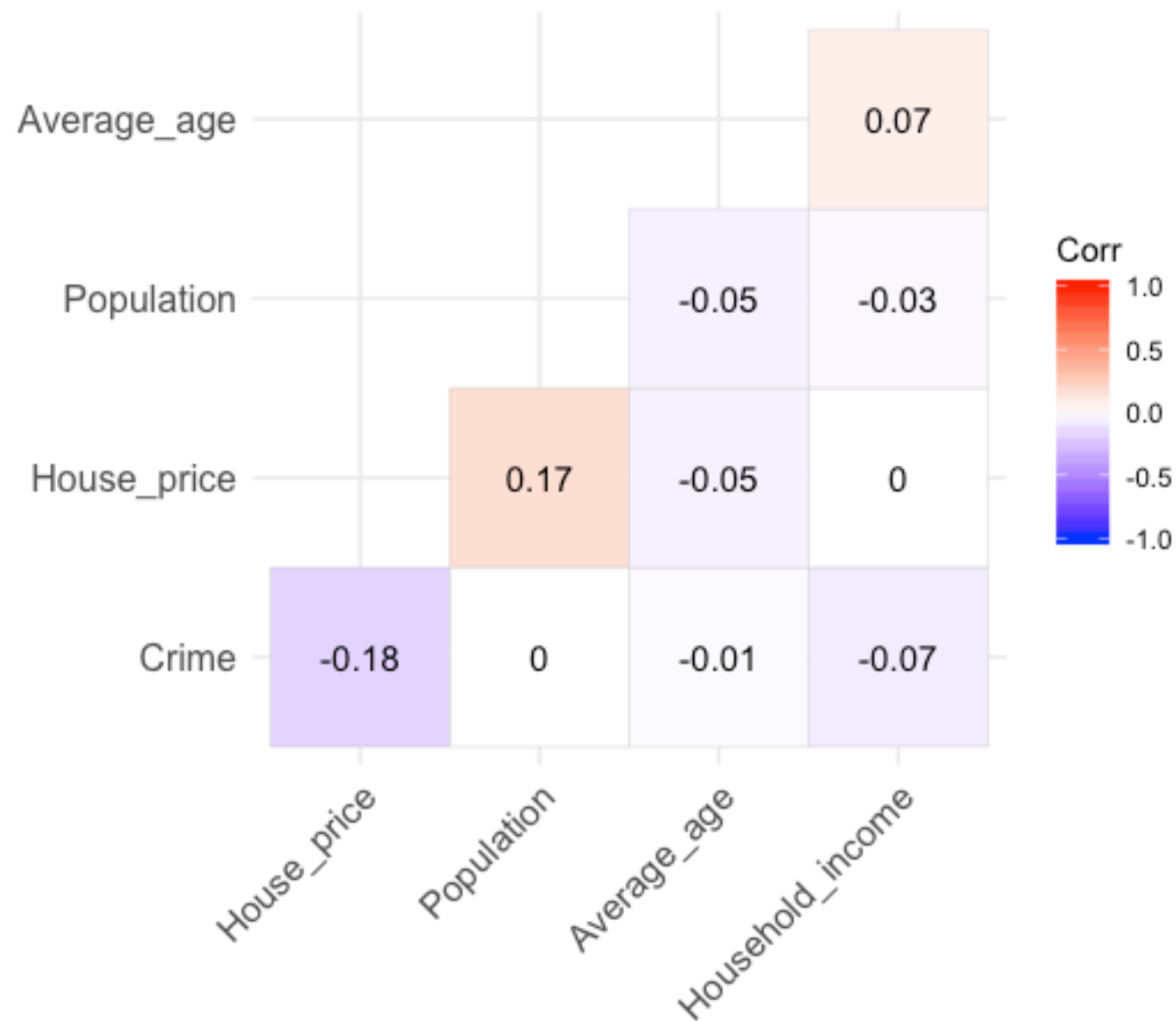
```
> str(my_data)
Classes 'tbl_df', 'tbl' and 'data.frame': 250 obs. of  6 variables:
 $ Region      : num  1 2 3 4 5 6 7 8 9 10 ...
 $ House_price : num  193735 201836 191643 215952 203295 ...
 $ Population  : num  49004 48307 50379 53664 45481 ...
 $ Crime       : num  14 25 19 17 22 32 21 21 20 25 ...
 $ Average_age : num  72.7 78.1 71.4 72.1 76.1 ...
 $ Household_income: num  20843 19130 20411 16863 19964 ...
```

```
> head(my_data)
# A tibble: 6 x 6
   Region House_price Population Crime Average_age Household_income
   <dbl>      <dbl>      <dbl> <dbl>      <dbl>      <dbl>
1     1     193735     49004     14      72.7     20843.
2     2     201836     48307     25      78.1     19130.
3     3     191643     50379     19      71.4     20411.
4     4     215952     53664     17      72.1     16863.
5     5     203295     45481     22      76.1     19964.
6     6     191795     51582     32      81.2     20207.
```

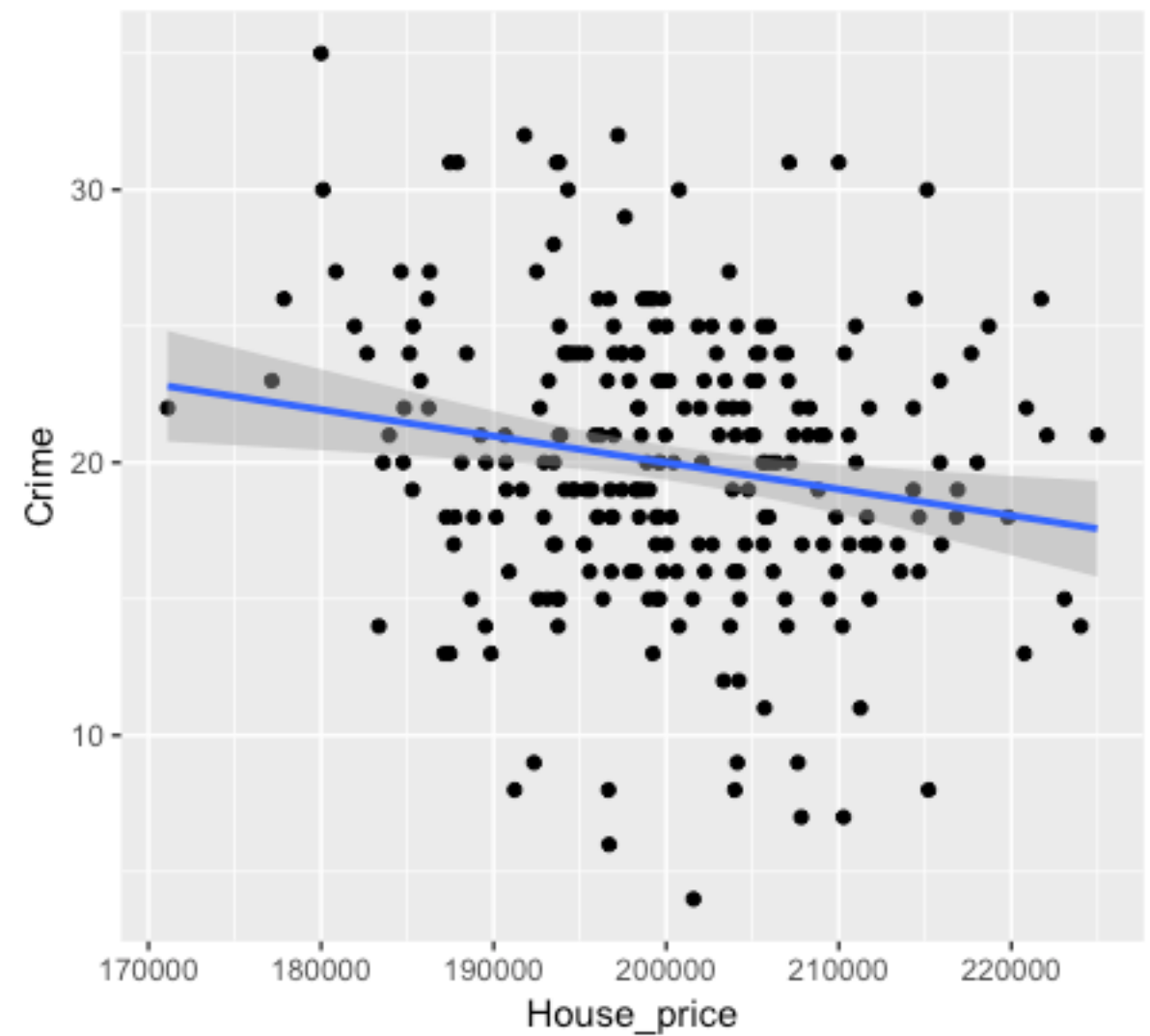
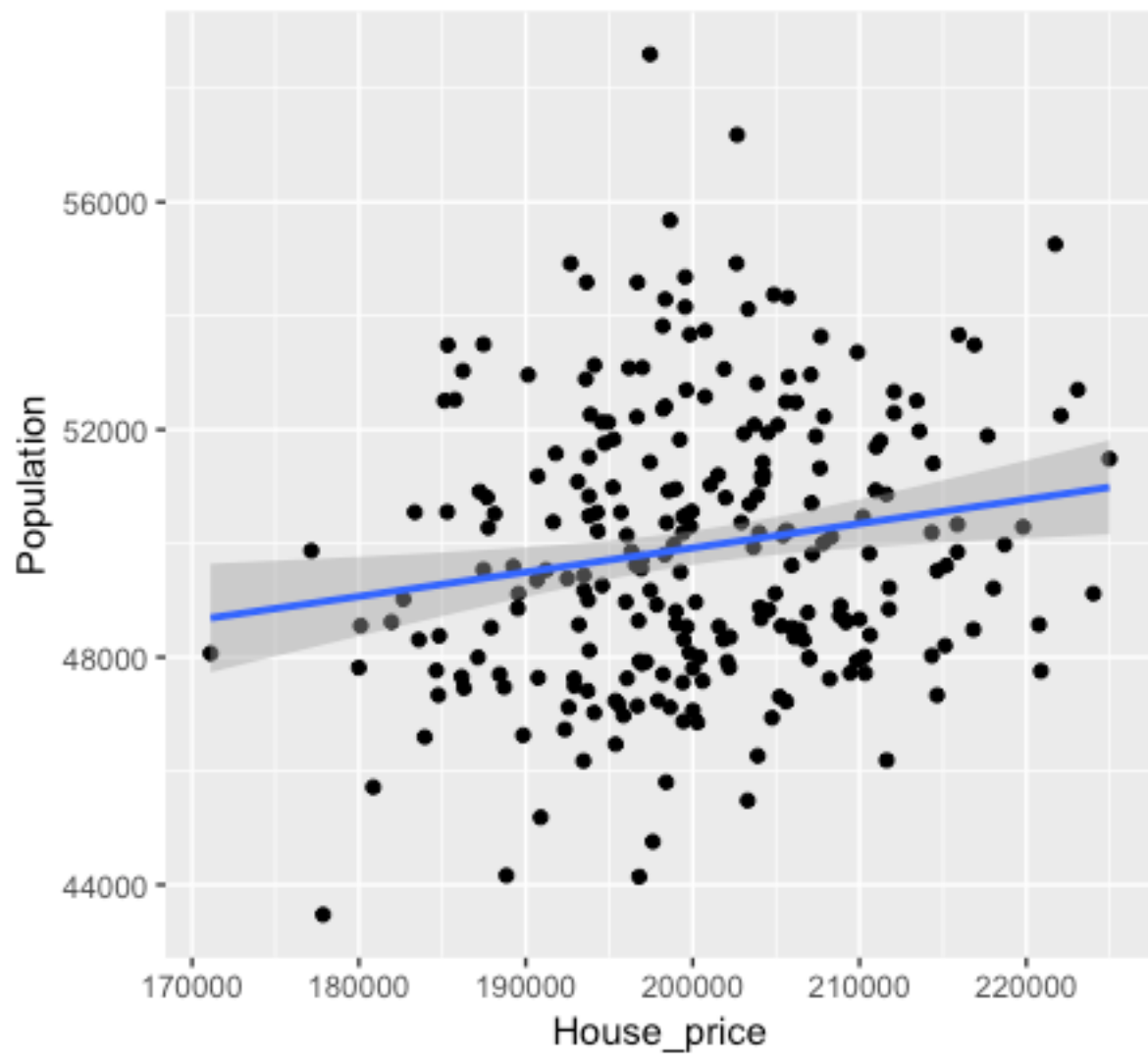
Let's build a correlation plot - need to install the package `ggcorrplot` first:

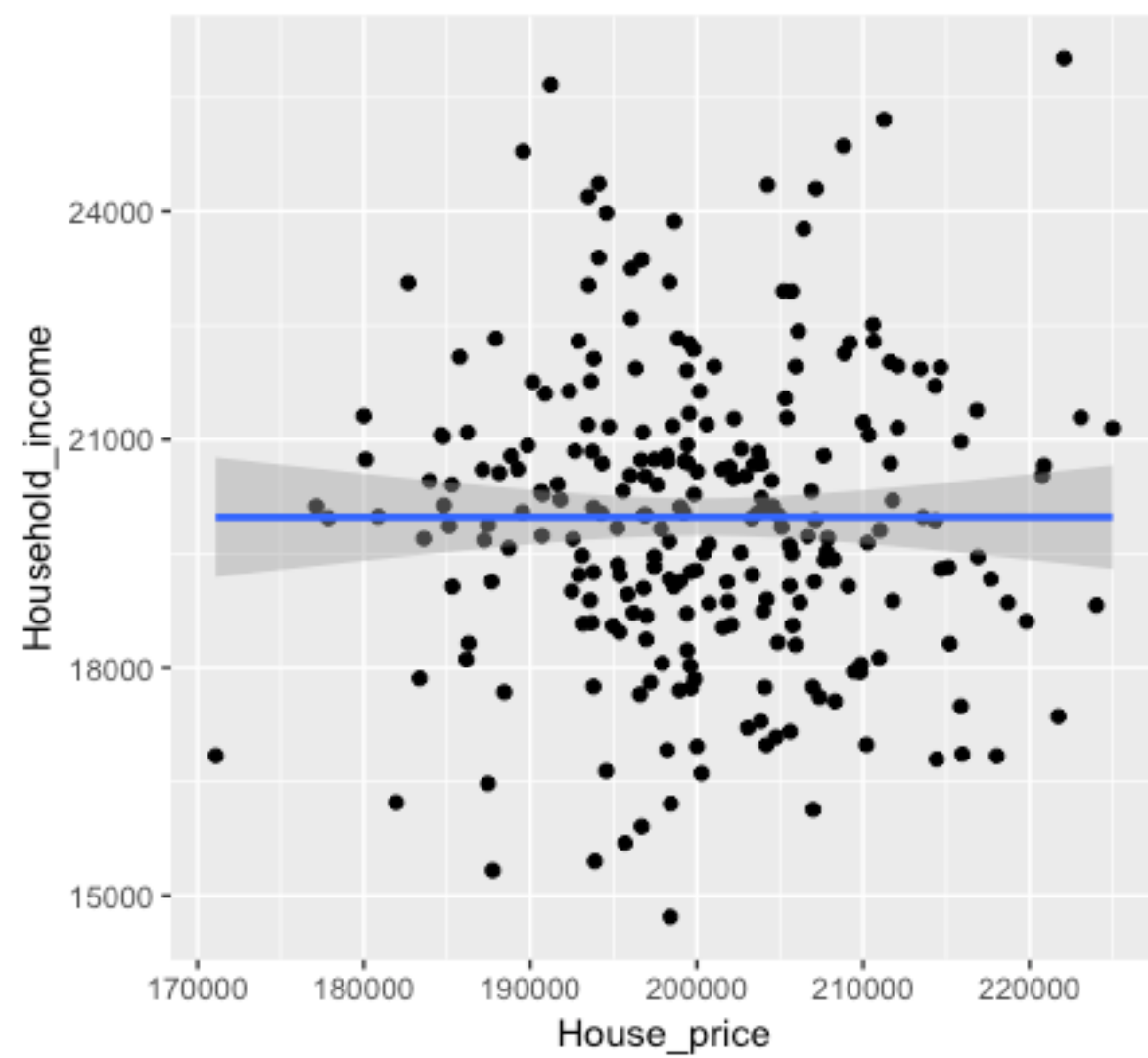
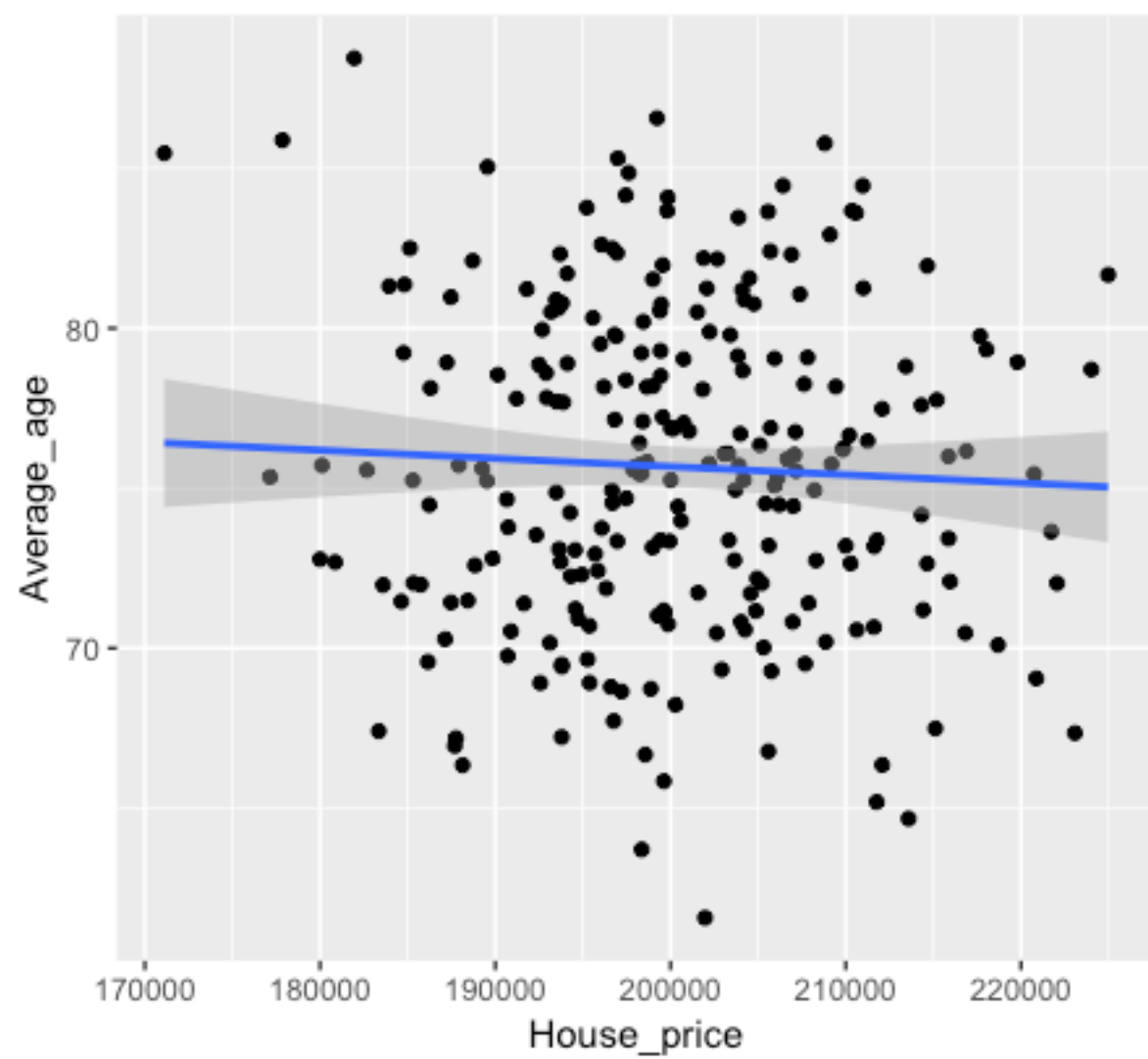
```
corr <- cor(dplyr::select(my_data, -Region))
```

```
ggcorrplot(corr, hc.order = TRUE, type = "lower",  
           lab = TRUE)
```



Let's build some plots looking at the relationships between each predictor and our outcome variable.





First we will build a linear model with all 4 predictors, then a second model with just the intercept (i.e., the mean) - we then compare them - is the model with the 4 predictors a better fit to our data than the model with just the mean?

```
> model0 <- lm(House_price ~ 1, data = my_data)
> model1 <- lm(House_price ~ Population + Crime + Average_age + Household_income,
data = my_data)
> anova(model0, model1)
Analysis of Variance Table

Model 1: House_price ~ 1
Model 2: House_price ~ Population + Crime + Average_age + Household_income
  Res.Df      RSS Df Sum of Sq    F Pr(>F)
1     249 2.3069e+10
2     245 2.1622e+10  4 1446836735 4.0985 0.00311 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The F-ratio comparing our two models is 4.0985 indicating our model with all the predictors is a better fit than our model with just the intercept (the mean). We then need to get our parameter estimates using the function `summary()`

```
> summary(model1)
```

Call:

```
lm(formula = House_price ~ Population + Crime + Average_age +  
    Household_income, data = my_data)
```

Residuals:

Min	1Q	Median	3Q	Max
-26460.7	-6011.9	-386.4	6331.8	24591.6

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	1.807e+05	1.680e+04	10.754	< 2e-16	***
Population	6.577e-01	2.453e-01	2.682	0.00782	**
Crime	-3.358e+02	1.153e+02	-2.913	0.00391	**
Average_age	-8.218e+01	1.186e+02	-0.693	0.48915	
Household_income	-1.957e-02	3.033e-01	-0.065	0.94861	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9394 on 245 degrees of freedom
Multiple R-squared: 0.06272, Adjusted R-squared: 0.04741
F-statistic: 4.098 on 4 and 245 DF, p-value: 0.00311

Here we have our parameter estimates and the t-tests associated with our predictors.

Here are the R-squared and Adjusted R-squared values (which reflects the number of predictors in our model).


```
# Notice that Average_age and Household_income do not seem to predict house prices
# Let's drop them in model2
model2 <- lm(House_price ~ Population + Crime, data = my_data)
anova(model2, model1)
```

```
> anova(model2, model1)
Analysis of Variance Table
```

```
Model 1: House_price ~ Population + Crime
```

```
Model 2: House_price ~ Population + Crime + Average_age + Household_income
```

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	247	2.1666e+10				
2	245	2.1622e+10	2	43401593	0.2459	0.7822

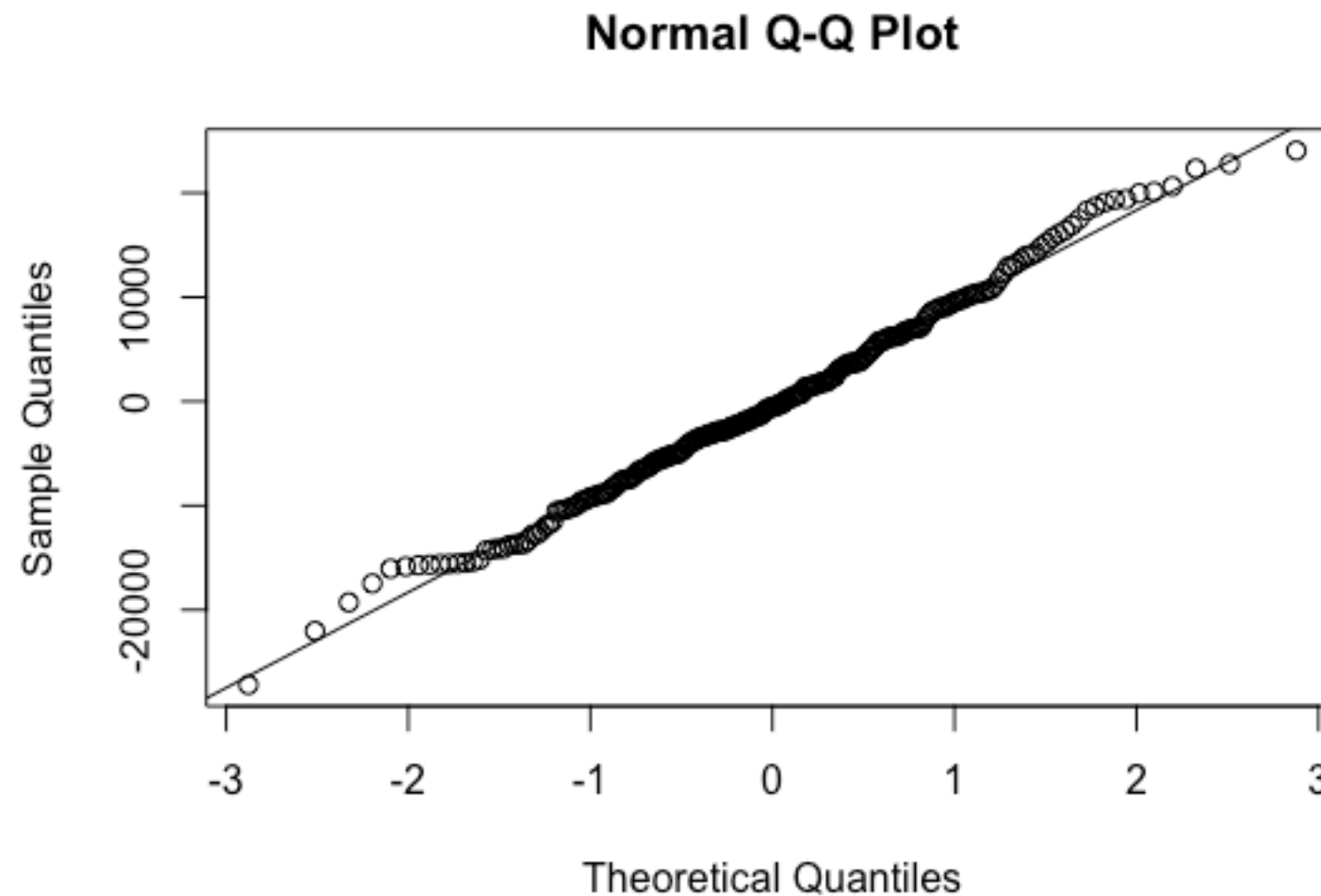
OK, so the models do not differ significantly by this test - we can use another measure of goodness-of-fit - AIC (Aikaike Information Criterion). AIC tells us how much information in our data is not captured by each model - lower values are better - can only be interpreted in a relative sense (i.e., comparing one model to another)...

```
> AIC(model1)
[1] 5290.354
> AIC(model2)
[1] 5286.855
```

We defined `model2` as having just two predictors - as `model2` has the lower AIC value (so more information in our data is explained by `model2` than by `model1`), we would be justified in selecting that as our ‘best’ model. AIC penalises models with increasing number of parameters (but not as much as BIC) so gives us a good trade-off of fitting our data and model complexity.

In the linear model, our residuals need to be normally distributed - the easiest way to check this is to plot them:

```
qqnorm(residuals(model2))  
qqline(residuals(model2))
```

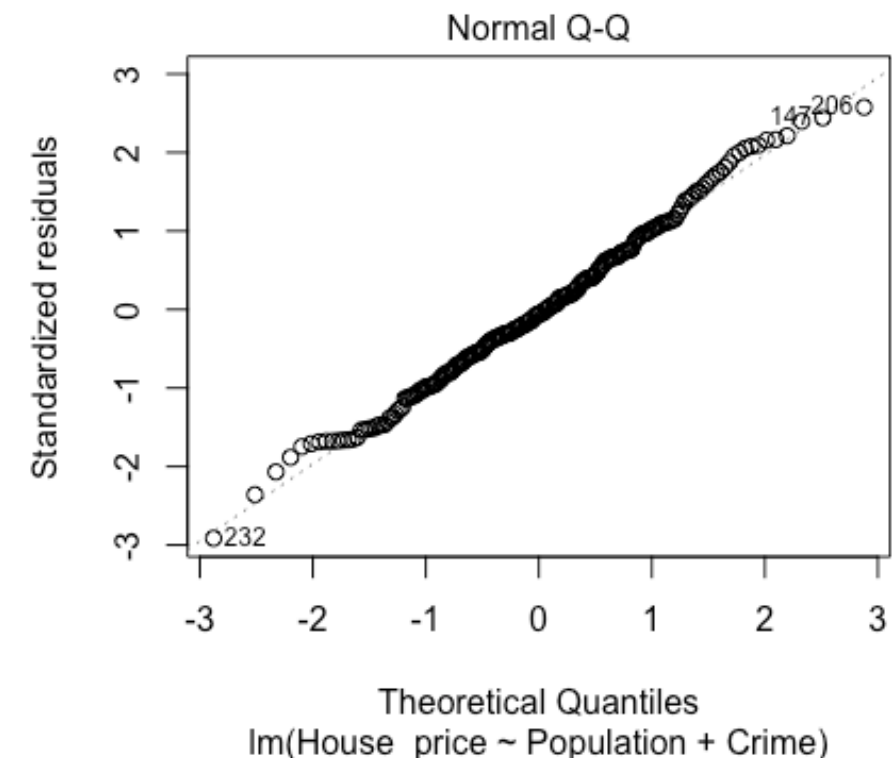
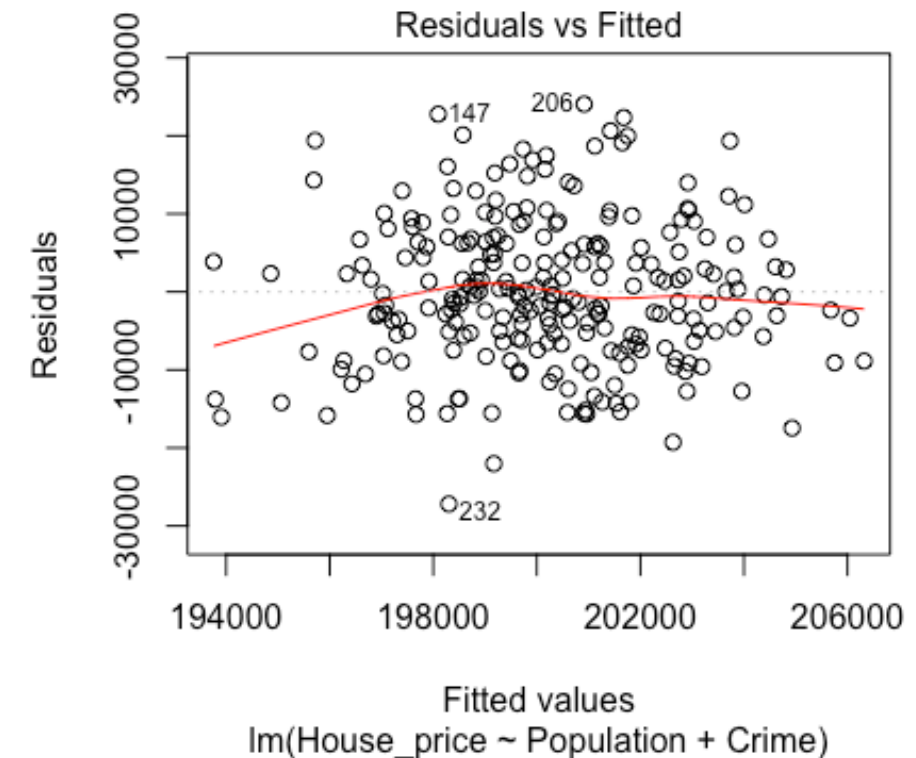
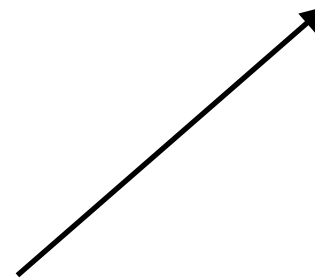


Now let's look at a number of diagnostic plots...

We can use the following command to get some visual representations of model fit:

```
> plot(model2)
```

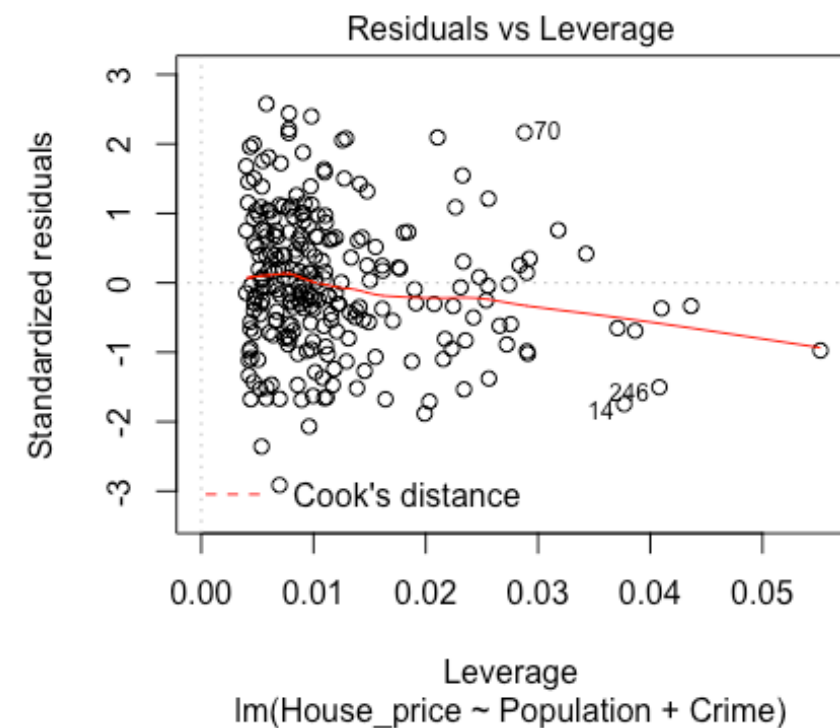
We should have a similar distribution of points (via LOESS Curve Fitting) either side of zero - if we don't it would suggest non-random errors (see Durbin Watson test later). In the Q-Q plot we should see a diagonal line if our residuals are normally distributed.



The Scale-Location plot shows if residuals are spread equally along the ranges of predictors. We used this to check the assumption of equal variance (homoscedasticity). We really want to see a horizontal line with equally, randomly spread points.



The Residuals vs. Leverage plot tells us about influential outliers (i.e., outliers that are affecting our model) - when cases are outside of Cook's distance (beyond the dashed line) it means they are having an influential affect on the regression model - we'd might want to exclude these points and rebuild our model.



Durbin Watson Test

- This tests for the non-independence of errors - our errors need to be independent (one of the assumptions of regression). This test needs the `car` package to be loaded.

```
> durbinWatsonTest(model2)
lag Autocorrelation D-W Statistic p-value
1      -0.03048832      2.055433      0.66
Alternative hypothesis: rho != 0
```

A D-W value of 2 means that there is no autocorrelation in the sample - our calculated value is pretty close to that - $p = .66$ so we conclude our errors are independent of each other.

Stepwise Regression Based on AIC Improvement

Rather than building our regression model step by step manually, we can use the `step` function in R - it takes a starting model, and then uses forwards or backwards procedures (or a combination of both) to produce the best model.

Let's apply the procedure to `model0` and `model1` as our limits - we can specify the stepwise procedure with the parameter "direction":

```
> steplimitsboth <- step(model0,  
                           scope = list(upper = model1),  
                           direction = "both")
```

- Let's focus on the combined method that adds predictors which improve model fit, and removes ones that don't - based on minimising AIC:

```
> summary(steplimitsboth)

Call:
lm(formula = House_price ~ Crime + Population, data = data)

Residuals:
    Min       1Q   Median       3Q      Max
-27192.2  -6161.4   -555.2   6203.4  24061.0

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.736e+05  1.243e+04  13.973  < 2e-16 ***
Crime        -3.343e+02  1.147e+02  -2.915  0.00388 **
Population    6.662e-01  2.442e-01   2.729  0.00682 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9366 on 247 degrees of freedom
Multiple R-squared:  0.06084, Adjusted R-squared:  0.05323
F-statistic:      8 on 2 and 247 DF,  p-value: 0.0004301

> AIC(steplimitsboth)
[1] 5286.855
```

We can see the procedure has settled on the model with Crime and Population. AIC value is 5286.855. In this case the stepwise model is the same as what we arrived at manually.

We can also estimate the confidence intervals for each of our parameters using the `confint()` function - repeating the study, 95% of calculated confidence intervals will include the true value of the parameter.

```
> confint(steplimitsboth, level = 0.95)
              2.5 %              97.5 %
(Intercept)  1.491596e+05 198110.856517
Crime        -5.602084e+02   -108.461481
Population   1.853052e-01    1.147126
```

Stepwise Regression Based on Adjusted R Squared Improvement

- Use the `ols_step_forward` function to work out the model with predictors entered on the basis of improvement via p -value and adjusted R^2 . For this we need the package `olsrr`.

```
# Using ols_step_forward  
  
> library(olsrr)  
> pmodel <- ols_step_forward_p(model1)  
> pmodel
```

```
> pmodel <- ols_step_forward_p(model1)
```

```
Forward Selection Method
```

```
-----
```

Candidate Terms:

1. Population
2. Crime
3. Average_age
4. Household_income

We are selecting variables based on p value...

Final Model Output

```
-----
```

Model Summary			
R	0.247	RMSE	9365.686
R-Squared	0.061	Coef. Var	4.678
Adj. R-Squared	0.053	MSE	87716066.079
Pred R-Squared	0.039	MAE	7416.676

```
-----
```

RMSE: Root Mean Square Error

MSE: Mean Square Error

MAE: Mean Absolute Error

```
-----
```

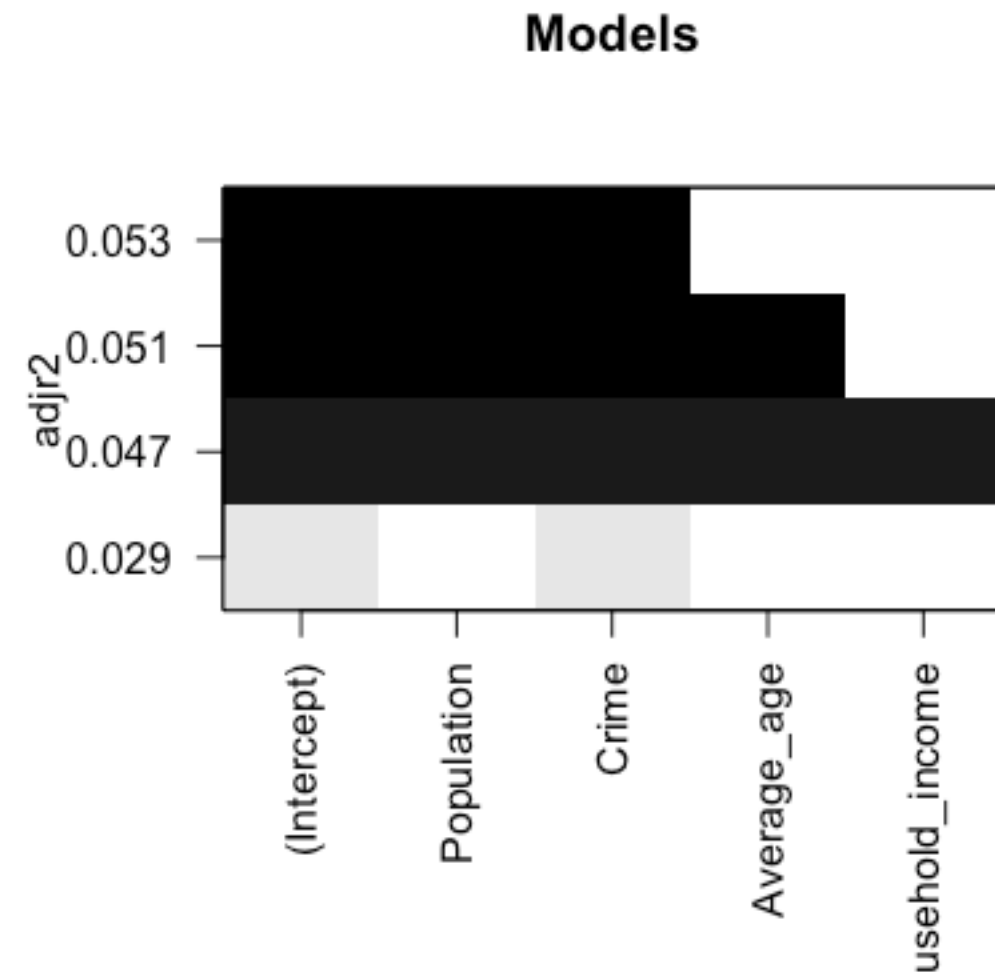
Parameter Estimates							
model	Beta	Std. Error	Std. Beta	t	Sig	lower	upper
(Intercept)	173635.236	12426.603		13.973	0.000	149159.616	198110.857
Crime	-334.335	114.679	-0.180	-2.915	0.004	-560.208	-108.461
Population	0.666	0.244	0.168	2.729	0.007	0.185	1.147

```
-----
```

The model determined by p-value improvement is also the one with the lowest AIC value - but this may not always be the case.

- Visualise the possible models (incl. the one with the largest adjusted R^2 value) using the *leaps* package.

```
> library(leaps)
> leapsmodels <- regsubsets (House_price ~
Population + Crime + Average_age +
Household_income, data = data)
> plot(leapsmodels, scale = "adjr2", main
= "Models")
```



Collinearity?

- We can apply the `vif()` function to our model - it will work out the VIF values for each of our variables - `vif()` is in the `car` package so don't forget to load that...

```
> vif(steplimitsboth)
      Crime Population 
1.000012  1.000012
```

- As a rule of thumb VIF greater than 10 suggests a multicollinearity issue (although greater than 5 has been suggested too - more conservative).
- For our case, we don't have a collinearity problem as the VIF values are low.

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	1.736e+05	1.243e+04	13.973	< 2e-16	***
Crime	-3.343e+02	1.147e+02	-2.915	0.00388	**
Population	6.662e-01	2.442e-01	2.729	0.00682	**

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

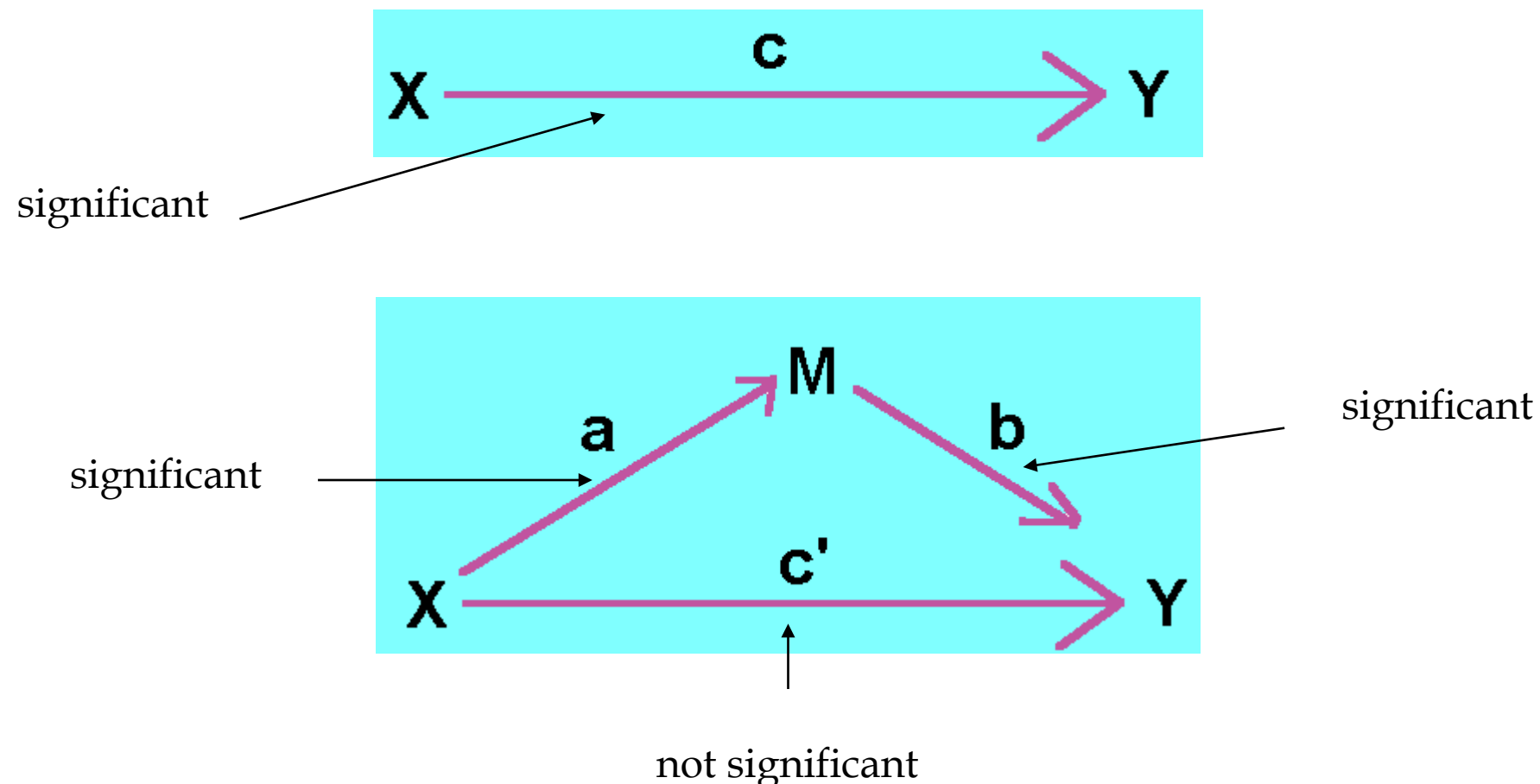
- Using these regression coefficients, we could write our regression equation as something like:

House price = 173,600 - 334.3(Crime) + 0.6662(Population) + residual

- So, crime has a *negative* influence on house prices (more crime = lower prices) while population size has a *positive* influence on house prices (more people = higher house prices).

Mediation Effects

- At times, a variable's predicting power of an outcome is lost (complete mediation) or significantly reduced (partial mediation) when another predictor is introduced.
- This added variable mediates the initial effect by its absence and cancels the effect by its presence.



Step 1: The initial variable is correlated with the outcome.

- X is the predictor and Y the outcome variable (path c).

Step 2: Show that the initial variable is correlated with the mediator.

- X is the predictor and M is the outcome variable in the regression equation (path a).

Step 3: Show that the mediator affects the outcome variable.

- X and M are both predictors and Y is the outcome variable (path b).
- X (and not just M) is included so that the initial variable could be controlled in establishing the effect of the mediator on the outcome.

Step 4: After introducing M the effect of X on Y controlling for M (path c') should be zero.

Mediation in R

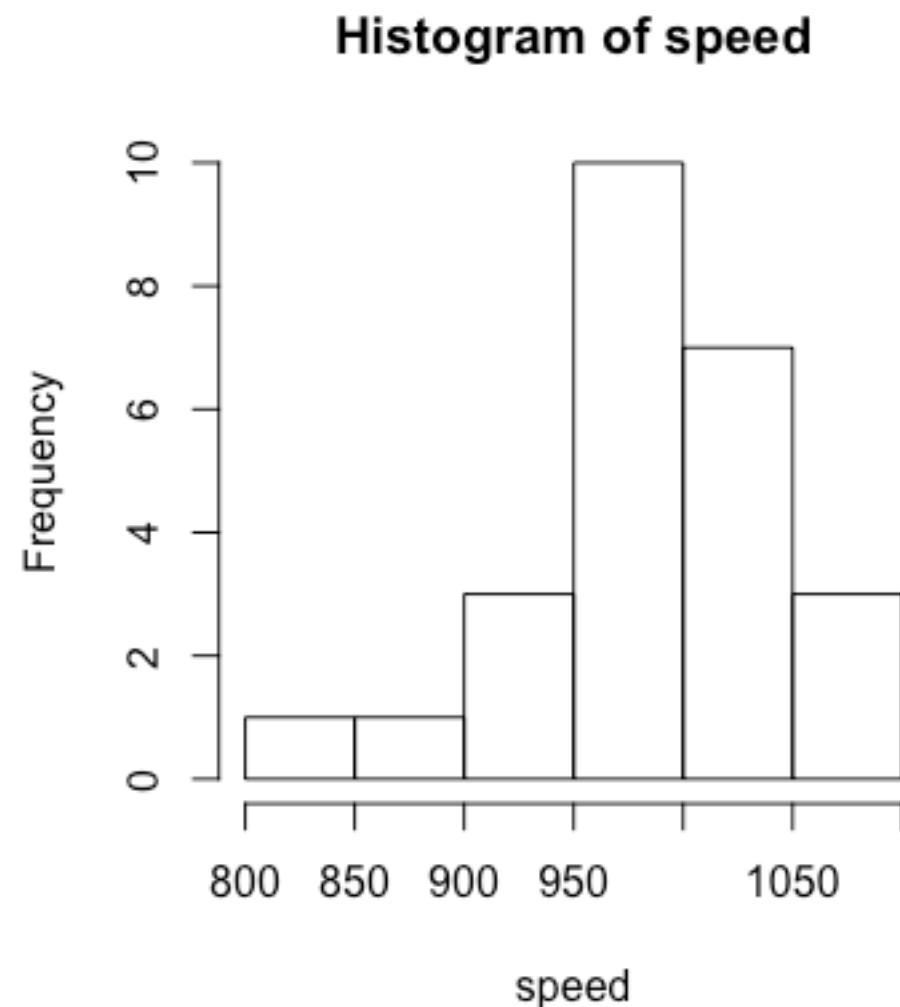
- Imagine we are interested in whether the number of hours since dawn (X) affects the subjective ratings of wakefulness (Y) of 100 graduate students through the consumption of coffee (M).
- The datafile is called `Meddata`

Mediation via Bootstrapping

- Sobel test (which you might have come across previously) can be quite conservative - *bootstrapping* and the *mediation* package a more powerful method to investigate mediation effect. Bootstrapping better for small samples.
- Imagine you have a sample of n measurements, you can sample this in many ways as long as you allow some values to appear more than once and others to be left out (sampling with replacement). Calculate the mean lots of times (one for each sampling - often as many as 10,000) and work out the CIs and other population parameter estimates. This is *bootstrapping*.

Example

- Imagine we measure the speed of 25 military planes and plotted the data:

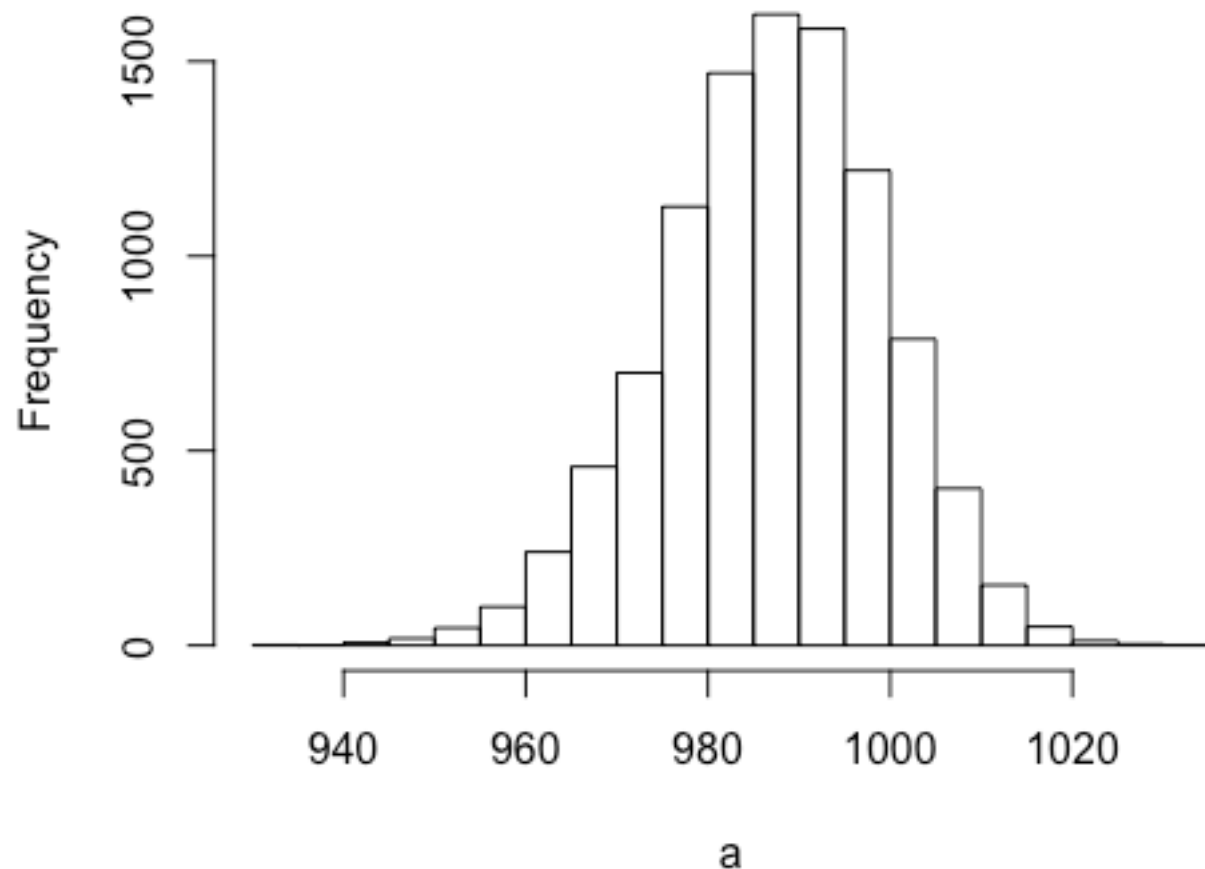


- What might the underlying population of military planes we are sampling from look like?

```
a <- numeric(10000)
for (i in 1:10000) {
  a[i] <- mean(sample(speed, replace = TRUE))
}
hist(a)
```

- This bit of code creates 10,000 samples based on our 25 observations and plots a histogram of these 10,000 sample means.

Histogram of a



- We can use this to estimate population parameters such as mean and CI.
- If we measure a plane with speed of less 950, how likely is that given our estimated population? We can work out the total number of planes with a speed of 950 (which is 18)...
- The probability is therefore less than or equal to 18 in 10,000 (so $p \leq .0018$)

The following mediation and moderation content is from the interactive R book by Alexander Demos & Carlos Salas.

email: ademos@uic.edu

The full book with lots more content is available from here:

<http://ademos.people.uic.edu/index.html>

The Mediation Package

- Based on Preacher & Hayes (2004) - more power than Sobel test
- well suited for small sample sizes.
- Need to build two models - one for direct effect of our predictors and one for the effect of our predictor and mediator.

```
> library(mediation)
> fitM <- lm(M ~ X, data = Meddata) #IV on M; Hours since dawn predicting
coffee consumption
> fitY <- lm(Y ~ X + M, data = Meddata) #IV and M on DV; Hours since dawn and
coffee predicting wakefulness
```

- Then use the `mediate()` function in the `mediation` package to compare the two models - allows us to estimate effect of the mediator...

```
> fitMed <- mediate(fitM, fitY, treat = "X", mediator = "M")
> summary(fitMed)
```

```
> summary(fitMed)
```

Causal Mediation Analysis

Quasi-Bayesian Confidence Intervals

	Estimate	95% CI Lower	95% CI Upper	p-value
ACME	0.276803	0.144987	0.43	<2e-16 ***
ADE	-0.115043	-0.316462	0.07	0.268
Total Effect	0.161760	0.000729	0.31	0.048 *
Prop. Mediated	1.653327	0.507091	9.66	0.048 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample Size Used: 100

Simulations: 1000

- By running this we get the Average Causal Mediation Effects (ACME), our Average Direct Effects (ADE), combined indirect and direct effects (Total Effect), and the ratio of these estimates (Prop. Mediated).
- The ACME is the indirect effect of M (Total Effects - ADE) and the associated p -value value tells us if our mediation effect is significant.
- We can bootstrap our data and fit a model based on our estimated population parameters (which is recommend over the default CI estimation method above)...

```
> fitMedBoot <- mediate(fitM, fitY, boot = TRUE, sims =
10000, treat = "X", mediator = "M")
> summary(fitMedBoot)
```

```
> summary(fitMedBoot)
```

Causal Mediation Analysis

Nonparametric Bootstrap Confidence Intervals with the Percentile Method

	Estimate	95% CI Lower	95% CI Upper	p-value
ACME	0.28078	0.14112	0.43	<2e-16 ***
ADE	-0.11179	-0.29548	0.09	0.273
Total Effect	0.16899	-0.00862	0.35	0.066 .
Prop. Mediated	1.66151	-3.92801	10.91	0.066 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample Size Used: 100

Simulations: 10000

- We now see that the ACME is the only one that is significant - this tells us we have a significant mediator - with no direct effect of our predictor or combined effect of predictor and mediator when the mediator is taken into consideration.

Moderation Effects

- Moderation tests whether a variable (Z) affects the direction and/or strength of the relation between an IV (X) and a DV (Y).
- In other words, moderation tests for interactions that affect when relationships between variables occur.
- Imagine we are interested in whether the relationship between the number of hours of sleep (X) a student receives and the attention that they pay to this lecture (Y) is influenced by their consumption of coffee (Z).

Our predictor variables are X and Z - good idea to centre them (i.e., mean of zero) using the `scale` function. It is important to centre both the moderator and predictor to make interpretation easier.

```
#Centering Data
Xc      <- c(scale(X, center = TRUE, scale = FALSE)) #Centering
IV; hours of sleep
Zc      <- c(scale(Z, center = TRUE, scale = FALSE)) #Centering
moderator; coffee consumption
```

```
> fitMod <- lm(Y ~ Xc + Zc + Xc:Zc) #Model interacts IV & moderator
> summary(fitMod)
```

Call:

```
lm(formula = Y ~ Xc + Zc + Xc : Zc)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-21.466	-8.972	-0.233	6.180	38.051

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	48.54443	1.17286	41.390	< 2e-16	***
Xc	5.20812	0.34870	14.936	< 2e-16	***
Zc	1.10443	0.15537	7.108	2.08e-10	***
Xc:Zc	0.23384	0.04134	5.656	1.59e-07	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 11.65 on 96 degrees of freedom

Multiple R-squared: 0.7661, Adjusted R-squared: 0.7587

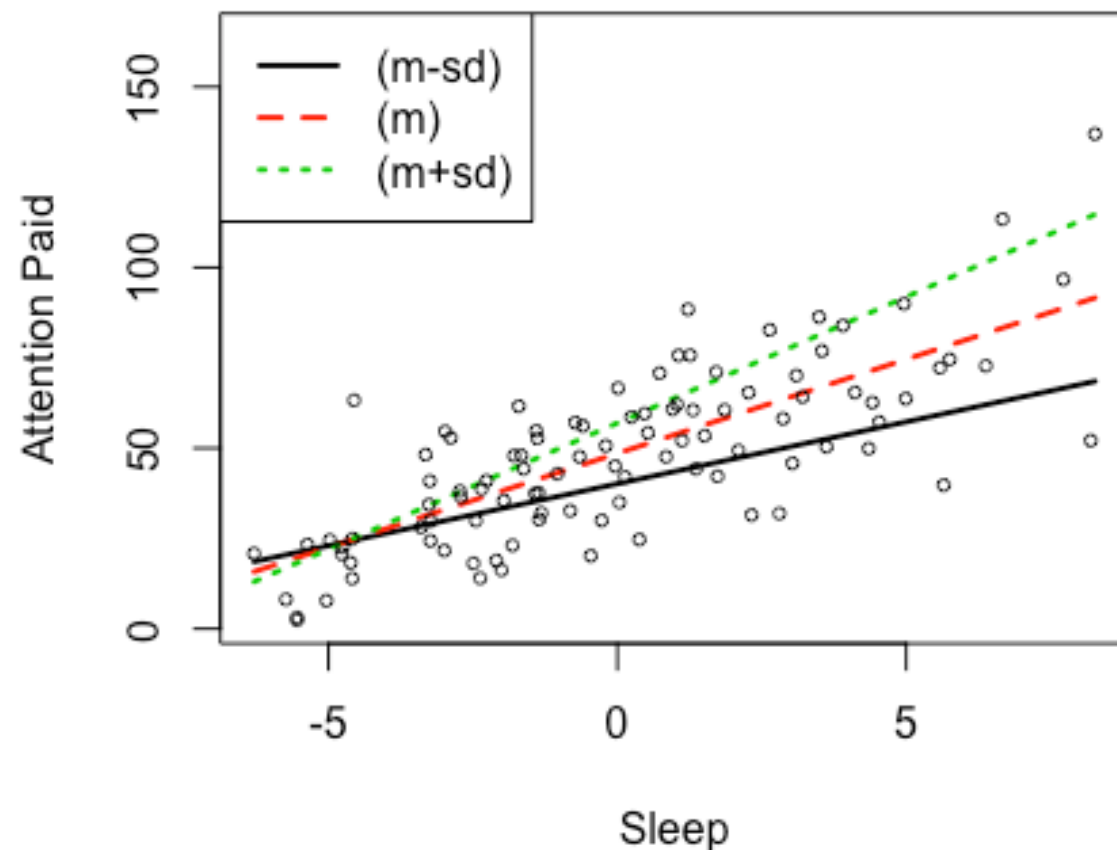
F-statistic: 104.8 on 3 and 96 DF, p-value: < 2.2e-16

We can use the `plotSlopes` function in the `rockchalk` package to plot the slopes (1 SD above and 1 SD below the mean) of the moderating effect.

The plot below shows that those who drank less coffee (the black line) paid more attention with the more sleep they got last night, but paid less attention overall than average (the red line).

Those who drank more coffee (the green line) paid more attention when they slept more as well and paid more attention than average.

The difference in the slopes for those who drank more or less coffee shows that coffee consumption moderates the relationship between hours of sleep and attention paid.



```
ps <- plotSlopes(fitMod, plotx = "Xc", modx = "Zc", xlab = "Sleep", ylab = "Attention Paid", modxVals = "std.dev")
```

Suppressor Effects

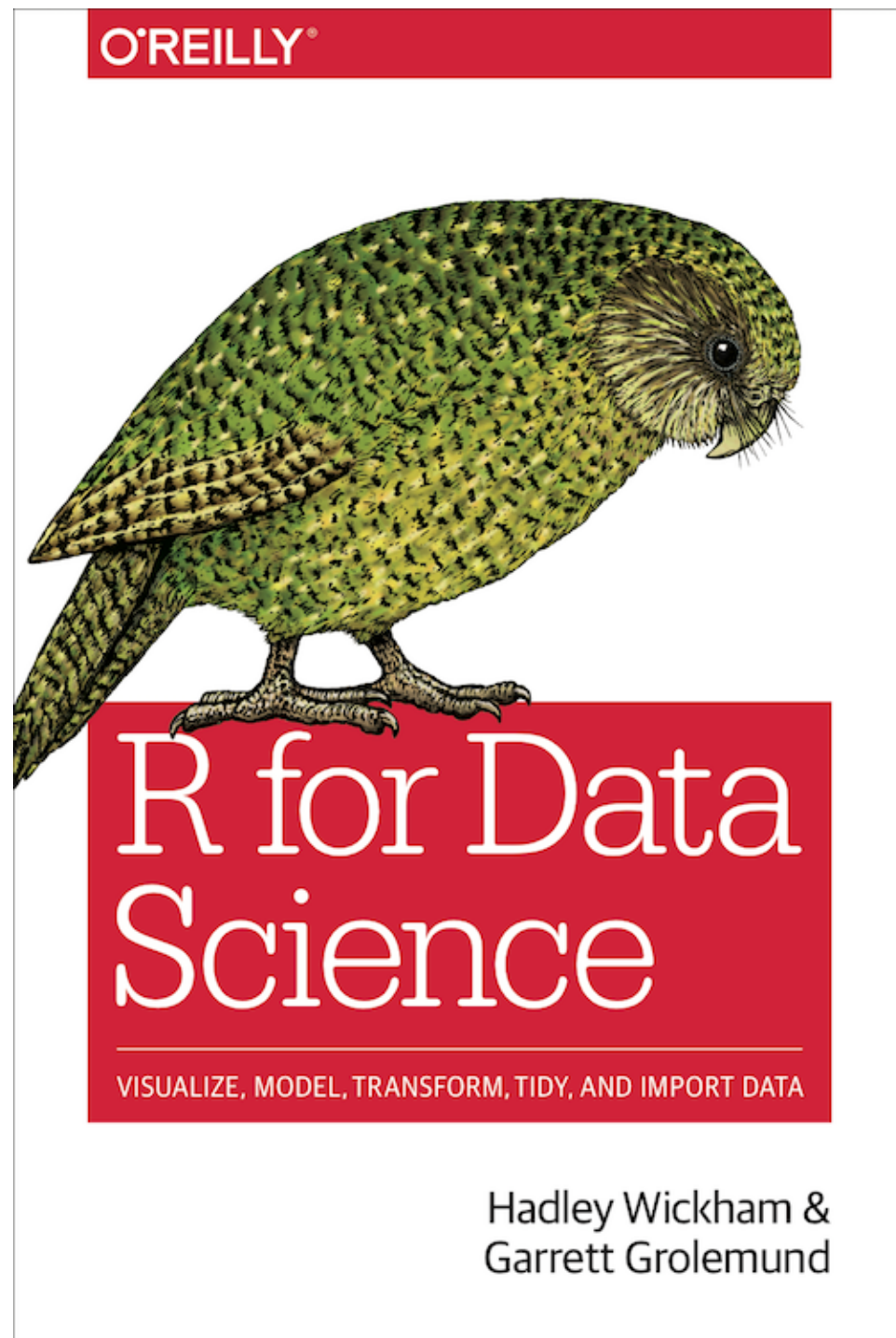
- Normally, we would not consider a variable that does not correlate significantly with our outcome variable as an important predictor and not include it in a regression analysis.
- But sometimes a variable that does not correlate with the outcome appears to have a significant effect on the model when entered in the regression.
- This variable is called a *suppressor* and it improves the regression model by suppressing variance in other predictor variables.
- Although the suppressor variable does not correlate with the output variable it does correlate (often quite strongly) with at least one of the other predictor variables.

Example of a suppressor effect

Imagine research shows that in train drivers there's a relationship between IQ and tendency to make errors (missing red lights, speeding etc.) with higher IQ drivers tending to make fewer errors - however, higher IQ drivers might also become bored more quickly (which actually increases errors).

In this case, boredom might be a suppressor variable and adding it as a predictor should give a better indication of the relationship between IQ and error making while train driving. We'll look at the effect of a suppressor in the lab...

Over Reading Week...



You might want to consolidate your R knowledge by working through the R4DS book - I recommend the first 10 chapters at this point - you can buy the book, but it is also freely available electronically at:

<http://r4ds.had.co.nz>

To the worksheet...