|  |
| --- |
| *Engine type and number of gears* |
| **Background**  This is an unusual post for me, I have avoided writing about R Markdown because there are so many resources already available on the topic. The aim of this was to demonstrate the usefulness of R Markdown (and hopefully convert a few people). For this session I created a set of resources[1](http://feedproxy.google.com/~r/Cillianmacaodh/~3/ToCJIVrxpEA/#fn1) aimed at making the transition from SPSS to R Markdown a bit easier. The *statistics* content of these resources is mainly just some of the simpler standard tests taught to psychology undergraduate students.  **R Markdown Basics**  I started off the workshop by going through some of the basics of R Markdown. When working with R Markdown, there are three types of text to be concerned with.   1. **Plain text**: this is what you will use to write your manuscript. The formatting of this plain text is fairly straight forward 2. **Code chunks**: these are denoted by three back ticks ``` at the top and the bottom of a chunk with commands in curly brackets in the title (on the same line as the firs set of back ticks). Code chunks allow you to run analyses without leaving your document. Output from code chunks can be included or omitted from your main output document. 3. **In-line code**: this is denoted by a single back tick to begin a piece of code and a single back tick to close it code goes here.   Each time you open a piece of code (a chunk or in-line code) you have to identify what language you want to code in. This is done by including the letter “r” in the curly brackets or immediately after the opening back tick.  **Working with R**  In order to show some of the basic functionality of R, I ran some analyses using the data sets that are built into R. This means that anyone should be able to reproduce the analyses conducted using the template I provided (without needing to worry about loading data from other files). I also provided another document with an accompanying template detailing the steps for inputting data into R. But this process is more prone to errors and if you are not familiar with R it can be a bit unintuitive.  **Working with dataframes**  A dataframe is structured much like an SPSS file. There are rows and columns, the columns are named and generally represent variables. The rows (can also be named) generally represent cases. You can have multiple data frames loaded with different names, although they are commonly saved as df (and these can be numbered df1 df2 df3. If your document/code is well organised, it can be useful have a generic name for dataframes that you are working with. This means that much of your code can be recycled (particularly if the variable names are the same – if you run repeated studies, or studies with only minor changes, you will find that there is massive scope for recycling code – both chunks and in-line)  **Some basics:**   1. The entire dataframe can be printed to the console by running the name of the data frame 2. The dollar sign can be used to call specific variables from the data frame i.e., df$variable\_name 3. A function has the form “function name” followed by parenthesis: function\_name(). 4. The object that you want to run the function on goes in the parenthesis. e.g., if our dataframe was called df, and age was called age and we wanted to get the mean age we would run mean(df$age). 5. Sometimes missing data denoted by NA can mess with some functions, to account for this it is helpful to include the argument na.rm = TRUE in the function, e.g., mean(df$age, na.rm = TRUE)   The mtcars dataset comes with R. For information on it simply type help(mtcars). The variables of interest here are am (transmission; 0 = automatic, 1 = manual), mpg (miles per gallon), qsec (1/4 mile time). Below we practice a few simple functions to find out information about the dataset.  **Example code and output:**   * Load the mtcars dataset into an object called df using the command df <- mtcars * View the variable names associated with df by running variable.names(df)   ## [1] "mpg" "cyl" "disp" "hp" "drat" "wt" "qsec" "vs" "am" "gear"  ## [11] "carb"   * The mean miles per gallon can be calculated using mean(df$mpg)   ## [1] 20.09062   * The standard deviation of the same variable is calculated using sd(df$mpg)   ## [1] 6.026948   * Or if you want to see basic descriptives use descriptives(df$mpg)[3](http://feedproxy.google.com/~r/Cillianmacaodh/~3/ToCJIVrxpEA/#fn3)   ## mean sd min max len  ## 1 20.09062 6.026948 10.4 33.9 32   * To index by a variable we use square brackets [] and the which() function.   + The following command gets the mean miles per gallon for all cars with manual transmission:     - mean(df$mpg[which(df$am==1)])   ## [1] 24.39231  **Statistical tests**  Now that we know some of the basics, we’ll try running some statistical tests.  **Running a t-test**  I’m going to use the mtcars dataset to illustrate how to run and write up a t-test, in such a way that:   1. there is no copy/paste error in reporting values; and 2. you can recycle code and text.   In addition to mpg (miles per gallon), the other variable I am interested in is am (transmission; 0 = automatic, 1 = manual).[4](http://feedproxy.google.com/~r/Cillianmacaodh/~3/ToCJIVrxpEA/#fn4)  The question I’m going to look at is:   1. Is there a difference in miles per gallon depending on transmission?   **T-test: Transmission and MPG**   * Load mtcars and save it in your environment using df <- mtcars * Create a new dataframe with a generic name e.g., x using the command: x <- df * This command runs the t-test and you can see the output in the console t.test(x$mpg~x$am)   ##  ## Welch Two Sample t-test  ##  ## data: x$mpg by x$am  ## t = -3.7671, df = 18.332, p-value = 0.001374  ## alternative hypothesis: true difference in means is not equal to 0  ## 95 percent confidence interval:  ## -11.280194 -3.209684  ## sample estimates:  ## mean in group 0 mean in group 1  ## 17.14737 24.39231   * The following code runs the t-test but saves the output as a list t that can be called later: t <- t.test(x$mpg~x$am) * As with dataframes, specific variables within a list can be called using the dollar sign * To call the p value simply type t$p.value   ## [1] 0.001373638   * To call the t statistic, type t$statistic   ## t  ## -3.767123   * And to call the degrees of freedom, type t$parameter   ## df  ## 18.33225   * Finally, to calculate the effect size and save it to an object type td <- cohensD(mpg~am, data=x)   From the above we can call each value we need using in-line code to write up our results section as follows  **This is what the paragraph will look like in your Rmd document:**  An independent samples t-test revealed a significant difference in miles per gallon between cars with automatic transmission (\*M\* = `r mean(x$mpg[which(x$am==0)])`, \*SD\* = `r sd(x$mpg[which(x$am==0)])`), and cars with manual transmission, (\*M\* = `r mean(x$mpg[which(x$am==1)])`, \*SD\* = `r sd(x$mpg[which(x$am==1)])`), \*t\*(`r t$parameter`) = `r t$statistic`, \*p\* `r paste(p\_report(t$p.value))`, \*d\* = `r td.  **The above syntax will return the following:**  An independent samples t-test revealed a significant difference in miles per gallon between cars with automatic transmission (*M* = 17.15, *SD* = 3.83), and cars with manual transmission, (*M* = 24.39, *SD* = 3.83), *t*(18.33) = -3.767, *p* = .001, *d* = 1.48.  If you want to run another t-test later on in your document you simply run it in a code chunk and create new objects (t and td) with the same names as before and you can use the same write up as above to report it.  **Chi-square**  To illustrate a chi-squared test I will test if there is an association between engine type vs (0 = V-shaped, 1 = straight) and transmission type (0 = automatic, 1 = manual). Again this is to illustrate the code/write up – not a real test.   * First create a table using the command table(x$am,x$vs) * Running a chi-squared test using chisq.test(table(x$am,x$vs)) returns:   ##  ## Pearson's Chi-squared test with Yates' continuity correction  ##  ## data: table(x$am, x$vs)  ## X-squared = 0.34754, df = 1, p-value = 0.5555   * As with the t-test, in order to report it using in-line code you need to save the test as an object e.g., c using c <- chisq.test(table(x$am,x$vs)) * Save the effect size using w <- sqrt(c$statistic/(length(x$mpg)\*c$parameter)) * And calculate the observed power using pw <- pwr.chisq.test(w=w,N=length(x$mpg),df=(c$parameter),sig.level = .05)   **Report using the following**  A chi-squared test for independence revealed no significant association between engine type and transmission type, χ^2^(`r c$parameter`, \*N\* = `r length(x$mpg)`) = `r c$statistic`, \*p\* `r paste(p\_report(t$p.value))`, \*V\* = `r w, the observed power was `r pw$power`.  **The above returns the following:**  A chi-squared test for independence revealed no significant association between engine type and transmission type, χ2(1, *N* = 32) = 0.348, *p* = .556 *V* = 0.1, the observed power was 0.09.  **Tables**  Again using the mtcars dataset, I’m going to make a few tables. Bear in mind that the code for these tables is designed to be used along with the papaja package and rendered to pdf, so the html tables in this post will not be formatted as correctly as they should be (refer to the resources on the OSF to see what it should look like).   * Firstly save mtcars as an object using x <- mtcars * We will make a table of engine type (V-shaped vs Straight) agains number of gears (3, 4, or 5) using table(x$vs,x$gear) which returns:   ##  ## 3 4 5  ## 0 12 2 4  ## 1 3 10 1   * This table is useful to us, but it doesn’t really sit well in our document * We can address this using the apa\_table() function * Before using this function we need to create a matrix that can be passed through the function using the command test <- as.data.frame.matrix(table(x$vs,x$gear)) * We can give names to the rows using test <-rownames<-(test, c("V-shaped","Straight")) * And we can give names to the columns using test <-colnames<-(test, c("3 gears","4 gears","5 gears")) * Finally we pass our object (matrix) test through the apa\_table() function to render a table that will be embedded in our document * The code below will generate the Table [1](http://feedproxy.google.com/~r/Cillianmacaodh/~3/ToCJIVrxpEA/#tab:simpletable):   apa\_table(  test  , align = c("l", "c", "c", "c")  , caption = "Engine type and number of gears"  , added\_stub\_head = "Engine Type"  #, col\_spanners = makespanners()    ) | Table 1: |