Student Exercises

your name here

Exercise 1.1: Exploring R Studio

Take a few minutes to familiarize yourself with the R studio environment by locating the following features: - The windows clockwise from top left are: the code editor, the workspace and history, the plots and files window, and the R console. - In the plots and files window, click on the packages and help tabs to see what they offer. - See what types of new files can be made in R studio by clicking the top left icon- open a new R script.

Now open the file called 'Student_Exercises_for_Workshop_Lectures.Rmd'. This file will serve as your digial notebook for parts of the workshop and contains the other exercises.

Exercise 1.2: Intro to `R Markdown` Files

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
```

Each code chunk begins and ends in the same way- with a fence. You can further specify what you want to show up in your final document using the `echo` and `eval` commands in the opening line. Insert a few code chunks below using the `insert` tab at the top of this window. Then, change the `echo` and `eval` arguments to TRUE or FALSE and see how different combinations of these arguments change the output when you knit. I have done the first one for you. Notice too that each `R` code chunk requires a unique title argument (here 'cars variant 1'), or the Rmd will not knit.

```
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
```

What do you think `echo` and `eval` do, based on your manipulations?

What are the defaults for `echo` and `eval`, based on your manipulations?

Exercise 1.3: Basic Mathmatics in `R`

Insert a code chunk below and complete the following tasks: 1. Add and subtract 2. Multiply and divide 3. Raise a number to a power using the ^ symbol 4. Create a more complex equation involving all of these operations to convince yourself that `R` follows the normal priority of mathematical evaluation

Exercise 1.4: Assigning Variables and Arithmetic Functions in `R`

Insert a code chunk below and complete the following tasks: 1. Assign three variables using basic mathmatic operations 2. Take the log of your three variables 3. Use the print function to display your most complex variable 4. Use the `concatenate` function to print a sentence

Exercise 1.5: Vectors and Factors

Insert a code chunk below and complete the following tasks: 1. Create a numeric vector using the `c` function 2. Create a multi-level character factor using the `c` function 3. Use `str` and `class` to evaluate your variables

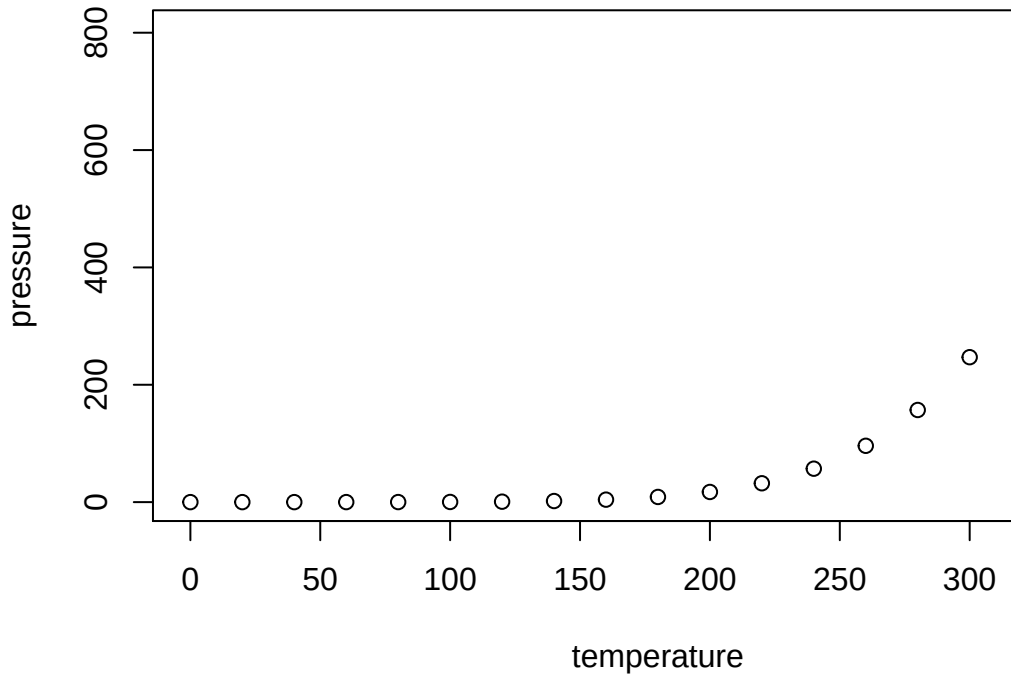
Exercise 1.6: Basic Statistics

Insert a code chunk below and complete the following tasks: 1. Create a vector and calculate the mean, sd, sum, length, and var 2. Use the log and sqrt functions on your vector 3. What happens when you try to apply these functions to a factor? 4. Type the first couple letters of a function into your chunk, then hit tab- what happens?

Exercise 1.7: Creating Larger Vectors and Random Sampling

Insert a code chunk below and complete the following tasks: 1. Create a vector with 100 elements using the `seq` function and calculate two basic statistics on your vector 2. Create a variable and `sample` it with equal probability 3. Create a normally distributed variable of 1000 elements using the `rnorm` function then `sample` that variable with and without replacement 4. Use `hist`, `curve`, and `dnorm` to plot your normally distributed variable

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.

Exercise 1.8: Basic Visualization

Insert a code chunk below and complete the following tasks, make sure to label all plot axes and have fun with colors! 1.Create a variable using `seq` and make two different plots by changing the `type` argument 2.Create a normally distributed variable using `rnorm` and make two different plots using `hist` by varying the `breaks` argument (what does `breaks` appear to do?) 3. Modify your parameter arguments to create a composite figure of the above graphs.

Exercise 1.8: Basic Visualization