Save your file as an R SCRIPT file with the name CHAPTER_FIVE_EXERCISES.R in your WORKING DIRECTORY folder. As you work through the exercises in this chapter, remember to regularly save the contents of your R CONSOLE window (which will contain the R objects you have created up to that point) to your WORKSPACE file and, if you are using RStudio, the contents of your SCRIPT EDITOR window to your R SCRIPT file.

Finally, you need to remove any data that are currently held in R's temporary memory. To do this, enter the following command into R:

```
rm(list=ls())
```

If you are using RGUI, you can simply type this code after the command prompt at the bottom of the R CONSOLE window (it looks like this: >) and then press the ENTER key on your keyboard to run it. If you are using RStudio, you can type this command into the SCRIPT EDITOR window (the upper left hand window). To run this command, select it and then click on the RUN button at the top of this window. This will run it in the R CONSOLE window (the lower left hand one in the main RStudio user interface). You are now ready to start the exercises in this chapter.

EXERCISE 3.1: HOW TO CREATE AN X-Y SCATTER PLOT DISPLAYING ONE DATA SERIES USING GGPLOT:

An X-Y scatter plot is the most common type of graph used by biologists to display the values for individual data points in a data set. On such graphs, one point is plotted for each row of data, with their positions being determined by the values that are plotted on the X and Y axes. In order to be able to do this, you need to have your data arranged in a spreadsheet or table where each row contains data from a single record in your data set. In this table, there also needs to be separate columns containing the values for the variables you wish to plot on the X and Y axes of your graph. These will usually be continuous variables, but one or both of them can also be ordinal variables (that is, ones measured on an ascending scale). In general, you should avoid plotting data on categorical scales on an X-Y scatter plot. Instead, it would be better to use a bar graph for such data.

In this exercise, you will create an X-Y scatter plot which shows how relative investment in reproductive tissues in male bats species with complex echolocation (known as microchiropteran bats) varies with the average body mass of each species. The relative investment in reproduction will be measured as the percentage of the total body mass that is made up of the testes during the breeding season (see MacLeod and MacLeod 2009. *Oikos*. 118: 903-916). In both cases, the body mass data and the percentage testes mass (or PTM for short) data have been log-transformed. To do this, work through the following flow diagram (**NOTE:** If you have not already done so for an earlier exercise, you will need to download the `ggplot2` package and install it in your version of R before you start working through these instructions – see Exercise 1.1 for details of how to do this):

Data set held in a comma separated values (.CSV) file

For this example, the data set you will use is stored in a file called `all_bat_data.csv` that is located in the WORKING DIRECTORY folder you created during the introduction to this chapter.

1. Set the WORKING DIRECTORY for your analysis project

Before you start any analysis in R, you first need to set the WORKING DIRECTORY. To do this, enter the text `setwd("` and then type the address of your WORKING DIRECTORY, using slashes (/) as the folder separators, before entering a second quotation mark followed by a closing bracket, like this `")`. For example, if your WORKING DIRECTORY has the address C:\STATS_FOR_BIOLOGISTS_TWO, your `setwd` command should look like this:

```
setwd("C:/STATS_FOR_BIOLOGISTS_TWO")
```

If you are using RGUI, enter your `setwd` command in the R CONSOLE window (remembering to use the address of your own WORKING DIRECTORY folder in it) and then press the ENTER key on your keyboard. If you are using RStudio, enter your `setwd` command into the SCRIPT EDITOR window. To run it, select it and then click on the RUN button at the top of this window. You will enter all the remaining commands for this exercise in a similar manner, depending on the user interface you are using.

To check that your WORKING DIRECTORY has been set properly, enter the command `getwd()` and carefully check that the address it returns is the same as the one for the STATS_FOR_BIOLOGISTS_TWO folder you created at the start of this chapter.

Before you move on to step 2, make sure that all the data you wish to use in your analysis project are located in this WORKING DIRECTORY folder. In this case, this is a file called `all_bat_data.csv`. **NOTE:** If the data you are going to import into R in step 2 are not located in the WORKING DIRECTORY you set in this step, the import code provided in the next step will not work.

2. Load your data into R using the `read.table` command

The `read.table` command provides the easiest way to load data held in a .CSV file (and stored in the WORKING DIRECTORY you set in step 1) into R so you can analyse it. To do this for the data set being used in this example, enter the following command into R:

```
all_bat_data <- read.table(file=
"all_bat_data.csv",sep=",",as.is=FALSE,
header=TRUE)
```

This code has to be entered exactly as it is written here or it will not work. If you wish to use the copy-and-paste approach for entering this command, copy the text directly below CODE BLOCK 69 in the document R_CODE_DATA_ VISUALISATION_WORKBOOK.DOC and paste it into R.

This command will create a new object in R called `all_bat_data` which will contain the data from the specified .CSV file. To import a different .CSV file into R, all you need to do is change the file name in the `file` argument to the name of the one you wish to import. You can also use whatever name you wish for the R object which will be created by this command. To do this, simply replace `all_bat_data` at the start of the first line of the above code with the name you wish to use for it. <u>**NOTE**</u>**:** If your .CSV data set uses a semicolon as the column separator, you would need to replace the `sep=","` argument with `sep=";"`.

3. Check the data have loaded into R correctly by checking the names of the columns and by viewing it

Whenever you import any data into R you need to check that they have loaded correctly. First, you need to check that all the required columns are present in the R object you just created. To do this, enter the following command into R:

```
names(all_bat_data)
```

This is CODE BLOCK 70 in the document R_CODE_DATA_ VISUALISATION_WORKBOOK.DOC. This command will return the names used for each column in the R object you just created. For this example, the names should be: `id`, `log_body_mass`, `log_ptm` and `type`.

Next, you should view the contents of the whole table using the `View` command. This is done by entering following code into R:

```
View(all_bat_data)
```

This is CODE BLOCK 71 in the document R_CODE_DATA_ VISUALISATION_WORKBOOK.DOC. This command will open a DATA VIEWER window where you can examine your data set and check that the correct data have been loaded into R.

In the data set being used in this example, there are data from two different types of bat: Mircochiropteran bat species, which have complex echolocation, and macrochiropteran bat species, which do not. However, for this example, you only wish to create an X-Y scatter plot of data from the microchiropteran bat species. This means that before you can do anything else, you will need to create a new R object that only contains the data from the microchiropteran bat species. In R, this is done using the `subset` command. To do this for the data set being used in this example, enter the following code into R:

```
micro_bat_data <- subset(all_bat_data,
              type=="micro")
```

4. Create a new R object containing just the subset of data you wish to create an X-Y scatter plot from

This is CODE BLOCK 72 in the document R_CODE_DATA_ VISUALISATION_WORKBOOK.DOC. This command will create a new R object called `micro_bat_data` which will only contain the rows from the `all_bat_data` data set that have the label `micro` in the `type` column. This indicates these data are from microchiropteran bat species

Next, you should view the contents of the subset of data you have just created using the `View` command. This is done by entering following code into R:

```
View(micro_bat_data)
```

This is CODE BLOCK 73 in the document R_CODE_DATA_ VISUALISATION_WORKBOOK.DOC. This command will open a DATA VIEWER window where you can examine your data set and check that it contains the required subset of data. In this example, it should only contain data from microchiropteran bat species and not from any macrochiropteran bat species (which have the label `macro` in the `type` column).

Once you have successfully subsetted your data, you are ready to create your initial scatter plot. However, before you can do this, you need to load the `ggplot2` command library into your analysis project. To do this, enter the following command into R:

5. Load the `ggplot2` command library into your analysis project

```
library(ggplot2)
```

This is CODE BLOCK 74 in the document R_CODE_DATA_ VISUALISATION_WORKBOOK.DOC.

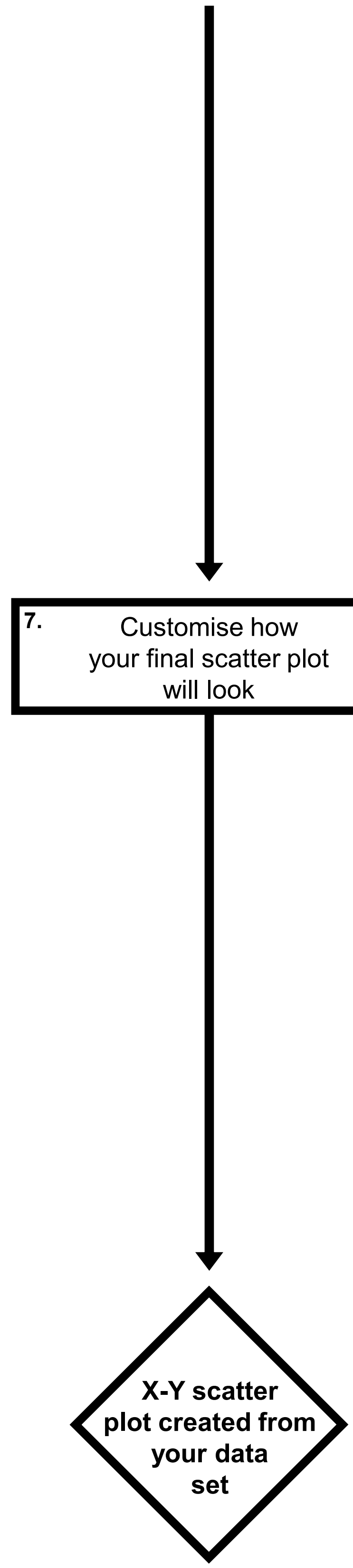
6. Create your initial scatter plot based on the subset of data created in step 4

After you have loaded the ggplot2 command library into your R project, you are ready to use it to create your initial scatter plot based on the log body mass and log percentage testes mass (PTM) data from the microchiropteran bat species. To do this, enter the following block of code into R:

```
ggplot(data=micro_bat_data,
aes(x=log_body_mass,y=log_ptm)) +
geom_point()
```

This is CODE BLOCK 75 in the document R_CODE_DATA_ VISUALISATION_WORKBOOK.DOC, and it contains two commands separated by a + symbol. These are the ggplot command and the geom_point command. The ggplot command sets the data set which will be used for the graph. This is done using the data argument and, in this case, it will be the R object called micro_bat_data created in step 4 of this exercise. The data that will be plotted on the X axis of the resulting graph is set using the x argument in the aes element of this ggplot command. In this case, it is the column called log_body_mass in the micro_bat_data data set, which contains log-transformed body mass data for each bat species. The data that will be plotted on the Y axis of the resulting graph is set using the y argument in the aes element of this ggplot command. In this case, it is the column called log_ptm, which contains log-transformed percentage testes mass data for each bat species.

The second command in this code block, geom_point, sets the type of graph that will be created from the data specified in the ggplot command. In this case, it will be a scatter plot containing one point for each row of data in the data set specified in the data argument of the ggplot command. Initially, no arguments will be included in this command.


7.
Customise how your final scatter plot will look
X-Y scatter plot created from your data set

Once you have created your initial scatter plot, and you are sure that it is displaying the intended data on each axis, you can customise how your final graph will look. This can be done by adding a number of new style commands to the block of code used to produce it. To do this, edit the code from step 6 so that it looks like this (the newly added style commands are highlighted in **bold**):

```
ggplot(data=micro_bat_data,
aes(x=log_body_mass,y=log_ptm)) +
geom_point() + labs(x="Log Body Mass (g)",
y="Log Percentage Testes Mass")+
xlim(c(0.5,2.5)) + ylim(c(-1.0,1.0)) +
theme_classic()
```

This is CODE BLOCK 76 in the document R_CODE_DATA_VISUALISATION_WORKBOOK.DOC, and it adds four new style commands separated by `+` symbols to the code block used to create your initial scatter plot in step 6. These are the `labs` command, the `xlim` command, the `ylim` command, and the `theme_classic` command. The `labs` command sets the labels that will be used tor the X axis (using the `x` argument) and the Y axis (using the `y` argument). The `xlim` command sets the minimum and maximum values for the X axis of the graph. In this case, these will be `0.5` for the minimum value and `2.5` for the maximum value. The `ylim` command sets the minimum and maximum values for the Y axis of the graph. In this case, these will be `-0.1` for the minimum value and `1.0` for the maximum value. Finally, the `theme_classic` command sets the remaining style elements of the final graph to those of the pre-existing classic theme. Once you have finished editing this code block, you can run it again to create the final version of your graph.