5 Plotting

Graphical data representations are one of the strong points of R. The program can produce a vast variety of plots for different kinds of data, all of which can be customised and adapted to the user's taste and needs. You can type the command `demo(graphics)` to get a flavour. Here we will use our brain and body size data to introduce some of the basic plotting commands.

Before we start, we shall make use of the `attach` function and type

```
> attach(massFrame)
```

This command attaches our data set to the current work space, telling R that this is the data frame we are currently using. Doing so saves us time, because we can now simply refer to the columns within `massFrame` without specifying the data frame name each time. For example, we can now access the column with body mass values directly with `Body_mass` rather than `massFrame$Body_mass`. If you try this without attaching the data frame, you will get an error message

```
> Body_mass
 Error: object 'Body_mass' not found
```

When we are finished with analysing `massFrame` we can use the command `detach(massFrame)` to revoke the attachment. Running this command will stop R from looking for variables in the data frame, meaning that we will have to specify the table again when trying to access columns. Otherwise, however, it will leave the data frame intact. If we want to completely erase the data set, we can type `remove(massFrame)` or `rm(massFrame)` , which will remove it completely from the workspace.

5.1 Histograms

Note that this section assumes that you have added the log transformed columns (Section 4.3) and attached the data frame.

First of all we will make some histograms of the log transformed data. A histogram is a plot that shows the distribution of values of a continuous variable. For this purpose, equally-sized intervals are defined along the continuous axis of values and the number of observations that fall into each interval counted. To make a histogram in R, we use the `hist` function. We shall first make a histogram of the log transformed body mass (Figure 5.1). A histogram represents a series of data values, so the most important argument (and the only one required to run `hist`) is the name of the variable containing the data values. In our case, this is `log_Body_mass`, so we run

```
> hist(log_Body_mass, main="Histogram of log body mass", xlab="log
    body mass")
```

Here, in addition to specifying the data, we have given the histogram a title (through the `main` argument) and labelled the x axis (through the `xlab` argument). Note that R adds a label to the y axis by default but we could relabel it using the `ylab` argument. The `hist` function can take many more arguments, allowing you to alter many aspects of the plot (see the function's help page). A useful one is `breaks`, which lets you specify the intervals into which the data is subdivided.

We can also make the histogram for the brain mass which is shown in Figure 5.2.

```
> hist(log_Brain_mass, main="Histogram of log brain mass", xlab="log
    brain mass")
```

5.2 Scatter plots

To examine relationships between two continuous variables, we often want to make a scatter plot. We can do this in R using the `plot` function, which takes the name of the two variables containing the data values as arguments. For example, to examine the relationship between the logarithms of body mass and brain mass we can do the following.

```
> plot( log_Body_mass, log_Brain_mass, main="Scatter plot of body vs
    brain mass", xlab="log body mass", ylab="log brain mass" )
```

The resulting plot is shown in Figure 5.3. On this log-transformed data we can clearly see an approximate linear relationship between the body mass and the brain mass. This well known phenomenon is know as allometric scaling. The slope of the line on a log-log plot is known as the allometric scaling coefficient.

Wikipedia : Allometry.

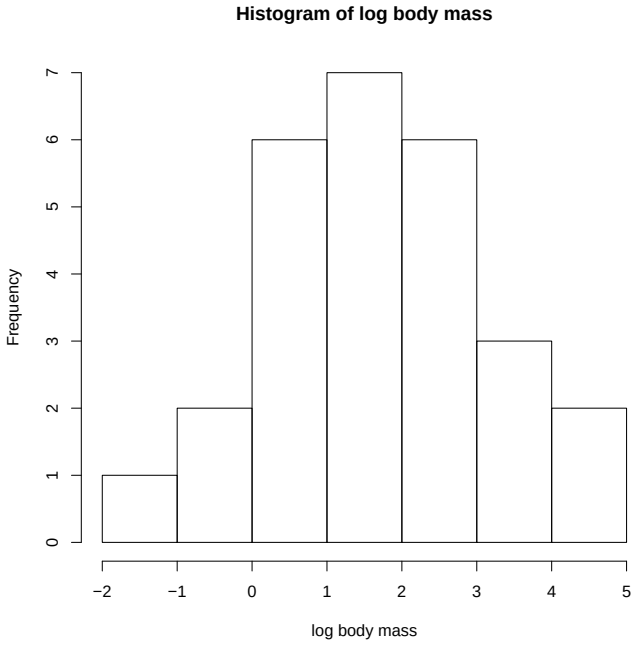


Figure 5.1: A histogram of the log transformed body mass values.

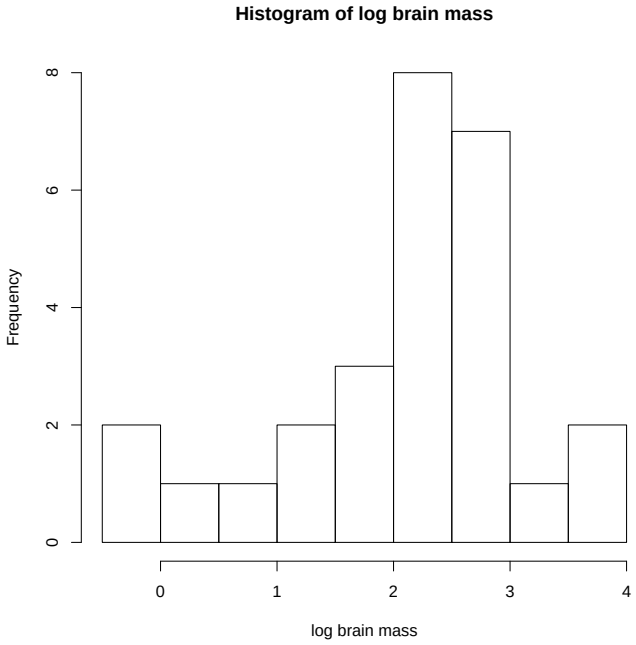


Figure 5.2: A histogram of the log transformed brain mass values.

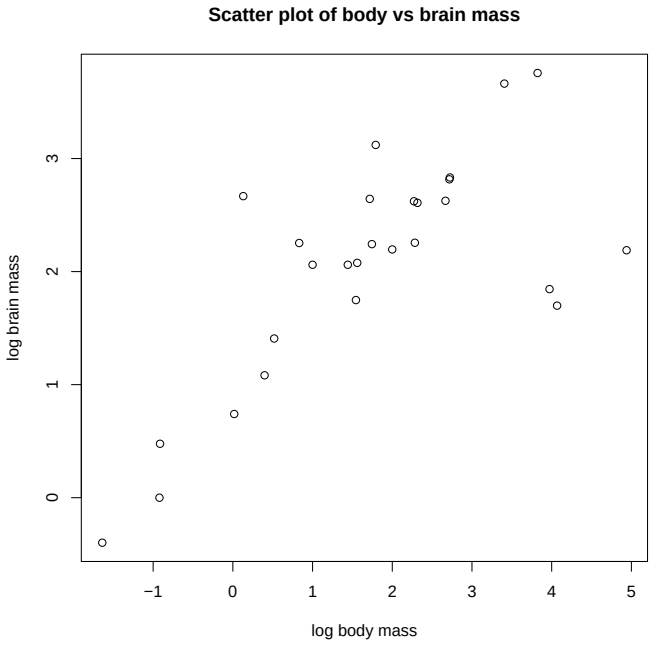
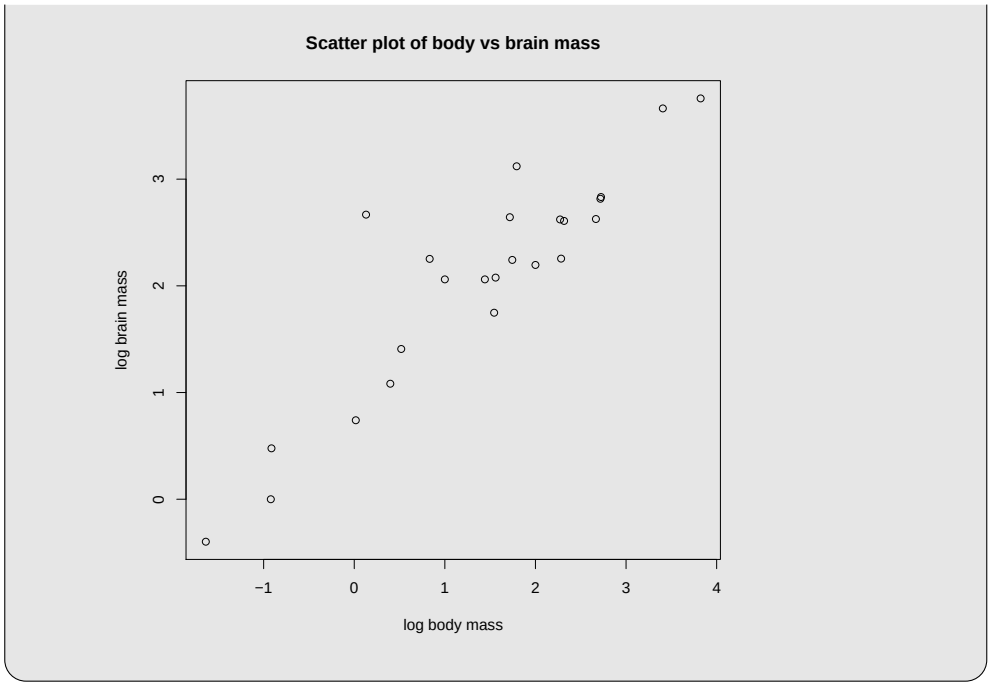


Figure 5.3: A scatter plot of the log transformed brain mass values against the log transformed body mass values.

Exercise 5.1

1. The scatter plot of animal body mass against brain mass has three points that lie to the right hand side corresponding to the dinosaur estimates. Remove these points and remake the plot. Hint: use the `subset` or `which` function and suitable values of the log body and brain mass to remove those points and create a new data frame.
   Solution:

```
> subFrame <- subset(massFrame, log_Body_mass < 3 | log_
    Brain_mass > 3 )
> plot( subFrame$log_Body_mass,
  subFrame$log_Brain_mass,
  main="Scatter plot of body vs brain mass",
  xlab="log body mass",
  ylab="log brain mass")
```



5.3 Saving plots

We shall finish this section on plotting with an illustration of how to save a plot to a file. There are many options and configurations available and here we shall just introduce a simple way to save plots (or any graphics object). R uses the concept of a graphics device. Essentially once a graphics device is started, the subsequent graphics output will be written to a device. There are different types of devices. A window showing a plot would be one, a file on your computer's hard drive to which the graphic is written would be another.

For example, to create a the log body mass histogram in a pdf file we can do the following.

```
> pdf("my_histogram.pdf")
> hist(log_Body_mass, main="Histogram of log body mass",
 xlab="log body mass")
> dev.off()
```

Here, we have first opened up a pdf graphics device and passed as an argument the name of the file that we wish to write to. We have then created the histogram in the normal way and then closed the device with the `dev.off()`. The latter command closes the last opened device, in this case our PDF file. When saving graphics to a file, we must always remember to close the device. Otherwise the file will not be saved properly.

There are many different graphics devices within R. Look at the documentation, `help(Devices)`, for a list of the supported formats e.g. PNG:

```
> png("my_histogram.png")
> hist(log_Body_mass, main="Histogram of log body mass",
 xlab="log body mass")
> dev.off()
```