Working with Data Files

Entering data into R can be troublesome; editing that data in R can be even more of a pain. So, we would typically save our data in a spreadsheet (like MS Excel or similar) that supports easy editing. Then we save this as a tab-separated (text) file and import to R.

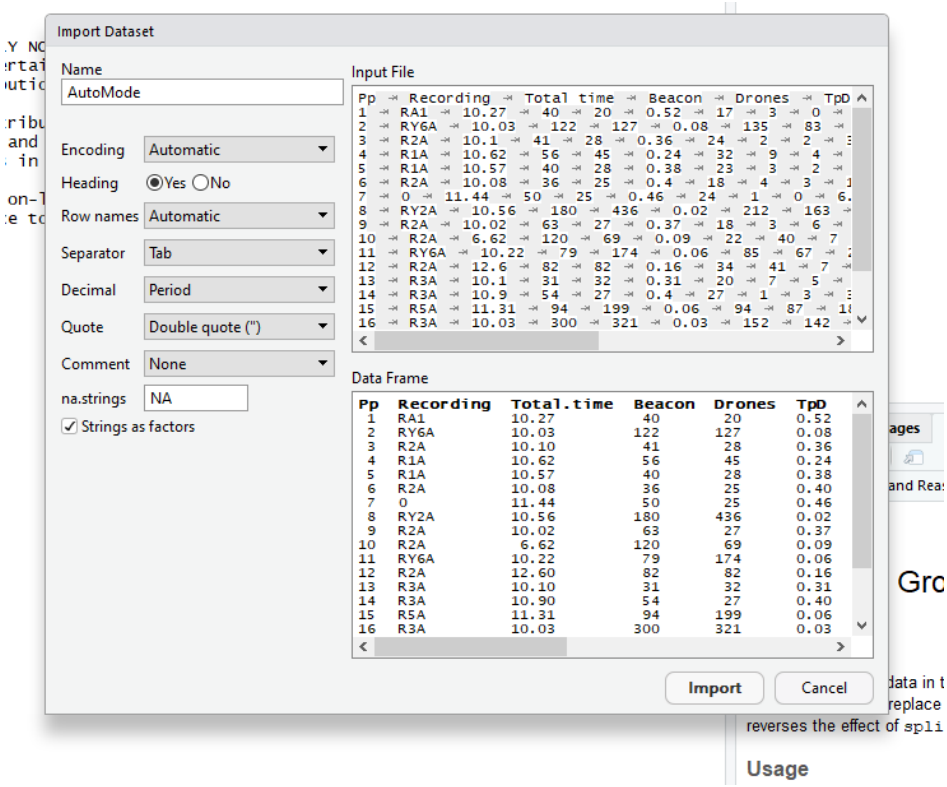
We will begin this session with data from 47 participants who were playing the Drones Game in the drones hidden mode.

From the EMS Canvas page, get the 'AutoMode' file.

From rstudio, use 'Tools' > 'Import dataset' > 'From Local file…'

Then find the 'AutoMode' file and Open.

You should get a pop-up with the option to 'Import'



Hitting 'Import', we should get the following message in the console:

```
> AutoMode <- read.delim("E:/DRONE_EXPTS/Student Expts/AutoMode.txt")
>   View(AutoMode)
```

The contents of the file should appear in the top frame of rstudio. Scroll across the window and you will see a note that 5 participants cannot be included in further analysis, so our dataset has 42. Reporting this, we would say that the experiment recruited 47 participants but initial check of the data indicated that 3 participants had failed to set the experiment properties correctly, 1 had failed to save any data, and 1 ran the game but did not use any of the controls.

Notice that each Dependent Variable is in a separate column, and each Participant is on a separate row. For most of the statistical tests that we will be doing with R, this layout of data is conventional.

We will want to use this file as a data frame:

```
> attach(AutoMode)
```

You can check that the file has been imported by typing its name and this will bring up the entire file in your console:

```
> AutoMode
```

You can also check individual columns, for example,

```
> d.
 [1]  3.47  0.64  3.58  1.80  3.10  1.89  6.36 -0.11  1.39 -1.52 -0.09 -0.69  1.02  3.50 -0.22  0.03 -1.05
[18] -0.33 -1.11 -0.33 -0.31  0.00  3.59 20.25 18.00  0.00  1.46  2.45  3.59  2.20  0.61 -0.38 -0.67 -1.67
[35]  1.87  4.70  0.30  0.81  1.98  0.52  2.20 -0.36
```

Or

```
> Total.time
 [1] 10.27 10.03 10.10 10.62 10.57 10.08 11.44 10.56 10.02  6.62 10.22 12.60 10.10 10.90 11.31
10.03  5.79
[18]  2.33 13.78 11.13 10.08  7.66  9.79 10.80  9.98 10.10 10.00 10.38  5.22 10.00  8.90  6.30  5.04
7.35
[35]  9.77 10.94 10.03 10.29 10.44  9.85 10.06 10.80
```

EXPLORING THE DATA

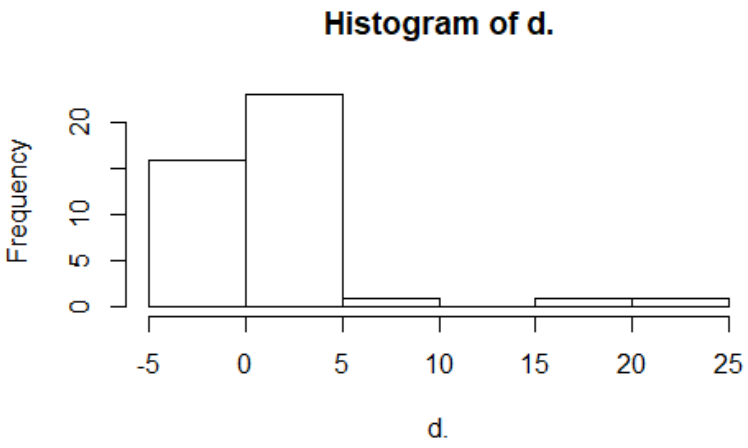
Prior to running tests, it makes sense to get a feel for the data. We might want to inspect individual columns. For example, we look at the signal detection sensitivity (d') scores (which R has labelled d.) using the 'sort' command.

```
> sort(d.)
 [1] -1.67 -1.52 -1.11 -1.05 -0.69 -0.67 -0.38 -0.36 -0.33 -0.33 -0.31 -0.22 -0.11 -0.09  0.00  0.00  0.03
[18]  0.30  0.52  0.61  0.64  0.81  1.02  1.39  1.46  1.80  1.87  1.89  1.98  2.20  2.20  2.45  3.10  3.47
[35]  3.50  3.58  3.59  3.59  4.70  6.36 18.00 20.25
```

Scores of less than 0 indicate a biased response in that the person is more likely to respond to a false alarm than a target. Scores of 0 indicate that the person is not distinguishing target from false alarm. Scores above 0 indicate positive discrimination, i.e., the person is mainly responding to targets and is ignoring false alarms.

Another way of inspecting these data is to plot a histogram.

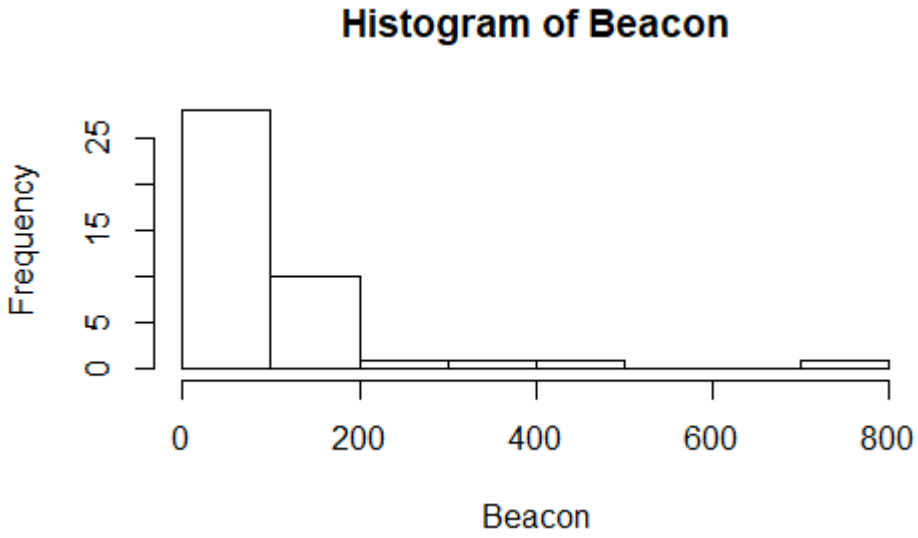
```
> hist(d.)
```



This shows that we have some very high values of d' (indicating that the participant achieved near perfect performance), but a lot of very low values (indicating that participants either had very low sensitivity or had not actually read the instructions).

The column 'Beacon' indicates how often participants activated the Beacons (i.e., pressed the space-bar).

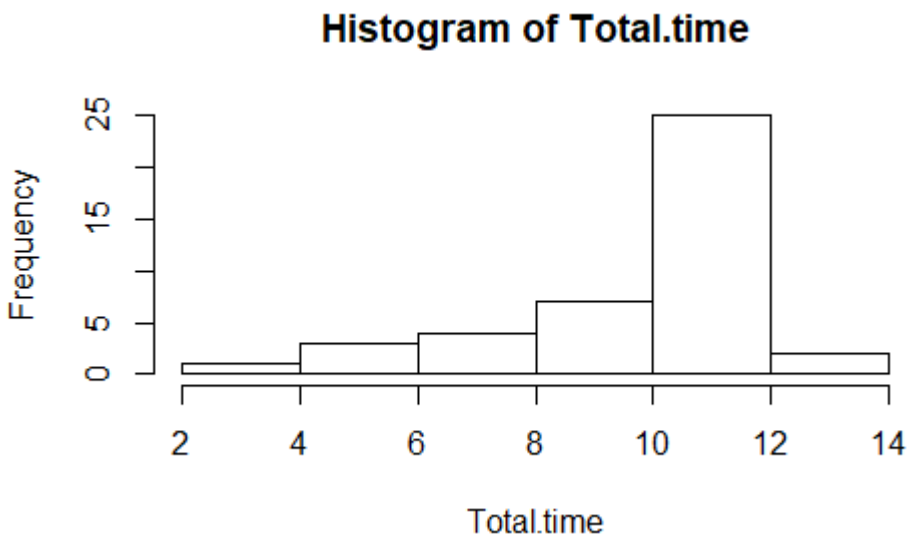
> hist(Beacon)



In this case, there are some very high values. This suggests that, rather than selectively activating the Beacon when a Drone was an eligible target, participants might have been hammering the space-bar as if this was a 'shoot-em-up' game (again, suggesting that they had failed to read the instructions).

Furthermore, participants were asked to play the game for 10 minutes…

> hist(Total.time)



We can see that, while most participants were able to follow this instruction, some played the game for far less time than was requested.

DECIDING ON THE DATA TO USE FOR ANALYSIS

Obviously, you will want to use as much of the data that you have collected as you can. This means that discarding any of the data should be seen as a last resort. However, in our initial exploration we could make a case for excluding participants who seem to be outliers; either because their performance is too good (which we probably won't want to do), or because they have not played the game properly (by pressing the space-bar too enthusiastically), or because they have not followed the instructions of game time (by playing for much less than 10 minutes).

We could remove participants (rows) by selecting a specific row, e.g., let's remove participant 14.

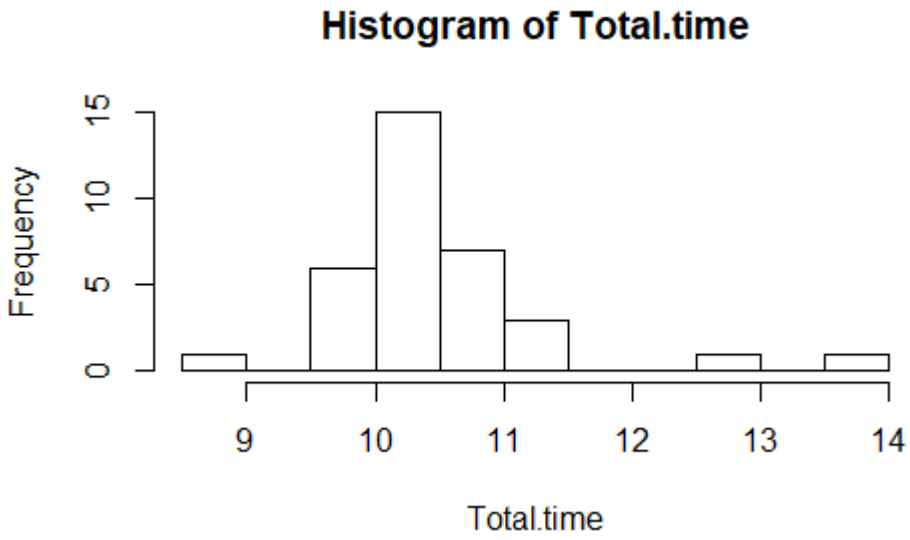
```
> test<-AutoMode[-c(14),]
> test
```

Or we could remove participants by some defined condition, such as a total time of 7.5 or more to be included in the analysis

```
> AutoMode2<-subset(AutoMode, Total.time>7.5)
> AutoMode2
```

To work with this revised set, we want to create a new data frame (because the labels would be confused with those in AutoMode).

```
> detach(AutoMode)
> attach(AutoMode2)
> hist(Total.time)
```



This has also removed samples (which we can check easily be counting the number of rows).

```
> nrow(AutoMode2)
[1] 35
```

So, now instead of 47 participants, we have 34 for our analysis.

ANALYSIS OF THE DATA

We have noticed that some participants have very low d' values. We can divide this set into two groups to explore this.

```
> split(AutoMode2, d.<0)
```

This gives group of 8 participants with scores of less than 0 and another group with the rest of the participants. As this is not balanced, we might play around with the cut-off.

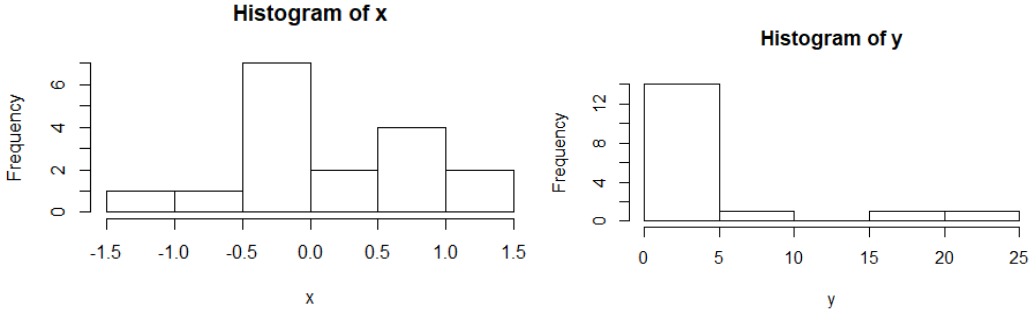
```
> split(AutoMode2, d.<1.4)
```

This splits into 2 groups of 17.

```
> high_sdt<-subset(AutoMode2,d.>1.4)
> low_sdt<-subset(AutoMode2,d.<1.4)
```

We want to compare our groups on d' (to make sure that they are distinguishable) so need to define these for comparison.

```
> x<-(low_sdt$d.)
> y<-(high_sdt$d.)
```



We can see that, while there might be a reasonable distribution for the histogram for the low_sdt, it is negatively skewed for high_sdt. However, we can see that there is some separation of the groups.

Let's explore these groups on the other measures.

```
> a<-(mean(low_sdt$Beacon))
> a
[1] 174.0556
> b<-(mean(high_sdt$Beacon))
> b
[1] 47.70588
```

The low_sdt group has much higher Beacon activation than the high_sdt group.

```
> c<-(mean(low_sdt$TpD))
> c
[1] 0.1941667
> d<-(mean(high_sdt$TpD))
> d
[1] 0.4658824
```

The low_sdt group has lower time between hitting targets than the high_sdt group.

These imply that participants with low signal detection scores were activating the beacon more often and this resulted in lower time between hitting a drone (with little discrimination being made between 'good' and 'bad' drones).