Exercises

Exercise 4: Visualising data using base R and lattice graphics

1. As in previous exercises, either create a new R script or continue with your previous R script in your RStudio Project. Again, make sure you include any metadata you feel is appropriate (title, description of task, date of creation etc) and don't forget to comment out your metadata with a `#` at the beginning of the line.

2. Download the data file *'squid1.xlsx'* from the **Data** link and save it to the `data` directory you created during exercise 1. Open this file in Microsoft Excel (or even better use an open source equivalent - LibreOffice is a good free alternative) and save it as a tab delimited file type. Name the file *'squid1.txt'* and save it to the `data` directory.

3. These data were originally collected as part of a study published in Aquatic Living Resources[1] in 2005. The aim of the study was to investigate the seasonal patterns of investment in somatic and reproductive tissues in the long finned squid *Loligo forbesi* caught in Scottish waters. Squid were caught monthly from December 1989 - July 1991 (`month` and `year` variables). After capture, each squid was given a unique `specimen` code, weighed (`weight`) and the sex determined (`sex` - only female squid are included here). The size of individuals was also measured as the dorsal mantle length (`DML`) and the mantle weight measured without internal organs (`eviscerate.weight`). The gonads were weighed (`ovary.weight`) along with the accessory reproductive organ (the nidamental gland, `nid.weight`, `nid.length`). Each individual was also assigned a categorical measure of maturity (`maturity.stage`, ranging from 1 to 5 with 1 = immature, 5 = mature). The digestive gland weight (`dig.weight`) was also recorded to assess nutritional status of the individual. If you're not familiar with squid morphology and are interested in finding out more see here.

4. Import the *'squid1.txt'* file into R using the `read.table()` function and assign it to a variable named `squid`. Use the `str()` function to display the structure of the dataset and the `summary()` function to summarise the dataset. How many observations are in this dataset? How many variables? The `year`, `month` and `maturity.stage` variables were coded as integers in the original dataset. Here we would like to code them as factors. Create a new variable for each of these variables in the `squid` dataframe and recode them as factors. Use the `str()` function again to check the coding of these new variables.

5. How many observations are there per month and year combination (hint: remember the `table()` or `xtabs()` functions?)? Don't forget to use the factor recoded versions of these variables. Do you have data for each month in each year? Which years have the most observations? (optional) Use a combination of the `xtabs()` and `ftable()` functions to create a flattened table of the number of observations for each year, maturity stage and month (aka a contingency table).

6. The humble cleveland dotplot is a great way of identifying if you have potential outliers in continuous variables. Create dotplots (using the `dotchart()` function) for the following variables; `DML`, `weight`, `nid.length` and `ovary.weight`. Do these variables contain any unusually large or small observations? Don't forget, if you prefer to create a single figure with all 4 plots you can always split your plotting device into 2 rows and 2 columns. Use the `pdf()` function to save a pdf version of your plot(s) in your `output` directory you created in Exercise 1. I have also included some alternative code in the solutions for this exercise using the `dotplot()` function from the `lattice` package.

7. It looks like the variable `nid.length` contains an unusually large value. Actually, this value is biologically implausible and clearly an error. The researchers were asked to go back and check their field notebooks and sure enough they discover a typo. It looks like a tired researcher accidentally inserted a zero by mistake when transcribing these data. We can clearly see this value is over 400 so we can use the `which()` function to identify which observation this is `which(squid$nid.length > 400)`. It looks like this is the 11^{th} observation of the `squid$nid.length` variable. Use your skill with the square brackets `[ ]` to first confirm the this is the correct value (you should get 430.2) and then change this value to 43.2. Now redo the dotchart to visualise your correction. Caution: You can only do this because you have confirmed that this is an transcribing error. You should **not** remove or change values in your data just because you feel like it or they look 'unusual'. This is scientific fraud! Also, the advantage of making this change in your R script rather than in Excel is that you now have a permanent record of the change you made and can state a clear reason for the change.

8. When exploring your data it is often useful to visualise the distribution of continuous variables. Create histograms for the variables; `DML`, `weight`, `eviscerate.weight` and `ovary.weight`. Again, its up to you if you want to plot all 4 plots separately or in the same figure. Export your plot(s) as pdf file(s). One potential problem with histograms is that the distribution of data can look quite different depending on the number of 'breaks' used. The `hist()` function does it's best to create appropriate 'breaks' for your plots (it uses the Sturges algorithm for those that want to know) but experiment with changing the number of breaks for the `DML` variable to see how the shape of the distribution changes (see the course manual for further details of how to change the breaks).

9. Scatterplots are great for visualising relationships between two continuous variables. Plot the relationship between `DML` on the x axis and `weight` on the y axis. How would you describe this relationship? Is it linear? One approach to linearising relationships is to apply a transformation on one or both variables. Try transforming the `weight` variable with either a natural log (`log()`) or square root (`sqrt()`) transformation. I suggest you create new variables in the `squid` dataframe for your transformed variables and use these variables when creating your new plots (ask if you're not sure how to do this). Which transformation best linearises this relationship? Again, save your plots as a pdf file (or try saving in your `output` directory as jpeg or png format using the `jpeg()` or `png()` functions if you feel the need for a change!).

10. When visualising differences in a continuous variable between levels of a factor (categorical variable) then a boxplot is your friend (avoid using bar plots - Google 'bar plots are evil' for more info). Create a boxplot to visualise the differences in DML at each maturity stage (don't forget to use the recoded version of this variable you created in Q4) . Include x and y axes labels in your plot. Make sure you understand the anatomy of a boxplot before moving on - please ask if you're not sure. An alternative to the boxplot is the violin plot. A violin plot is a combination of a boxplot and a kernel density plot and is great at visualising the distribution of a variable. To create a violin plot you will first need to install the `vioplot` package and make it available `library(vioplot)`. You can now use the `vioplot()` function in pretty much the same way as you created your boxplot.

11. To visualise the relationship between two continuous variables but for different levels of a factor variable you can create a conditional scatterplot. Use the `coplot()` function to plot the relationship between DML and square root transformed weight (you created this variable in Q8) for each level of maturity stage. Does the relationship between DML and weight look different for each maturity stage (suggesting an interaction)? If you prefer, you can also create a similar plot using the `xyplot()` function from the `lattice` package (don't forget to make the function available by using `library(lattice)` first).

12. To explore the relationships between multiple continuous variables it's hard to beat a pairs plot. Create a pairs plot for the variables; `DML`, `weight`, `eviscerate.weight`, `ovary.weight`, `nid.length`, and `nid.weight`. If it looks a little cramped in RStudio then click on the 'zoom' button in the plot viewer to see a larger version. One of the great things about the `pairs()` function is that you can customise what goes into each panel. Modify your pairs plot to include a histogram of the variables on the diagonal panel and include a correlation coefficient for each relationship on the upper triangle panels. Also include a smoother (wiggly line) in the lower triangle panels to help visualise these relationships. Take a look at the Introduction to R book to see how to do all this (or `?pairs`).

13. Almost every aspect of the figures you create in R is customisable. Learning how to get your plot looking just right is not only rewarding but also means that you will never have to include a plot in your paper/thesis that you're not completely happy with. When you start learning how to use R it can sometimes seem to take a lot of work to customise your plots. Don't worry, it gets easier with experience (most of the time anyway) and you will always have your code if you want to create a similar plot in the future. Use the `plot()` function to produce a scatterplot of DML on the x axis and ovary weight on the y axis (you might need to apply a transformation on the variable `ovary.weight`). Use a different colour to highlight points for each level of maturity stage. Produce a legend explaining the different colours and place it in a suitable position on the plot. Format the graph further to make it suitable for inclusion into your paper/thesis (i.e. add axes labels, change the axes scales etc).

[1] Smith JM et al (2005) Seasonal patterns of investment in reproductive and somatic tissues in the squid *Loligo forbesi*, Aquatic Living Resources. 18, 341–351.

End of Exercise 4