Exercises

Exercise: Graphical data exploration using R

1. Start RStudio on your computer. Check that you have a recent version of R (at least 4.0.0). If you haven't already done so, create a new RStudio Project (select File –> New Project on the main menu). Create the Project in a new directory by selecting 'New Directory' and then select 'New Project'. Give the Project a suitable name ('pgr_stats' maybe) in the 'Directory name:' box and choose where you would like to create this Project directory by clicking on the 'Browse' button. Finally create the project by clicking on the 'Create Project' button. This will be your main RStudio Project file and directory which you will use throughout this course. See Section 1.6 of the Introduction to R book for more information about RStudio Projects and here for a short video.

Now create a new R script inside this Project by selecting File –> New File –> R Script from the main menu (or use the shortcut button). Before you start writing any code save this script by selecting File –> Save from the main menu. Call this script 'graphical_data_exploration' or something similar. Click on the 'Files' tab in the bottom right RStudio pane to see whether your file has been saved in the correct location. Ok, at the top of almost every R script (there are very few exceptions to this!) you should include some metadata to help your collaborators (and the future you) know who wrote the script, when it was written and what the script does (amongst other things). Include this information at the top of your R script making sure that you place a # at the beginning of every line to let R know this is a comment. See Section 1.10 for a little more detail.

2. If you haven't already, download the data file *'loyn.xlsx'* from the **Data** link and save it to the `data` directory. 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 *'loyn.txt'* and also save it to the `data` directory.

3. These data are from a study originally conducted by Loyn (1987)[1] and subsequently re-analysed by Quinn and Keough (2002)[2] and Zuur et al (2009)[3]. The aim of the study was to relate bird density in 56 forest patches to a number of different environmental variables and management practices. A summary of the variables is: **ABUND**: Density of birds, Continuous response; **AREA**: Size of forest patch, Continuous explanatory; **DIST**: Distance to nearest patch, Continuous explanatory; **LDIST**: Distance to nearest larger patch, Continuous explanatory; **ALT**: Mean altitude of patch, Continuous explanatory; **YR.ISOL**: Year of isolation of clearance, Continuous explanatory; **GRAZE**: Index of livestock grazing intensity, 5 level Categorical explanatory 1= low graze, 5 = high graze. Add a description of your variables to the metadata you created previously. Clearly highlight which variable is the response variable and which variables are potential explanatory variables.

4. Import your *'loyn.txt'* file into R using the `read.table()` function and assign it to an object called `loyn`. 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 does the dataframe contain? Are there any missing values (coded as `NA`) in any variable? How is the variable `GRAZE` coded? (as a number or a string?). If you think this will cause a problem (hint: it will!), create a new variable called `FGRAZE` **in** the dataframe with `GRAZE` recoded as a factor.

5. Use the function `table()` (or `xtabs()`) to determine how many observations are in each `FGRAZE` level. See section 3.5 of the Introduction to R book to remind yourself how to do this.

6. Using the `tapply()` function what is the mean bird abundance (`ABUND`) for each level of `FGRAZE`? Can you determine the variance, the minimum and maximum for each `FGRAZE` level? Again see section 3.5 of the Introduction to R book to remind yourself how to do this.

7. Now onto some plotting action. Plot a Cleveland dotchart (Section 4.2.4) of each variable separately to assess whether there are any outliers (unusually large or small values) in the response variable (`ABUND`) or any of the explanatory variables (see Q3). Produce a Cleveland dotchart of each variable separately to assess this. If you feel in the mood, output these plots to an external PDF file.

8. If you do spot any unusual observations have a think about what you want to do with them (NOTE: do **not** just remove them without justification!). If you're unsure, be sure to speak to an instructor to discuss your options during our synchronous practical sessions. Perhaps you should apply a data transformation to see if this reduces the magnitude of any outlier. The best thing to do here is to play around with different transformations (i.e. `log`, `sqrt`) to see which one does what you want it to do. After you have applied these data transformations make sure you re-plot your dotcharts with any transformed variable to double check what the transformation is doing. Hint: a log to the base 10 transformation might help reduce the magnitude of the outliers for some of the variables.

9. Next, check if there is any potential collinearity between any of the **explanatory variables**. Remember, collinearity is a *strong* relationship between your explanatory variables. Plot these variables using the `pairs()` function (Section 4.2.5). You will need to extract your explanatory variables from the `loyn` dataframe (using `[]`) either before you use the `pairs()` function or whilst using it. Optionally, include the correlation coefficient between variables in the upper panel of the pairs plot (see section 4.2.5 of the introduction to R book for details) to help you decide whether collinearity is an issue.

10. Now that we've checked for collinearity let's assess whether there are any clear relationships between the response variable (`ABUND`) and individual explanatory variables. Use appropriate plotting functions (`plot()`, `boxplot()` etc) to visualise these relationships. Don't forget, if you have applied a data transformation to any of your variables (Q8) you will need to plot these transformed variables instead of the original variables. Also, don't forget, you can split your plotting device up to allow you to plot multiple graphs (Section 4.4) or again use a function like `pairs()` to create a multi-panel plot. Output these plots to the output directory as PDFs. Add some comments in your R code to summarise your findings.

11. One of the main aims of this study was to determine whether management practices such as grazing intensity (`GRAZE`) and size of the forest (`AREA`) affected the abundance of birds (`ABUND`). One hypothesis was that the size of the forest affected the number of birds, but this was dependent of the intensity of the grazing regime (in other words, there is an interaction between `AREA` and `GRAZE`). Use an appropriate plotting function to explore these data for such an interaction (perhaps a `coplot()` or `xyplot()` in Section 4.2.6 might be helpful?). Again, don't forget, if you have applied a data transformation to your `AREA` variable you

need to use the transformed variable in this plot not the original `AREA` variable. Output this plot as a PDF to your output directory and add some comments to your R code to describe any patterns you observe.

[1] Loyn, R. (1987). Effects of patch area and habitat on bird abundances, species numbers and tree health in fragmented Victoria forests. Nature conservation: the role of remnants of native vegetation. 65-77.

[2] Quinn, G. P., and Michael J. Keough. 2002. Experimental design and data analysis for biologists. Cambridge, UK: Cambridge University Press.

[3] Zuur, A.F., Ieno, E.N. and Elphick, C.S. (2010), A protocol for data exploration to avoid common statistical problems. Methods in Ecology and Evolution, 1: 3-14. doi:10.1111/j.2041-210X.2009.00001.x

End of Graphical data exploration using R Exercise