Week 8

We have discussed getting data into Excel and plotting it. We need to discuss manipulating and analysing data so that it suitable for plotting.

Script

Badly laid out data

We will use an existing dataset on which to do some calculations. Look at the dataset linked from Blackboard ("Badly laid out data"), which summarizes some measurements of some observed species. This dataset has some serious problems: we will look at fixing (some of) them in the next session. For now, we take the dataset as given.

Assignment

Excel is a front-end to a programming language. It has similarities to other languages like Python: there are fundamental types, like integers, floats, strings (but Excel adds dates as well). There are key differences:

- In Excel a *cell* is like a *variable*: it is a label, or location, that holds a value.
- In Excel, when a cell is associated with a calculation (a formula, or the output of a function), the it updates *immediately* when its inputs change.

To assign a value to a cell, click in the cell and type something in. For example, on the `2013` sheet, click on cell `K22` and type `Hello`. Hit `Return`. The value will be fixed. Replace this with a number, say `3`.

To assign a calculation, or formula, or function output, to a cell, we follow a similar logic. *As in Python*, assignment uses the `=` sign. So, in cell `K22` again, type `=K8`. This sets the value of cell `K22` to match that of cell `K8`. In Python language it cells the value of the variable with label `K22` to the value of the variable with label `K22`, which is `52`.

The key difference in Excel can now be shown by modifying cell `K8`. Change that value to `1` by clicking on `K8`, typing `1`, and hitting `Return`. You see the value in cell `K8` change, and also the value in cell `K22`. This is different to Python, where assignment does not form a "permanent link" between the values of two variables.

Functions and formulas

We use a similar logic to perform calculations on the data. We want to compute the sum of the weights for species `DO` collected in 2013. So, still on the `2013` sheet, we assign the value of the sum to the cell `K23`. Click on `K23`. The assignment means we need to start with an `=`. We want the sum, which in Excel is given by the function `SUM`. (Excel is not case sensitive for functions, but will convert everything to capitals). The function `SUM` takes a range of cells. As in `numpy`, this uses slicing notation `start:end`. *Unlike* `numpy`, the end point is *inclusive*. So our final command is `=SUM(K8:K19)`.

- Compute the average weight of the same data (function is `AVERAGE`).
- Compute the standard deviation of the same data (function is `STDEV`).
- Compute the sum of the weights where the weight is less than 45 (function is `SUMIF`).
- Compute the sum of the weights of the males in the same dataset (function is again `SUMIF` but uses all three arguments).

- Compute the percentage of male cases of this species (use the function `COUNT` to compute the total number of cases, and `COUNTIF` for the males).

There are a range of other useful Excel functions for logical operations (`IF` , `AND` , `OR`), for simple (and more complex) statistics (`MIN` , `MAX` , `MEDIAN` , ...), and for linking data together (`VLOOKUP` , ...).

Data organization

The general principles are outlined below. Look at the dataset linked from Blackboard ("Badly laid out data"), which summarizes some measurements of some observed species. Think about how you could, would, or should answer some questions?

- What was the heaviest observed case?
- When was the heaviest male observed?
- How much of the data is usable?

Once you have read through the data organization principles below, work out what the problems are with the existing spreadsheet. How would you modify, or add to, the spreadsheet to make answering these or similar questions more robust?

Data organization principles

This section should follow the paper of Broman and Woo (if the journal website is down again, and in case of subscription issues, also use the PeerJ link).

Communicating data via spreadsheets is easy to do, but also easy to do badly. By badly, we meant that it's easy for the process to lead to errors, or make it difficult for others to re-use the data. A range of principles can help here.

Be consistent

Say what you're going to communicate, how you're going to do it, and then stick to it.

Use good names

Think about Python variable names and use similar principles to allow re-use in other contexts. For example, no spaces in file or column names. Make sure names are meaningful.

Write dates as YYYY-MM-DD

This is the ISO format. Spreadsheet software like Excel often tries to be too clever about dates, so force this form.

Don't leave any cells empty

If there is no data, or the data for that cell was invalid in some way, say so. Use a standard form (NA, or NULL, or "-") to show this. Missing data should be clearly shown to avoid confusion.

Put one thing in a cell

Don't try to pack too much in. Don't merge cells. Don't encode information in cell formatting (colour or font).

Organize the data as a single rectangle

Subjects as rows. Variables as columns. One single header row. If the data doesn't naturally fit a rectangle, think hard before putting it in one file. It is likely that it will be more natural as multiple, linked files.

Don't make calculations in data files

Sometimes when talking about files you may want to think about "sheets" in an Excel workbook. However, it is best to have a standalone file with just the data and link to it. That way, if the data changes the analysis file does not. Cleanly separating data from analysis reduces errors from hand-modifying the data.