Exercises

Will Drysdale and Jack Davison

University of York

Set-up

- Download the .csv files from the `data_exercise` folder and put it somehwere in your documents folder.
- Make sure your packages are loaded!

```
library(openair)
library(dplyr)
#etc...
```

Day 1

Reading in data

Read the data into R, correct any mistakes, and plot a simple timeseries of NO2:NOx.

- They will need combining.
- Make sure the columns are correctly formatted (remember `class()`)
- You will need to create a new column (use `mutate()`)

Statistics

Find the mean and standard devation of your NO2:NOx column, and fit a trendline of NOx ~ NO2.

- You may need to drop NA values.
- Use `lm()` to fit the line, and `summary()` / `coef()` to pull coefficients.
- It may be a nice idea to plot a scatter with a trend-line.

Exploring `openair`

Read the openair book and have a go at some analysis.

- https://bookdown.org/david_carslaw/openair/
- Try out some different plots. Can you plot all the pollutants on one graph? What about by year?
- If you would like more data, `openair::mydata` contains some extra timeseries data that is ready to go!

Day 2

Reading in data

Read in the same data from yesterday, but using a more reproducibe workflow.

- Use lists and loops, similar to the example.
- This time, you will be column-binding rather than row-binding.

Data Manipulation

"Tidy" your data using `pivot_longer()`, then find the mean and standard deviation of each pollutant using `dplyr`. Also find the hour at which each pollutant peaks in the data frame.

- You'll not be able to work out the mean/sd *and* find the time at which the pollutants peak in the same pipeline; think about how you structure your script to avoid repetition.
- You could also use `mutate()` and `lubridate` to get an average *per year*.

Making visualisations with `ggplot2`

Use `ggplot2` to plot NO, NO2 and NOx timeseries.

- Remember you can assign colours using `aes(color = column)` and split the plot using `facet_wrap()` - what looks best?
- Is the plot too messy? Could you time average a bit to make it clearer?
- Can you plot a linear trendline using `geom_smooth()`?

Real World Exercise

Getting ready

- Download the `cape_verde.csv` file from the `data_exercise` folder.
- Complete one or more of these challenges!
- These are much more open-ended than things we've done so far and gives you a taste of "real" data science.
- If you're struggling, do work together and ask for help!

Challenge 0: Reading Data

Read in the data.

- To start, you'll need to read in the data.
- Remember that data often isn't read in perfect and ready to use - you may need the skills you've learned yesterday and today.
- Is the data ready to be used with `openair`?

Challenge 1: Diurnal Profiles

A key part of using time series data is plotting diurnal profiles. Can you plot some for, e.g., ozone?

- By the end, these should be plotted per airmass history (`Flag_name` column).
- To start, try plotting with `openair::timeAverage()` to see what they should look like.
- Can you recreate the `openair` plot using `dplyr` and `ggplot2`?

Challenge 2: Data Exploration and Validation

Real world instrument data can have some weird features - can you identify dodgy data in the VOCs?

- Tip: Use `select(contains("pp"))` to just select the VOCs (and some others).
- Calculate some summary statistics for the VOCs.
- Can you use simple visualisations to identify anomalous points? Can these be removed?

Challenge 3: Many Linear Models (& Elegant Scripting)

One of the most common statistical analyses is finding the correlation between two values. Can you find the correlation between CO2 and *every* VOC in the dataset?

- Recall the use of `lm()` to do linear modelling.
- You could do each VOC manually - but can you think of a better way?
- We haven't explicitly done this in the course, but you could use other things you've learned.

Summary

Read in the data and attempt any of Challenges 1, 2 or 3.

- (**Challenge 0:** Read in the data, fixing any issues.)
- **Challenge 1:** Plot reactive species diurnals in `ggplot2`.
- **Challenge 2:** Filter or flag anomalously high VOC data.
- **Challenge 3:** Find a concise way to correlate all VOC columns to CO2.