WACL R Training

Training for air pollution data analysis in R

Will Drysdale and Stuart Lacy

28th & 29th Nov.

University of York

Introduction

Welcome!

A course over two afternoons for beginners with R

- Introduction to R, RStudio and Programming for beginners
- Building a script; the benefits of programming over spreadsheets
- Reading, manipulating and visualising data, with tips and tricks to solve common problems
- Chance to practise skills with us on hand to help out

Approaches

- Authentic, live coding
- All course material will be made available
  - This will include all data and script files produced during this course
  - A bespoke self-teaching document will also be made available
  - Useful for post-course learning
- All material used in this course will be **entirely reproducible**
  - This means that you will be able to recreate all the outputs shown during the course (and afterwards)
- Questions are encouraged, and one of us will always be at hand to solve problems

Topics to be covered

Monday 28th November, 13:00-17:00

- Introduction to R for Air Quality Data
  - Getting familiar with R and RStudio
  - Reading and interrogating data within R
  - Introducing statistical analysis; averages and trend lines
  - Using `openair` for air quality data analysis

Tuesday 29th November, 09:00-16:00

- Further uses of R in Data Science
  - Reading and combining multiple data streams
  - Further data handling; reshaping, grouping and summarising
  - Real world data project
- Making publication standard visualisations with `ggplot2`

Who are we?

Stuart Lacy

I use R for:

- **Data** analysis from cleaning raw data to final modelling/visualisation
- **Statistical modelling / Machine learning** of data - R has lots of packages in this area
- Developing **reproducible data tools**
- Writing web app data dashboards (using `Shiny`)

README.md

`gramsper`: Remote Sensing Emission Factor Development in R

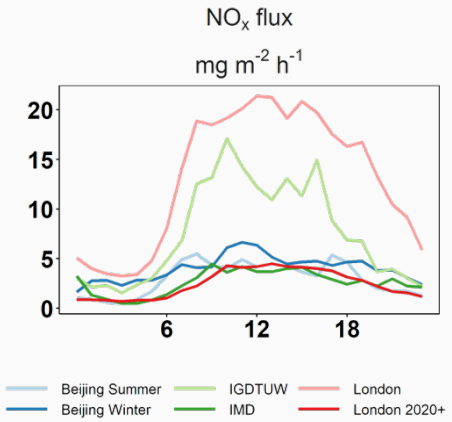
Introduction

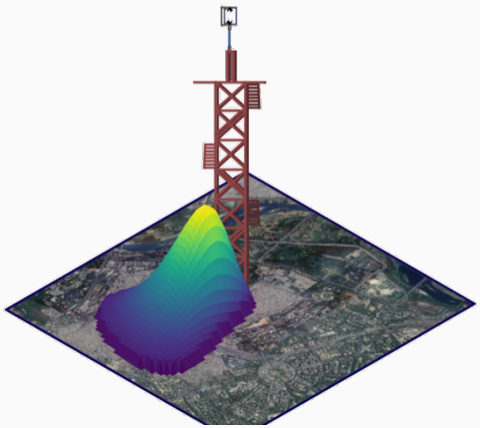
Who are we?

Will Drysdale

I use R for:

- **Eddy Covariance** - processing of high time resolution data (5 - 20 Hz) to calculate emissions using *eddy4R*
  - Perform analysis **automatically** and **reproducibly**
  - **Collaborate** with developers to add our own tools



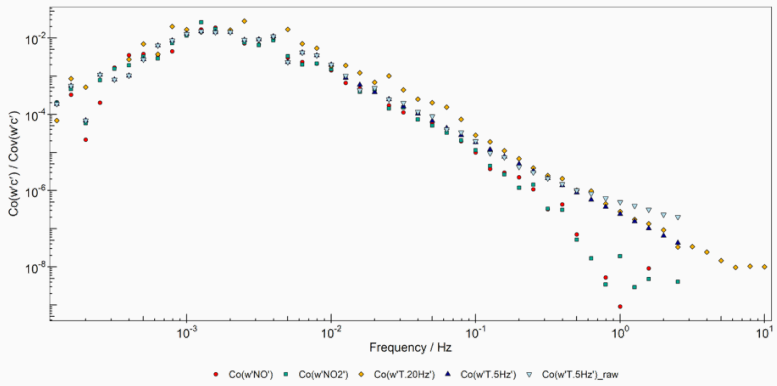


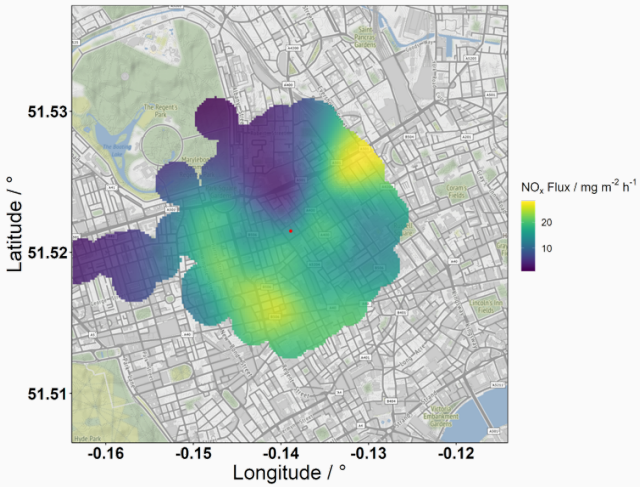
Who are we?

Will Drysdale

I also use R in many other aspects of my work:

- **Instrument data** work up
- Producing **Figures**
- **Mapping** spatial data





Who are you?

Introductions

- What is your name?
- What do you do?
- What kind of data do you use?
  - Big? Small? From the lab? Fieldwork? Modelled? Time-series? Categorical?
- What are you hoping to get out of these sessions?

Further Help

Learning R does not finish at the end of this short course

- There are many R users in WACL who are happy to help, including ourselves.
- There are lots of resources online that we'll point you to.
- WACL has a programming Slack channel for help with R & Python.
- If there is interest, we'll look to do shorter sessions on more specific problems

Setting up R

Installing R

- These slides guide you through the process of installing R and RStudio onto a desktop.
- R, RStudio and R Packages are three distinct things:
  - R is a **programming language** built for statistics.
  - RStudio is a **development environment** in which R runs. It makes R easier to use.
  - Packages are **collections of specialist tools** which make R more suited for certain tasks.
- Instructions are given for Windows computers, but should be the same for Mac and Linux; simply click on the appropriate option when they are presented.

How to Install R

1. Go to cloud.r-project.org.
2. Select **Download R for Windows**.
3. Select **base**.
4. Click **Download R 4.2.2 for Windows**.
5. **Run the .exe** to install R.

How to Install RStudio

1. Go to rstudio.com.
2. Click **Download RStudio** (top right blue button)
3. Scroll down and Select **RStudio Desktop – Open Source Edition** (Free option).
4. Scroll down to and click **Step 2: Install RStudio Desktop for Windows**
5. **Run the .exe** to install RStudio.

How to Install Packages

1. **Launch RStudio**.
2. In the console (the place you can type on the left-hand side of the screen, where the line starts with a "$>$") type the following commands, pressing enter between them:

```
install.packages("openair")
install.packages("tidyverse")
```

Exercises

Set-up

- Download the course materials from Google Drive and setup a Project
- Make sure your packages are loaded!

```
library(openair)
library(dplyr)
```

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

- **TODO: Remove this if think won't make it here, and replace with ggplot workshop contents**
- 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::timeVariation()` 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.