Programming Project 4

March 8, 2024

For your fourth and final programming project for Introductory Scientific Computing you will write code to either create a model or analyse scientific data, presenting your results as a report using a modified report structure as detailed below. There are two options provided of which you should **only complete and submit one**.

This will involve creating at least **two** documents: one (or more) containing your code and one containing your report.

To use Noteable to complete the coding part of your assessment, you should create a new Jupyter Notebook. If applicable to the option chosen you should also unzip and upload any associated data files (detailed within sections 1 and 2). After completion, your Jupyter Notebook should be downloaded from Noteable and uploaded to the submission point. For a short demo, see the video available on the Blackboard Access Python page entitled "Create new notebooks and download files". You can use an alternative tool, if you prefer, as long as your file contains Python code only and can be run using a Python interpreter (".ipynb" or ".py" file or files). Please ensure that your code is submitted in one of these two formats (".ipynb" or ".py" file or files) **otherwise the submission will not be accepted**.

The two options for this assessment are as follows:

1. **Traffic model** - build a basic traffic model incorporating both cars and buses and investigate trends in average speed and passenger throughout.
2. **Urban air quality** - compare air quality measurements inside and outside the city of London, investigating trends and considering correlation with meteorological data.

For the code document produced, this should include all the Python code used to perform the coding task including plots (either within the Jupyter notebook or referenced as external files and also submitted). The code document should be self-contained, and it should be made clear which parts of the analysis are being performed at each stage. The code should be well commented and additional details can also be added using separate markdown cells (available within Jupyter notebooks). The code document should be able to run as a whole (from top to bottom).

For your report, you should write this using a word processor of your choice and submit this alongside your code as a Word document (.doc or .docx) or as a PDF document (.pdf). The report should include background details of any scientific context, your approach and hypotheses as well as a description of your analysis, discussion and conclusions. You should also include an abstract to summarise your key results and include details of any references used. See Section 3 for these details laid out as bullet points. The report itself does not need to include explicit details of the code produced, just your overall approach and any relevant outputs but should be related to the code document submitted. The separate code submission document can be referred to if needed.

When completed, use the "Programming project 4" Blackboard submission point to upload all documents and data related to the code and report (on the Assessment, submission and feedback course content area for the unit).

1 Option: Traffic model

For this option, we would like you to build a basic model to simulate traffic flow. You will be provided with an outline for the conditions for this model and asked your model to examine relevant properties. You will not be expected to read any external data to complete this option. The outline is comprised of two parts detailed below in sections 1.1 and 1.2.

1.1 Average speed

Traffic models are an area of active research used to make decisions around planning (for example see this traffic model research introduction page from Linköping University in Sweden). They are used to consider implications for air quality and pollution, often related to throughput and congestion. To quantify levels of congestion we can consider how the numbers of vehicles on the road impacts the average speed.

Assessment (Option 1, Part 1): Build a basic traffic model and use this to evaluate the average speed of all cars for a fixed road length. See 1.1.1 for details of general conditions and model set up.

1. Start with a road length of 10 spaces and 3 cars, which should be randomly distributed. Run this model for at least 20 complete steps. Consider the following:
   - For each step, calculate and store the average speed $(0.0 - 1.0)$ across all the cars i.e. how many cars move in each step based on the total number of cars. Print out a single average for this calculated speed across all steps. How does this value align with your expectations?
2. For the initial model created, the ratio of cars to length of road was 0.3 (3 cars / road length of 10). From your model, how does the average speed change based on this ratio? Consider how to best present this relationship. As an example you could consider:
   - The highest ratio possible to give an average speed greater than 0.9 (i.e. a car moves almost every step).
   - What overall trend is present, if any.

For this question formulate your own hypothesis and use Python, along with any other library modules (such as numpy, pandas and matplotlib), to perform your analysis. Consider how you can present results from your analysis within your report.

1.1.1 Model setup

For this first part, we would like you to create a basic transport model for cars (only). This should include the following conditions:

- Each car occupies one space and each space can only contain one car.
- Each car moves forward if the space next to it along the direction of traffic flow is empty.
- The road has a fixed number of spaces but acts as a roundabout, where cars reaching the end of the road move back to the start of the road.

An example set up is shown in Figure 1. This shows a road with 10 spaces and 3 cars randomly distributed.

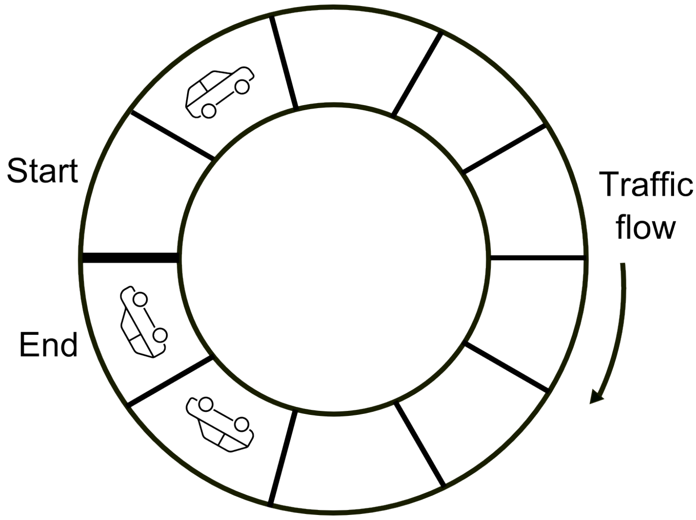

Figure 1: Consider a road of fixed length as a roundabout where a car reaching the end of the road will move to the start when possible.

To design the model the following considerations should be made:

- Setup: Assume your road is made up of a number of spaces, some of which are unoccupied and some are occupied by cars. This could be represented by values of 0 and 1, for instance.
- Initial configuration: the cars should be placed in random start positions on the road. This setup doesn't need to match to Figures 1 or 2.
- Update steps:
  - It can be assumed that the occupancy of each space on the road is updated instantaneously (i.e. all cars move at once)
  - Each step in the model should check all spaces on the road and update to the next state based on the conditions.
  - Ensure an explicit copy of the previous state is made before updating in each step (e.g. using the .copy() method for numpy ndarrays). This is to avoid state aliasing where the previous state is being updated as the checks are being made.
- Boundary conditions: Consider how to check and move a car when this reaches the end of the road and would loop back to the start.
- Output: The average speed for your model will be the number of cars which were able to move divided by the total number of cars. This means the average speed will always be between 0.0 and 1.0.

Considering that same road as in Figure 1, Figure 2 shows how this layout could be updated from an initial state to the next state. For this example, from left to right, Figure 2 shows that Car 1 is able to move as the next space is unoccupied, Car 2 cannot move because the next space already contains a car (Car 3) and Car 3 is able to move from the end of the road as the start space is unoccupied. Note that though Figure 2 shows traffic flow from left to right, this is not necessary for this model and traffic flow could be considered to run from right to left instead as long as this is consistently applied.

It may be advisable to start with the simplest case of one car and check this evolves logically before then increasing the number of cars. Additionally, to visualise each step of your model, the road layout could be stored at each step and the matplotlib imshow() function used to show this as a 2D output.

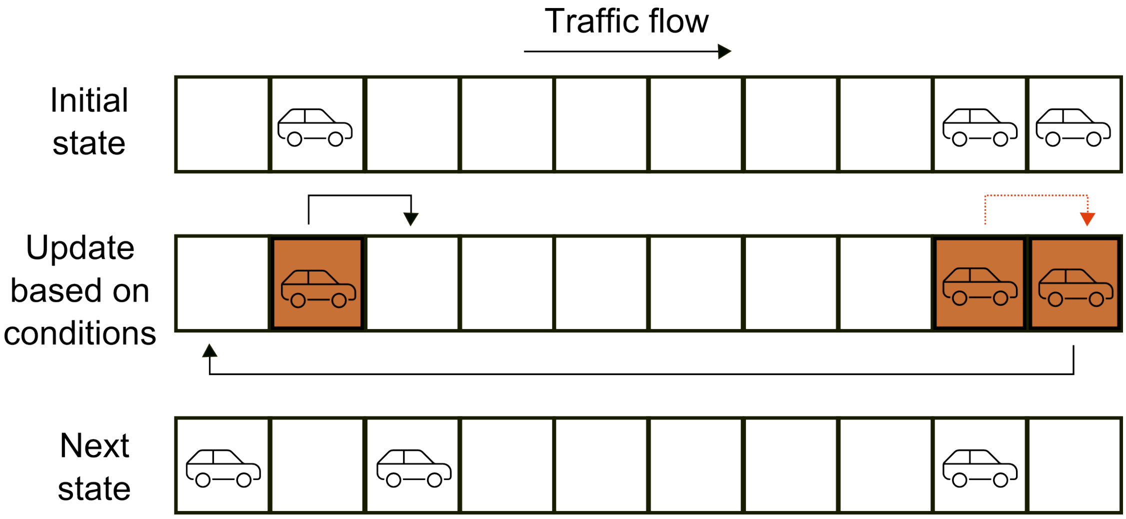

Figure 2: This figure shows the evaluation of the model for one step. The top panel shows three cars randomly distributed on a road containing ten spaces. The middle panel shows where the car wants to move and illustrates the evaluation of the condition that the next space be unoccupied. This includes evaluating whether a car can move from the end of the road back to the start (roundabout condition). The bottom panel shows the updated state ready for the next step.

1.2 Passenger throughput

To transport more people and avoid congestion by maintaining a reasonable speed we can also incorporate another mode of transport than cars: buses. In the model, buses will be slower to move (they require two spaces to be clear rather than one) but will have a higher capacity for passenger throughput.

Assessment (Option 1, Part 2): Update your basic traffic model to include buses as well as cars and use this to evaluate passenger throughput. See 1.2.1 for details of additional conditions and model set up.

1. Start with a road length of 10 spaces with 2 cars and 1 bus, which should be randomly distributed. Run this model for at least 20 complete steps. Consider the following:
   - As before, calculate and store the average speed ($0.0 - 1.0$) but now update this to include all vehicles i.e. how many vehicles move in each step based on the total number of vehicles. Again print out a single average for the speed across all steps. Has this changed from the previous value?
2. For a fixed number of passengers, investigate the average vehicle speed for varying ratios of cars to buses. Include only the minimum number of vehicles needed to transport the total number of people.
   - This should check (at least) the following parameters:
     - Total number of passengers: 500
     - Number of passengers per car: 5
     - Number of passengers per bus: 30
     - Road length of at least 100 spaces
   - As an example you could consider:
     - How much does the speed change as more buses and less cars are included in the model?
     - Minimum number of buses needed to give an average speed greater than 0.9. How does the vehicle ratio compare to the car ratio for the same speed?

For this question formulate your own hypothesis and use Python, along with any other library modules (such as numpy, pandas and matplotlib), to perform your analysis. Consider how you can present results from your analysis within your report.

1.2.1 Model setup

For this second part, we would like your model to still incorporate the conditions from section 1.1.1 for cars but also include additional conditions to incorporate buses:

- Each bus occupies one space (same as a car) and each space can contain only one vehicle (a bus or a car).
- Each bus moves forward if both of the next two spaces along the direction of traffic flow are empty. The bus will not move if only one of the next two spaces is empty.
- A bus at the end of the road can move to the start of the road, applying the same rules as above.

To update the model the following considerations should be made:

- Setup: Assume your road can be split into spaces, some of which are unoccupied, some are occupied by cars and some by buses. This could be represented by values of 0, 1 and 2, for instance.
- Initial configuration: the cars and buses should be placed in random start positions on the road.
- Update steps: Same as previous in addition to taking account of condition for buses to move.

- Boundary conditions: As well as considering how the car acts at the boundary, also consider how a bus acts. You should allow and include the check for two adjacent spaces to span from one before the end of the road back to the start to allow buses to move smoothly.
- Output: The average speed for your model will be the number of vehicles which were able to move divided by the total number of vehicles (the sum of buses and cars). This means the average speed will always be between 0.0 and 1.0.

Considering the same road as Figure 2, but this time including two cars and one bus (three vehicles), Figure 3 shows how this layout could be updated from an initial state to the next state when considering cars and buses. For this example, from left to right, Figure 3 shows that Car 1 is able to move as the next space is unoccupied, Car 2 cannot move because the next space already contains a vehicle (Bus 1) and Bus 1 is unable to move from the end of the road because one of the next two spaces (at the start) contains a car (Car 1). The bottom panel in Figure 3 also shows what will happen in the next step, including that Bus 1 will now be able to move since the next two spaces at the start of the road are unoccupied.

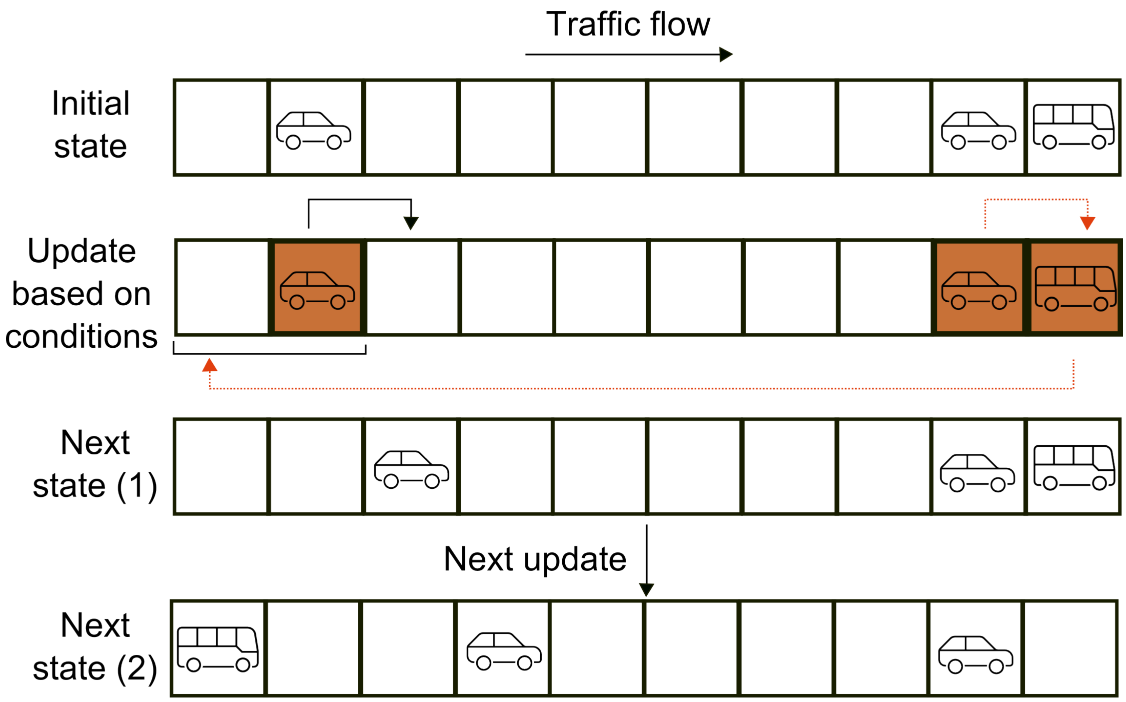

Figure 3: This figure shows the evaluation of the model for one step. The first panel shows two cars and one bus randomly distributed on a road containing ten spaces. The second panel shows where the car or bus wants to move and illustrates the evaluation of the condition that the next space / next two spaces be unoccupied. This includes evaluating whether a car or bus can move from the end of the road back to the start (roundabout condition). The third panel shows the updated state ready for the next step and the fourth panel shows the next step after this where the bus was able to move.

2 Option: Urban Air Quality

For this option we would like you to consider the impact that location and weather properties can have on air quality. Levels of air quality can be determined through measurements of various gases and particles present within the air at a given location. This analysis focuses on the measurements of extremely small airborne particles, known as particulate matter (PM), which affect air quality.

For this analysis you have been provided with three data files: two files containing air quality measurements from different locations (one within a city, referred to as urban, and one in the countryside, referred to as rural) and one file containing representative weather data for an overlapping time period. The analysis is split into two parts outlined below in sections 2.1 and 2.2.

2.1 Air quality data

The "Marylebone_AirQualityDataHourly_2018-2021_clean.csv" and "Rochester_AirQualityDataHourly_2018-2021_clean.csv" files contain measurements related to air quality taken at two sites: an urban traffic site within London and a rural background site within Rochester to the East of London (for more details of the site location type, see this description). This covers the time period from January, 2018 to February, 2021. This data has been taken (and cleaned) from the Defra (Department for Environment, Food and Rural Affairs) UK Air Data Archive (interactive map of sites).

Figure 4 shows the locations of these air quality measurements, as well as the location of the weather measurements discussed in section 2.2.

Among other properties, the data provided includes measurements of two sizes of the airborne particulate matter (PM): $< 10\mu m$ (PM_{10}) and $< 2.5\mu m$ ($PM_{2.5}$). $PM_{2.5}$ pollution is mainly produced due to combustion of fuel and wood whereas PM_{10} pollution includes dust from construction sites, industrial sources and burning. Both $PM_{2.5}$ and PM_{10} contribute to air pollution (e.g. see the Defra indicator of what constitutes high levels for each particle).

Assessment (Option 2, Part 1): Using the two data sets provided, investigate the measurements of the PM_{10} particulate matter comparing data between the urban and rural sites. Consider:

1. Are there any longer term average differences in the PM_{10} measurements between the two locations? For example you could investigate *one (or more) of*:
   - Average differences across the entire time period
   - Average difference per month for one (or more) years
2. What is the impact on the PM_{10} measurements, on average, across the hours of the day? For example you could investigate *one of*:
   - Morning (e.g. 6am) vs the evening (e.g. 6pm)
   - Differences between each hour of the day

For each of these questions formulate your own hypothesis and use Python, along with any other library modules (such as numpy, pandas and matplotlib), to perform your analysis. Consider how you can present results from your analysis within your report.

2.1.1 Data details

Provided data:

- Hourly measurements for 01/2018 - 02/2021

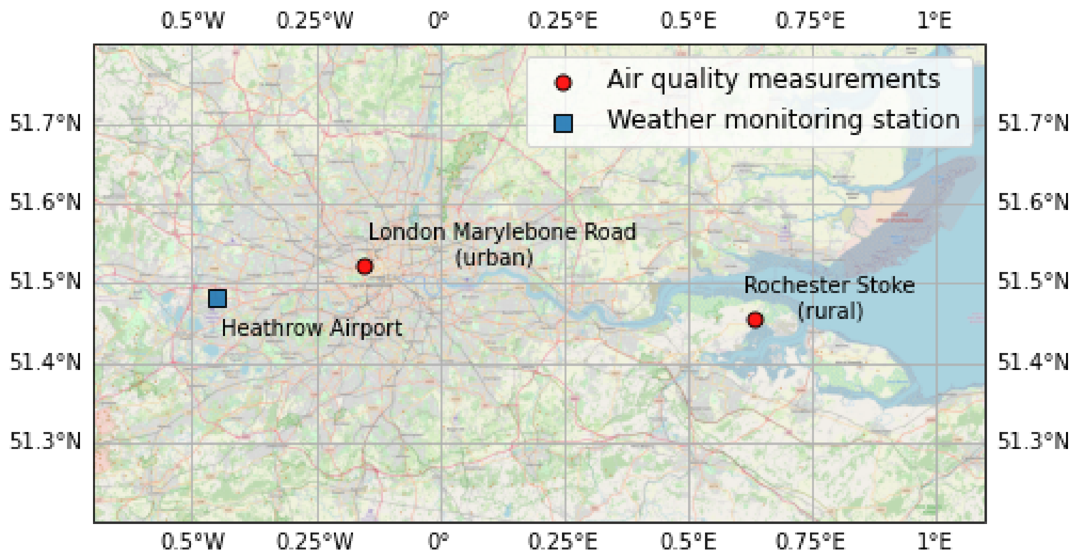

Figure 4: Positions of the air quality measurements taken (red circles) as well as the representative weather station at Heathrow airport (blue square). The London Marylebone Road site is in an urban traffic location within central London and the Rochester Stoke site is in a rural background location near the East coast. The map shows the area around London.

- London Marylebone Road (urban traffic site)
  - Filename: "Marylebone_AirQualityDataHourly_2018-2021_clean.csv"
- Rochester Stoke (rural site)
  - Filename: "Rochester_AirQualityDataHourly_2018-2021_clean.csv"
- Key columns:
  - "Date Time" contains the date and time of the observation
  - "Hour of Day" contains the hour value for each day (between 0 and 23)
  - "Day of Week" contains the day of the week value (0-6 where 0 is Monday and 6 is Sunday)
  - "PM10 particulate matter (Hourly measured)" contains the PM_{10} measurement data
  - "Status PM10" contains both the status and unit for the PM_{10} measurement data (see header within file for more details)
  - "PM2.5 particulate matter (Hourly measured)" contains the $PM_{2.5}$ measurement data
  - "Status PM2.5" contains both the status and unit for the $PM_{2.5}$ measurement data (see header within file for more details)
- **Make sure to open the file and read the header information**

2.2 Comparison to weather data

In a separate file, we have also provided data for measurements of weather (meteorological) parameters for an overlapping time period. This includes temperature, wind speed, wind direction and rainfall near London (Heathrow airport) from taken from the Meteostat web pages. The location for these measurements are also shown in Figure 4 but this has been chosen as a set of representative weather measurements for the whole region and so can be compared to the air quality data provided.

Assessment (Option 2, Part 2): Investigate the impact of meteorological conditions on the measurements of particulate matter at one location (urban or rural). Use the representative weather data provided (from London Heathrow airport) which contains data from an overlapping time period. Consider:

1. Which weather drivers could have an impact on the PM_{10} measurements made at your chosen site and what this impact is? (e.g. does higher precipitation generally mean a higher PM_{10} measurement?)
2. Is this impact similar for both PM_{10} and $PM_{2.5}$ measurements?

For each of these questions formulate your own hypothesis and use Python, along with any other library modules (such as numpy, pandas and matplotlib), to perform your analysis. Consider how you can present results from your analysis within your report.

2.2.1 Data details

Provided data:

- Hourly weather measurements for 12/1948 - 02/2021 (London Heathrow Airport)
- Filename: "Weather_data_hourly_Heathrow-Airport.csv"
- Key columns
  - "Date_Hour" contains the date and time of the observation
  - "Temperature (degrees C)" contains the temperature measurement data in degrees Centigrade (°C)
  - "Precipitation (mm)" contains the rainfall measurement data in millimetres (mm)
  - "Wind direction (degrees)" contains the wind direction data in degrees (°). Wind direction is defined in degrees between 0 and 360° such that wind coming from the North (northerly wind) will be registered as 0° (or 360°), from the East (easterly) as 90°, from the South (southerly) as 180° and from the West (westerly) as 270° with other values in between these points.
  - "Wind speed (km/h)" contains the wind speed data in kilometres per hour ($\mathrm{km\,h^{-1}}$)

3 Report

For the option you have chosen, use the analysis performed in both sections as the basis for your associated report. This should be a substantial piece of work, so you should aim for your report to be approximately 1000-2000 words or 3-6 sides of A4 (including plots). Though this is not a hard limit, where possible you should aim to provide adequate detail but to also be concise and directed.

The report itself should be clearly split into sections containing at least:

- Abstract – quick overview and summary of the key results of your analysis.
- Introduction – overall formulation of the questions being asked and any necessary background.

- Analysis and discussion – the hypotheses you are testing followed by the details of the approach and analysis performed for the different parts. Should include plots or summary tables as appropriate.
- Conclusions – any overall conclusions that can be drawn from your analysis.
- References – full details of references used in the construction of this report.