Coursework 2: Programming Task

Outline

You work as a consultant and have been asked by a client to design and implement an algorithm that estimates rainfall adjusted flow rates in a region of interest. The client would like to see a simple example algorithm that can calculate flow with random input data assuming constant rainfall and the same algorithm working with input ascii files (supplied) for rainfall and elevation (e.g. under a climate change scenarios). We have broken the problem down into five tasks.

Data provided: We provide you with a folder (coursework starter). In this there are several files:

- CourseWork1.py – a driver for the analysis that contains code required for the outputs. Running the code will not produce anything initially. But as with the learning diary, the errors produced can give you a clue as to what is required.
- You will also notice in the driver that the client has provided some named functions to perform particular tasks but that the methods have not been implemented in the modules. Use the errors and the driver method names to help you focus on where to start implementing the missing methods.
- Flow.py – module for flow. Some of which you were given in week 4 lecture material.
- Raster.py – again, you have seen most of this before.
- Rasterhandler.py – you have seen this before
- Points.py – you have seen this before
- DEM.txt – digital elevation model asci file for a part of Scotland.
- Rainfall.txt – worldclim rainfall data for the same part of Scotland as the DEM.

Expected format for submission

Upload a **word document** or **pdf** containing outputs from each of the tasks. We are assessing your ability to think about a problem in an object-orientated manner using spatial algorithms. We suggest that you provide a word document containing clearly labelled sections for each task. Within the section you should provide an explanation of what code you wrote and where you put it, with reasons for this.

The client would like to work collaboratively on this code in the future. It must be committed to a github repository with clear comments and a readme. For extra safety (their internet server is unreliable), they also require a zipped version to be submitted with the document.

Note that the code provided does not always follow good coding practice or style. Use an appropriate style for any additions and modifications you make.

Under each task heading you should also provide the requested output (answer to a questions, plots and figures). You should also supply the code (driver and modules) you used to generate your outputs, **in the github repository**, clearly labelling which commands were used for each task. In the modules we want you to highlight what you have added or changed and comments, docstrings etc should clearly explain what each method is doing. We should also be able to clearly see which bits of code were required for each task.

A note on marking criteria

As with many coding tasks there is more than one way to solve these problems. We have tried to limit the number of possible solutions by providing some of the codes required and a driver. If you change the driver as well as the modules then credit will still be awarded as long as you justify clearly the approach you have taken). You will also score credit for appropriately commented code. All plots should have appropriate captions explaining what they are and the tasks that they refer to and include all units of the data used. Finally, if you do not manage to complete a task in Python code but can supply pseudo-code for the problem credit will be awarded – again we are not just testing your coding abilities but your approach to spatial analysis.

Task 1: Calculate flow direction with constant rainfall

Calculate flow direction with constant rainfall using the flow.py module that we worked on in week 4, using either DEM.txt or a random raster. This simply requires you provide the output below.

Expected output: A flow diagram similar to the one that we worked on in week 4 titled "Network structure - before lakes" and a brief explanation (no more than 100 words) of what this is and how you produced it in Python. When we say "how you produced it" we want you to explain to the client how what the plot is showing and how the Python code produces this ie the method behind the plot. The "you" here means what your company has done, so you can assume that someone else in the company produced the code and shared it with you.

Total marks (5/100)

Task 2: Implement a method that calculates flow volume, assuming constant rainfall and plot the resultant data

Implement a method that calculates flow volume, assuming constant rainfall and plot the resultant data. We have given you the plotting code in the Coursework1.py you need to add some method(s) to the module(s) that the driver calls. These methods should recursively call each up-node and add one each time (we assume constant rainfall of 1 mm in each cell). This is a good way to test that the model and methods work before passing real data. Use either the provided DEM.txt or a random raster.

Expected output: plot of flow rate assuming constant rainfall. Brief explanation of what you did to get this (no more than 200 words).

Total marks: 10/100

Task 3. Repeat task 2 but this time using non-constant rainfall rather than assuming a constant rainfall.

Repeat task 2 but this time using non-constant rainfall rather than assuming a constant rainfall. Using the randomly generated rainfall data (held in variable *rainraster* in CourseWork1.py). You will need to think about where to add a new method to process these values. You will need to hold the values of flow volume in the flow-node while iterating. Use either DEM.txt or a random raster.

Expected output: plot of flow with variable rainfall. Brief explanation of what you did to get this (no more than 300 words).

Total marks: 25/100

Task 4. Something seems to be missing from source so far. The client asks "where is the water going? It does not seem to flow across the landscape and exit into the sea" (we assume here the sea is the edge of the DEM). The client asks if you can improve the algorithms.

Something seems to be missing from source so far. The client asks "where is the water going? It does not seem to flow across the landscape and exit into the sea" (we assume here the sea is the edge of the DEM). The client asks if you can improve the algorithms. ***Run with either real data or a random raster.*** At the moment once the water gets to a local low point or 'pit' the flow stops, this is not very realistic in a hydrological model. Join up the catchments so that the water can flow to the sea. The code calling this is provided in the driver (calling *calculateLakes()* on FlowRaster) but is commented out. You need to implement the methods in the appropriate modules, and then uncomment the call to it.

Expected output: We want you to provide Pseudo-code and the real python code for this task in an appendix. Add both to the document along with the resultant figures that come from running task 4 code in the driver. A brief explanation of what was done (no more than 300 words). Figures should be:

- Network structure (i.e. watersheds) with lakes – similar to task 1 but with lakes included.
- River flow rates with variable rainfall.
- A plot showing lake depths.

Total marks: 25/100

Task 5. Get the code to work with real data to answer the following two questions:

1. **What is the maximum flow rate in the DEM?**
2. **What is the location of the maximum flow rate?**

This is the only task that requires both the DEM.txt and the rainfall.txt files to be used. We provide you with a Digital Elevation Model (DEM.txt) and a rainfall dataset (rainfall.txt). Both are ascii files of real data. You need to check the data carefully before using it. Look over the array lecture material in week 3. You will find that the data if used in their existing format will take a long time to run so you need to think of a way that the data can be resized to run quicker. If you cannot work out how to do this then write down pseudo-code of how you think it can be done.

Expected output: Answers to the two questions above plus the commented code (in the appendix) and the resulting plots for the flow.

Total marks: 25/100

Overall Discussion

Discussion section (<750 words) on the limitations of the code that has been developed. Discussion points to consider include:

1. What else could be done to make the flow rate estimations more realistic?
2. Any additional datasets that you could use to improve the flow estimations?

Total marks: 10/100