

Geog0111 Scientific Computing

Coursework (Assessed Practical) Part B

Instructions and marking grids

P. Lewis

[p.lewis@ucl.ac.uk](mailto:p.lewis@ucl.ac.uk)

5th December 2022

URL: <https://github.com/UCL-EO/geog0111>

1. **Introduction**

1.1 Task overview

The coursework for Geog0111 Scientific Computing consists of two parts (Part A and Part B). The course is assessed entirely using these two submissions. This document describes the requirements for Part B (50% of the total marks), which is due for submission on the first Monday after the start of Term 2. Part A covers data preparation and presentation, and part B covers environmental modelling (using the data from part A and other geospatial data).

In this task, you will be writing codes to do do two tasks: (i) **snow data generation**. Develop a ***function*** to generate a gap-filled daily snow cover dataset for the Del Norte catchment in Colorado, USA for the years 2018 and 2019 using MODIS data and streamflow and temperature datasets you prepared in part A. Then, separately, demonstrate the ***running*** of the function and ***plot*** the datasets alongside one another: (ii) **snowmelt model calibration and validation**. Develop a ***function*** to **calibrate** the parameters of a snowmelt model driven by snow cover and temperature using observations of streamflow for one year of data for Del Norte. Develop ***another*** ***function*** to **validate** the model with these parameters against an independent year of data for the same area. Then demonstrate the ***running*** of these functions and ***visualise*** the results.

The main coding exercises involve building a set of Python functions, then running these, passing data between the functions, and visualising results. a set of You must provide and run the functions that you should develop in a Jupyter notebook, as well as showing results in the notebook.

1.2 Submission

The due dates for the two formally assessed pieces of coursework are:

* Part A (this piece of work): 14 Nov, 2022 (50% of final mark) - first Monday after reading week.
* Part B (the next piece of work): 10 Jan, 2023 (50% of final mark) - first day of term 2

Submission is through the usual Turnitin link on the [course Moodle page](https://moodle.ucl.ac.uk/course/view.php?id=26595).

You must develop and run the codes in a **single Jupyter notebook**, and submit the work in a single notebook as a PDF file. As usual with coursework, you must **attach a cover page** declaration.

**2 Background**

2.1 Model background

The hydrology of the Rio Grande Headwaters in Colorado, USA is snowmelt dominated. It varies considerably from year to year and may very further under a changing climate. One of the tools we use to understand monitor processes in such an area is a mathematical ('environmental') model describing the main physical processes affecting hydrology in the catchment. Such a model could help understand current behaviour and allow some prediction about possible future scenarios.

In this part of your assessment you will be using, calibrating and validating such a model that relates temperature and snow cover in the catchment to river flow.

We will use the model to describe the streamflow at the Del Norte measurement station, just on the edge of the catchment. You will use environmental (temperature) data and snow cover observations to drive the model. You will perform calibration and testing by comparing model output with observed streamflow data.



2.2.Del Norte

Further general information is available from various [websites](http://www.usclimatedata.com/climate.php?location=USCO0103), including [NOAA](http://www.ncdc.noaa.gov). You can visualise the site Del Norte 2E [here](http://mesonet.agron.iastate.edu/sites/site.php?station=CO2184&network=COCLIMATE). This is the site we will be using for river discharge data.



2.3 Model

2.3.1 Model basics

We will build a simple **mass balance model** that is capable of predicting daily streamflow at some catchment location for given temperature and catchment snow cover. This defines the *purpose* of our model (how we will use it) and describes the *model output* (daily streamflow Q(t) at some catchment location) and drivers (temperature T and catchment snow cover p):

We will need 2 datasets to run the model:

* T : temperature (C) at the Del Norte monitoring station for each day of the year
* p : Catchment snow cover (proportion)

and one dataset to calibrate and test the model:

* Q : stream flow data for each in units of megalitres/day (ML/day i.e. units of [1000000](tel:1000000)litres a day)

You should already have the datasets T and Q for the years 2016-2019 inclusive, and will need to make use of the datasets for 2018 and 2019 in this work. In the first part of this submission we will deriving the snow cover data from MODIS satellite data. We will explain this below.

As we have noted, you will be running, calibrating and testing a mass-balance snowmelt model in the Rio Grande Headwaters in Colorado, USA. In such a model, we keep track of the mass of some parameter of interest, here the snow water-equivalent (SWE), the amount of water in the snowpack for the catchment. We assume the reservoir of water is directly proportional to snow cover p, so:

SWE(t) ~= p

with a constant of proportionality relating to *snow depth and density*. The model SWE in the system at time t is a function of the SWE released from the snowpack and entering the system at time t and previous time steps.

In the model, we do not consider mechanisms of when the snow appears or disappears from any location or thinning/thickening of the snowpack. Rather, we use the snow cover as time t as a direct surrogate of the SWE at time t.

2.3.2 Water release

We need a mechanism in the model that converts from SWE(t) to dQ(t), the water flowing in to the system from the snowpack at time t. We can consider this as:

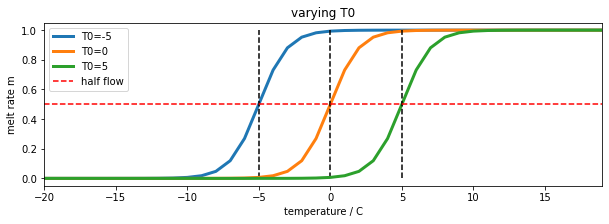
dQ(t) = k m SWE(t)

where dQ(t)is the amount of water flowing into the catchment at time t, m is a proportion of the snowpack assumed to melt at time t, and k a constant of proportionality.

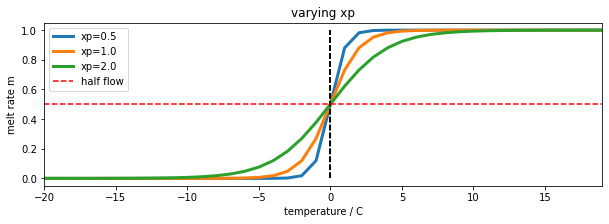
We model the rate of release of water from the snowpack as a logistic function of temperature:

m = expit([T-T0]/xp)

where expit the logistic function that we have previously used in phenology modelling. This is a form of 'soft' switch between the two states (not melting, m=0, and melting m=1). If the temperature is very much less than the threshold T0, it will remain frozen. If it is very much greater than T0, there will be an amount proportionate to SWE(t)flowing into the system on day t.



The parameter, xp (C) increases the slope of the function at T==T0 with increasing xp. So it can be used to modify the 'speed' of action of the soft switch. We will use a default value of xp=1.0. It is likely to have only a minor impact on the modelling results so we can use this assumed value of the parameter. By keeping this parameter at a fixed value, we can simplify the problem you need to solve to one involving a single parameter to model the water release.



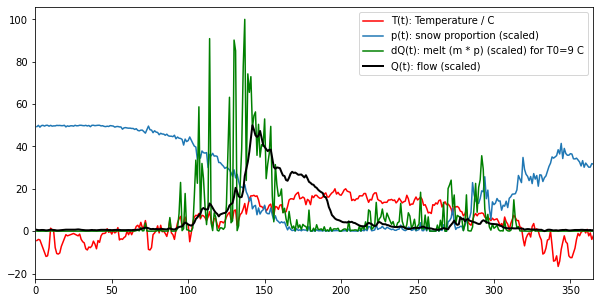
So:

dQ(t) = k expit([T-T0]/xp) SWE(t)

This is driven by T(t) and controlled by parameters T0 and to a lesser extent xp.

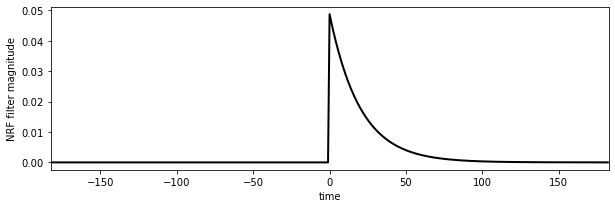
We now have a model of dQ(t)that we can compare with the other datasets.

In the figure below, we see the melt water that corresponds to a T0 of 9 degrees. It is remarkably similar to the flow data, but much noisier. We also see that it occurs some time before we see the water flow at the monitoring station. The reason for this is that there is a 'network delay' between the melt happening in the snowpack and it reaching the monitoring station. This final component of our model is a network response function (NRF) that models this delay.



2.3.3 Flow delay to the measuring stations

To be able to generate our desired model output, we now have to consider how this reservoir of water is transported to predict daily streamflow at some catchment measurement location. We do this using the concept of a Network Response Function (NRF). In this sub-model, we assume that the flow from out reservoir dQ(t)to the catchment measurement location can be characterised as a decay function after time t=0. So, some proportion of the water released is immediately transported to the station, and a lesser amount reaches the next day, and less the next day and so on. So, if we put a *pulse* of water into the system we would measure a simple decay function. This pulse response is the NRF.



The NRF is effectively a one-sided smoothing filter. It imparts a delay on the signal dQ, and smooths it. We can use a function such as a one-sided Laplace function, a one-sided exponential parameterised by a rate of decay value f (days):

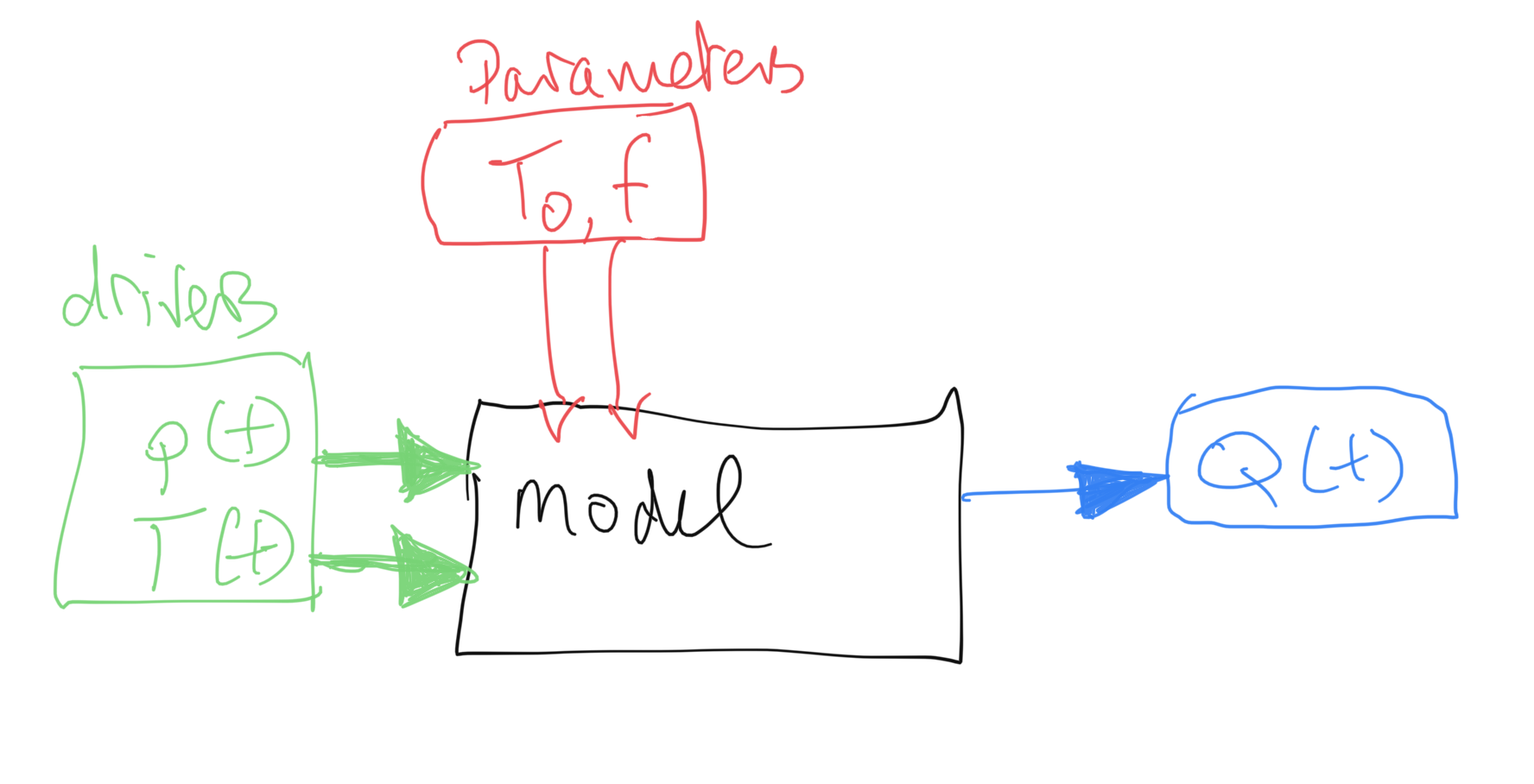
Qstation(t) = Q(t) \* exp(-t/f)

Where Qstation(t)is the modelled output, \* is the convolution operator, the flow at time t at a measuring station, Q(t)is the snowmelt entering the reservoir, and exp(-t/f) is the NRF response.

This gives an additional parameter to the model, f, so we now have 3 parameters.

2.3.4 Model

If, for the moment, we ignore the parameter xp, then we can illustrate out model as in the figure below:



The complete model has three parameters that control model behaviour:

* the threshold temperature T0 (C);
* a delay parameter  f (days) of the Network Response Function (NRF) used in routing the water from snowmelt to river flow.
* a temperature response metric xp (C)

**In this coursework, you will estimate the two main parameters for the catchment in a model calibration stage using data from 2018, and then validate them against independent data from the year 2019. This requires you to compare the predicted station streamflow** Qstation(t) **with measured values in some optimisation routine.**

**3. Coursework Detail**

3.1 Basic requirements for this submission

In Part A of the assessment (that you have already completed part), you generated and visualised environmental datasets for daily temperature (Celsius) and stream flow (ML/day) that we will be using in this part. If you believe you have made a significant error in generating these data and you don’t have anything useable for this part, you should contact the course tutor (Professor Lewis) to obtain alternative datasets. If you use these alternative data in your submission, *you must indicate that you have done so*.

**4. Coursework Instructions**

4.1 Required components:

* Snow data preparation [40%]
* Model inversion [60%]

Each of these parts has multiple components.

4.1.1 Snow data preparation

The aim of this part of the work is for you to produce datasets of snow cover for the Hydrological Unit Code (HUC code) catchment 13010001 (Rio Grande headwaters in Colorado, USA) using MODIS snow cover data. You should by now have plenty of experience of accessing and using the MODIS LAI product, and we have already come across the snow product in 030\_NASA\_MODIS\_Earthdata.

**You must provide**:

* A function that takes an integer year as an argument and returns a Pandas dataframe or dictionary containing keys for day of year (doy) and catchment mean snow cover p. It should also store the data in a CSV format file called work/snow\_cover\_YYYY.csv for year YYYY. When you run it for year, it must create and store a (gap-filled) measured snow cover dataset for that year, for each day of the year, averaged over HUC catchment 13010001. This gives 25 marks in total and is judged using the generic rubrik for functions.
* Outside of this function, in a notebook cell, you must demonstrate the running of this function for the years 2018 and 2019. You must show the size and dates of the files and created for each year, plot graphs showing and the snow cover datasets you have generated, alongside the data for T and Q that you generated in part A. This gives 15 marks (25+15 = 40%)

4.1.2 Model inversion

You should have access to datasets for

* T : temperature (C) at the Del Norte monitoring station for each day of the year
* Q : stream flow data for each in units of megalitres/day (ML/day i.e. units of 1000000 litres a day)
* p : Catchment snow cover (proportion)

for the years 2018 and 2019. These should be stored in files work/snow\_cover\_YYYY.csv (and similar) for year YYYY.

If, for any reason you have been able to produce these, discuss the matter with your course tutors before completing this section. You must NOT simply use the datasets provided with the notes without consultation.

**You must provide**:

* **A calibration function** that takes an integer year as an argument and returns a dictionary or Pandas df containing the model parameters and the goodness of fit metric at the LUT minimum in calibration, along with appropriate datasets that you can use to visualise and verify the calibration results. It should use a LUT approach to calibrate the 2-parameter snowmelt model presented above for a given year of data. It should read the driving datasets (T, Q, p) from their CSV files for year, performing any necessary filtering or gap-filling (e.g. replace NaN values).You might use the 3rd parameter as a keyword argument. This gives 35 marks.
* **A validation function** that takes an integer year\_val and the output of the calibration function for data for some other year\_cal as arguments and returns a dictionary or Pandas df containing the goodness of fit metric achieved in validation and other appropriate datasets that you can use to visualise the validation results. It should read the driving datasets (T, Q, p) from their CSV files for year year\_val, and perform any necessary filtering or gap-filling (e.g. replace NaN values). It should compare the model-predicted and measured values of Q and provide appropriate summary statistics of the goodness of fit for the validation. This gives 15 marks.
* Outside of this function, in a notebook cell, you must demonstrate the running of these functions for the years 2018 and 2019, using one as calibration and the other validation. You must show the size and dates of the files and created for each year, plot graphs showing and the snow cover datasets you have generated, alongside the data for T and Q that you generated in part A. This gives 15 marks (25+15 = 40%)

calibrate(year) to

It should output a dictionary or Pandas df containing the model parameters and the goodness of fit metric at the LUT minimum in calibration, along with appropriate datasets that you can use to visualise and verify the calibration results.

result = validate(year,calibration)

Write a function called result = validate(year,calibration)

It should read the driving datasets (T, Q, p) from their CSV files for year, and perform any necessary filtering or gap-filling (e.g. replace NaN values).

It should take as argument the calibrated model parameters (contained in calibration) and validate the model.

The main output result, a dictionary or Pandas df, will contain the goodness of fit metric achieved in validation and other appropriate datasets that you can use to visualise the validation results.

To demonstrate operation, you must:

In a Jupyter notebook cell, run the model calibration for a given year of data, returning calibration, then from calibration, print a table of the calibration parameters and associated statistics.

produce visualisations of the calibration results along with a paragraph of text describing what you are showing.

Use the other year of data to validate the model using the information in calibration, i.e. run the model for the second year with the parameters you derived from the first (calibration) year.

produce visualisations of the validation results, along with a paragraph of text describing the validation results.

* 1. Submission

The due date for Part A (this piece of work) is 10 Jan 2023 (first Monday of term). Part B represents 50% of final mark for the course. Submission is through the usual Turnitin link on the [course Moodle page](https://moodle.ucl.ac.uk/course/view.php?id=26595).

You must develop and run the codes in a single Jupyter notebook, and submit the work in a single notebook as a PDF file.

You must work individually on these tasks. If you do not, it will be treated as plagiarism. By reading [these instructions](https://github.com/UCL-EO/geog0111/blob/36e8bdbdd51fa351888f87f39109a3000d583a5a/notebooks/063_Part1_code.ipynb) for this exercise, we assume that you are aware of the UCL rules on plagiarism. You can find more information on this matter in your student handbook. If in doubt about what might constitute plagiarism, ask the course conveners.