University of Glasgow

Tuesday, 23 April 2019
2.00 pm – 3.30 pm
(1 hour 30 minutes)

DEGREES OF MSci, MEng, BEng, BSc, MA and MA (Social Sciences)

DATA FUNDAMENTALS (H)

Answer any two questions from three.

This examination paper is worth a total of 50 marks.

The use of calculators is not permitted in this examination.

INSTRUCTIONS TO INVIGILATORS
Please collect all exam question papers and exam answer scripts and retain for school to collect. Candidates must not remove exam question papers.

1. You are tasked with building a system to optimise the routing of autonomous cars in a city. As part of the initial analysis, you are asked to analyse the positions of one million pickup and drop-off points in the city captured from traditional taxis. You receive the data as an 1000000x4 array of float64 called `taxi_data`, with four columns: `(time, taxi_id, x_grid, y_grid)`.

(a) (i) Your manager asks you to summarise the *location* of pickups and dropoffs as a *multivariate normal distribution*. Explain how you would parameterise a normal distribution to model these locations, including a description of the array shape of any parameters that the distribution would have.

[4]

(ii) Explain briefly in words how would you might estimate those parameters from the data you have, for example, suggesting NumPy functions that would compute these estimates. You do not need to write any code.

[2]

(iii) The city is located on a long, thin island. Explain how the eigendecomposition could be used on the parameters estimated above to identify the major axis of this island; i.e. a unit vector pointing in the direction in which the island is longest. Draw a simple sketch to show: an island; some data points; the estimated normal distribution; the relevant eigenvectors.

[8]

(b) Points are recorded at with timestamps representing whatever time the taxi happened to make the dropoff or pickup. You are asked to reconstruct trajectories that individual taxis made in the city, assuming that each taxi moves in a straight line from each pickup to dropoff.

(i) Write a NumPy expression which will select all of the taxis with a specific id number `id_number` from the dataset and store it in a variable `taxi_path`. The column representing the `taxi_id` should be removed, such that the output array has columns `(time, x_grid, y_grid)`.

[4]

(ii) Each taxi location can be represented as a 2D vector $\mathbf{x_i} = [x_{grid}, y_{grid}]$ at some time point t_i. Assume you know:

- a pair of *consecutive* location vectors $\mathbf{x_0}$ and $\mathbf{x_1}$ at two times t_0 and t_1;
- and a third time t_j, such that $t_0 \leq t_j \leq t_1$;

Using elementary vector operations, give a mathematical formula to compute $\mathbf{x_j}$, a location vector that represents an interpolated location at

time t_j, assuming a straight line between the two consecutive location vectors.

[4]

(iii) Your boss suggests taking the raw data and applying a moving average to smooth out the paths of the taxis. Explain why a moving average should *not* be applied directly to this data, and explain briefly how to preprocess it such that it *would* make sense to apply a moving average to it.

[3]

2. You are tasked with modelling the behaviour of bees. The scientific team are researching food sources bees prefer to optimise honey production.

(a) A team of junior scientists have been asked to watch bees flying to food sources. They have counted how many times each food source has been visited, and collated the results in the table below.

- Elderflower: 3010
- Synthetic sugar-water: 81
- Foxgloves: 1237
- Heather: 9611

The scientists wish to *simulate* the behaviour of the bees. As a "zeroth-order" approach, they assume that each bee chooses a food source independently with some probability $P(X = x)$, where X is a random variable representing the food chosen.

(i) Explain how to compute the empirical probability mass function from the count data. You do not need to do the calculation.

[2]

(ii) Assuming that you have a random number generator capable of generating uniformly distributed floating point numbers in the range $[0, 1]$, describe an algorithm that would draw samples from this empirical mass function.

[3]

(b) The scientists believe that the food preferences are influenced by their feeding history. For example, a bee that has just visited the sugar-water is very unlikely to return to it immediately; while the opposite is true of heather, which has many immediate return visitors.

The scientists collect consecutive pairs of food sources visited by individual bees. They capture these as pairs of counts (x_{i-1}, x_i); the number of times any bee visited food source x_{i-1} then food source x_i. From this they compute the probability mass function for the joint distribution of two random variables $P(X_{i-1}, X_i)$

(i) From this joint probability distribution *alone*, give an equation showing how the conditional probability $P(X_i|X_{i-1})$ can be computed.

[3]

(ii) This type of model is a Markov model; it models the selection of food sources as a Markov process. State the key assumption of a Markov process.

[1]

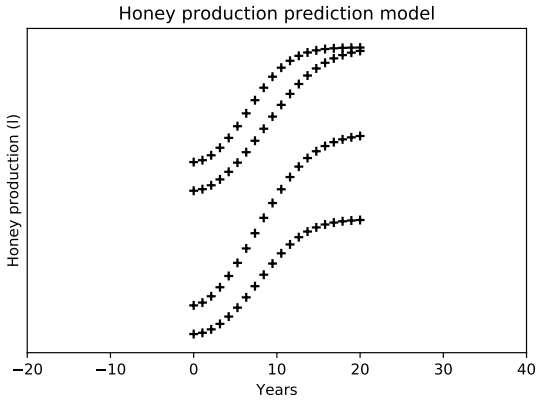
(iii) The scientists buy two commercial bee simulators to model the behaviour of their bees. They wish to test which of the two simulators is most

compatible with their Markov model. Explain how they could do this quantitatively in terms of *likelihood*, giving pseudo-code for the process you would recommend.

[8]

(c) (i) From one of the simulators, the scientists have produced the plot below, showing projected honey production over the next two years. The plot is supposed to show the production (in litres) as a function of time (in years), averaged over many runs of the randomised simulation for each of the four food sources. Criticise this graph, and redraw a sketch that improves each of the points you have identified.

[5]



(ii) The scientists believe that the predicted honey production as a function of temperature is governed by an equation of the form:

$$y = t^k$$

where y is the production (litres), t is the temperature (degrees Celsius), and k is an unknown constant.
Given a collection of measured production rates $y_1, y_2, \ldots$ and matching measured temperatures $t_1, t_2, \ldots$ explain how the scientists could *visually* establish whether this relationship was likely to be true and how the value of k might be eyeballed.

[3]

3. You are asked to design a software floating point algorithm for a new class of embedded processors. This will use SIMD to accelerate the computation of vectorised code that will run on a drone.

(a) To demonstrate the power of the new vectorised code, you are asked to write a short program, which takes three 1D arrays of the same length `a`, `b` `c`. `a` and `b` consist of finite floating point numbers, and `c` is a sequence of binary values, either 0 or 1. Write a *vectorised* NumPy expression that implements the following algorithm:

```
z = 0
for i in range(len(a)):
  if c[i]==0:
    z += a[i] ** 2 + b[i]
  else:
    z -= a[i]
```

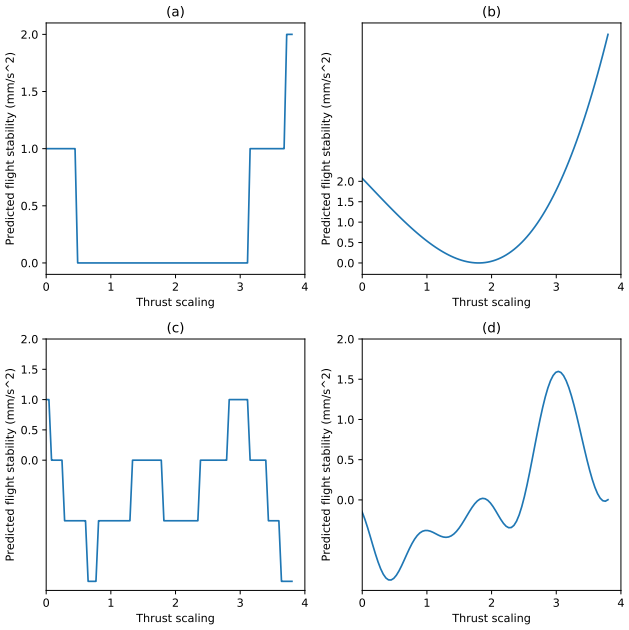
[4]

(b) A very basic prototype for a floating point addition unit has been implemented that computes IEEE754 addition of two values: `c = a + b`. This correctly adds two IEEE754 floating point numbers in simple cases, but does not handle any special values. Comment on any special values (i.e. values of `a` and `b`) that need specific handling and how they should be dealt with.

[5]

(c) (i) The floating point software will be used to perform online optimisation of a drone's flight controller. Four versions of the software have been simulated. For each version, a graph is shown below that illustrates the objective function (flight stability) as a function of an adjustable parameter (thrust scaling). Describe each graph in terms of continuity and convexity, and argue which of these curves, if any, you would think would be more amenable to numerical optimisation.

[6]



(ii) The optimisation team decide to use a gradient descent based method to optimise the setting of thrust scaling. The gradient descent algorithm is given by the update:

$$\theta_{i+1} = \theta_i - \delta \nabla L(\theta)$$

Given an objective function $L(\theta)$ which has a gradient defined everywhere, state conditions under which gradient descent will approach a local minima, and any practical requirements to implement gradient descent efficiently.

[3]

(iii) The optimisation team find that ordinary gradient descent is not working effectively, despite attempts to adjust the step size in the optimisation. Suggest **two** metaheuristics for gradient descent that might improve performance in this context, and discuss what features of the objective function they would mitigate.

[4]

(iv) The thrust scaling parameter can only range in between 0.3 and 1.9. Assume that ordinary gradient descent is used to optimise the controller. Discuss the issues with enforcing this range limitation, suggest how this range limitation could be implemented, and discuss any issues that must be considered.

[3]

END OF QUESTION PAPER