University of Glasgow

DayOfWeek, DayOfMonth Month 2019
XX.XX am/pm – XX.XX am/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]

Solution: A normal distribution estimated from 2D grid positions would have a 2 element (1) mean vector (1) and a (2,2) (1) element covariance matrix (1).

(ii) Explain briefly in words how would you might estimate those parameters from the data you have. You do not need to write any code. [2]

Solution: The mean could be estimated via the sample mean, and the covariance matrix via the sample covariance matrix, or any other words to that effect.

(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 eigenvector. [8]

Solution: The eigendecomposition of the covariance matrix (1) would reveal the eigenvectors/principal components (1). The eigenvector with largest absolute corresponding eigenvector (1) would correspond to the major axis/longest direction (1). Diagram should have: an island shape (1); some data points as dots or similar markers (1); a normal distribution as an ellipse or nested ellipses (1); the eigenvectors as a cross aligned with the ellipse (1)

(b) These 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]

Solution: `taxi_path = taxi_data((taxi_data(:,1) == id_number), (0,2,3))` (1 mark for indexing by boolean)(1 mark for testing correct column) (1 mark for removing column 1)(1 mark for storing in correct variable) There are many ways to do this, including splitting over multiple statements. These should receive full credit even if they don't qualify as an "expression"

(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]

Solution: $a = (t_j - t_0)/(t_1 - t_0)$ (2 marks for interpolation variable) $\mathbf{x_j} = a * \mathbf{x_1} + (1-a) * \mathbf{x_0}$ (2 marks for interpolating vectors correctly) can be combined into a single formula, or with a=(1-a)

(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]

Solution: The data is not regularly sampled (1) and therefore cannot be averaged as if timesteps were equally spaced (1). The data must be resampled at some constant samping rate (gridded) (1) before it can be directly filtered.

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 by a bee, 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 in their optimisation. 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]

Solution: Each count can be normalised (1 mark) by dividing by the sum of all counts (1 mark), which results in a valid probability distribution; alternatively the formula $P(X = x) = \frac{n_x}{N}$ can be stated.

(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 simulate the bee behaviour by drawing samples from this empirical mass function. [3]

Solution: A uniform number would be drawn (1 mark). The interval $[0, 1]$ would be divided into consecutive bins (in arbitrary order) with each bin equal to the width of the corresponding probability (1 mark) (alternatively could state that the probabilities are accumulated during the iteration; this is equivalent) . The index of the bin the random number lies in will be the outcome drawn (1 mark) (there are many alternative ways of expressing this basic algorithm)

(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]

Solution:

$$P(X_i|X_{i-1}) = \frac{P(X_{i-1}, X_i)}{\sum_{X_i} P(X_i, X_{i-1})}$$

(1 mark for fraction) (1 mark for marginalisation) (1 mark for correctly combining)

(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]

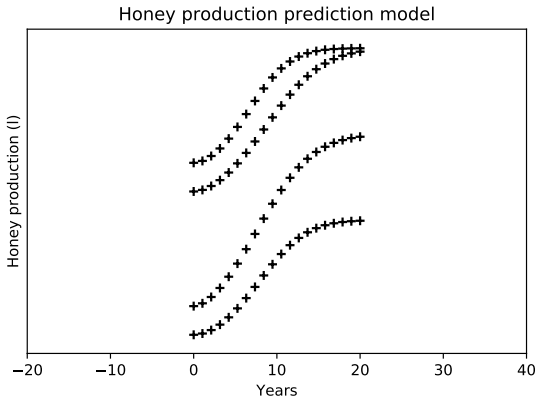
Solution: A Markov depends *only* on the state in the previous time step.

(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]

Solution: Assuming we have estimated the conditional probabilities $P(X_i|X_{i-1})$ from the count data (1 mark for needing to know PMF) for each sequence (1 mark for testing both sequences) For each prev x_{i-1}, next x_i element of the sequence (1 mark for iterating over sequential pairs) compute the log-likelihood of observing that transition given the conditional probabilities (1 mark for likelihood, 1 mark for using log-likelihood) $log(L(x_{i-1}, x_i))$ sum the log likelihoods (2 marks for summing log-likelihoods or 1 mark if product of raw likelihood) compare the log likelihoods; the one with larger (i.e. closer to 0) value represents a model more likely to be compatible with the data (2 marks for correct comparison of likelihoods)

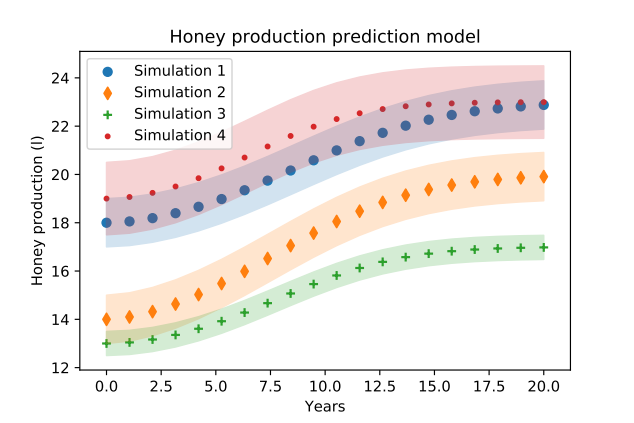
(c) (i) From one of the commercial simulators, the scientists have produced the plot below, showing the projected honey production over the next two years. The graph is supposed to show the honey production (in litres) as a function of time (in years), averaged over many runs of the randomised simulation for each of the four food sources. Criticise this graph, and redraw a sketch that improves each of the points you have identified. [5]



Solution: For each of the following: 1/2 mark if problem is identified, 1/2 mark if fixed in sketch

- (1 mark) markers are indistinguishable
- (1 mark) no legend present
- (1 mark) no representation of uncertainty/errors from averaging
- (1 mark) bad scale on x axis

- (1 mark) no ticks on y axis



(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]

Solution: A plot on a log-log scale (1 mark) will be a straight line if the equation is of this form (1 mark). The value of k will be given by the gradient of this line on the log-log scale.

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` are any 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]

Solution:

```
z = np.sum(np.where(c==0, a**2+b**2, -a))
```

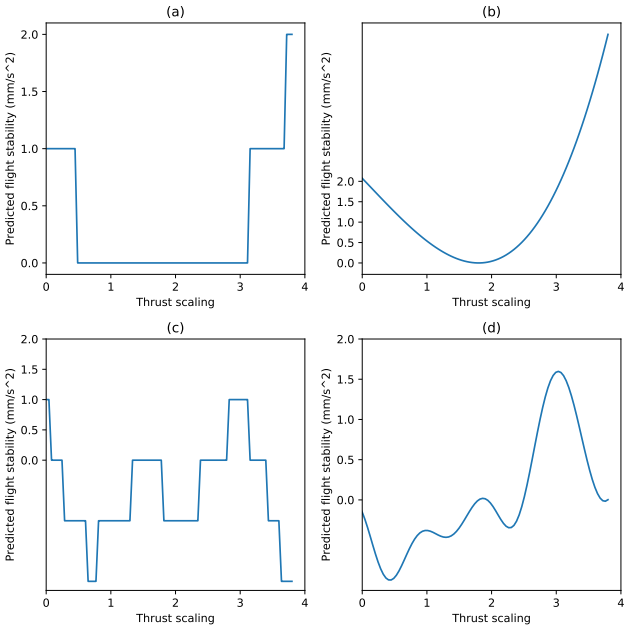
(1 mark for sum)(1 mark for where) (1 mark for condition)(1 mark for correct arithmetic on a and b) Many other versions acceptable, as long as no explicit iteration or branching involved.

(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 together two IEEE754 floating point numbers in simple cases, but does not handle any special values. You are asked to comment on any special values (i.e. values of `a` and `b`) that need specific handling and how they should be dealt with. [5]

Solution: Any of these for up to five marks:

Special values to watch out for: 0.0 and -0.0 are equal in value but not representation (1 mark); sign must be propagated correctly in 0+0 operations (1 mark) inf and -inf must evaluate to inf for finite operands (1 mark), but to nan (or raise invalid operation) for opposite signed infs (1 mark), correct signed inf for same signed infs (1 mark) nan results in nan or invalid operation exception for any operand (1 mark) Also acceptable to mention overflow (1 mark) when a+b greater than representable range.

(c) (i) The floating point software will be used to perform online optimisation of a drone's flight controller. Four different 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]



Solution: (a) is convex but has (apparently) discontinuous first derivative (1/2 mark for each) (b) is continuous and convex (1/2 mark for each) (c) is discontinuous in first derivative and non-convex (1/2 mark for each) (d) is continuous and non-convex (1/2 mark for each)
(b) is preferable; if it is truly convex, a single global minimum is preferable (1 mark), and there are fast algorithms for convex problems (1 mark). Continuous (first) derivatives mean that gradient-based (first order) methods are effective (1 mark), which can be much faster than zeroth order methods (1 mark)

(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]

Solution: Gradient descent will approach local minima if: the function is Lipschitz continuous (1 mark) and the value of δ is adjusted appropriately (1 mark). It is practical to implement if the exact derivatives of $L(\theta)$ can be computed for any θ(1 mark) e.g. by differentiable programming.

(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 perfor-

mance in this context, and discuss what features of the objective function they would mitigate. [4]

Solution:

- *momentum* (1 mark) may help if the gradient descent is getting stuck in shallow minima or is inefficient where the objective function is very flat / trapped by saddle points(1 mark)
- *random restart* (1 mark) will help explore a wider range of the parameter space if there are many local minima (1 mark)

Any other reasonable metaheuristics can be given, but **not** stochastic gradient descent as the problem is not decomposable into a sum of smaller problems.

(iv) The thrust scaling parameter can only range in between 0.3 and 1.9. Assuming that ordinary gradient descent is used to optimise the controller, discuss the issue with enforcing this in the optimisation, suggest how this range limitation could be implemented and any issues that must be considered. [3]

Solution: Gradient descent is an unconstrained optimiser and cannot directly incorporate constraints (1 mark) Instead, a penalty function may be used (1 mark) to increase the cost of solutions that exceed these bounds. These will necessarily be "soft constraints" (1 mark) if they are to be differentiable and the shaping/scaling of the constraints must be chosen (alternatively for 1 mark).

Question	Points	Score
1	25	
2	25	
3	25	
Total:	75	