How to answer questions

If you understand the mathematical principles, you can answer most projects very briefly. Try to think around the question ('Why are they asking this?') to understand the point.

(Just because it's coursework doesn't mean you have to produce a literary dissertation!)

- If a question is precise:
  your answer should be precise.
- If a question can be interpreted broadly:
  you should explore the problem space, considering a range of alternatives and possibilities, on your own.
- If you use simulation:
  choose appropriate parameters to vary; investigate parameter space systematically; repeat simulation, to estimate reliability; plot well-designed graphics.

What to write up

Exercise discretion and selection in what you choose to report. You will not get good marks for a large table of undigested numbers.

You will gain quality marks by thinking around the problem, investigating alternative approaches, analysing them, and writing them up concisely.

Who to ask for help

The questions have (for the most part) been carefully phrased.

- If there's an inconsistency,
  email the help-desk.
- If the problem setup is ambiguous,
  email the help-desk.
- If you don't understand the language,
  ask a friend.
- If you don't understand the maths,
  ask a colleague or supervisor.

9.2 Value iteration *10 units.*

Recommend IIB Optimization and Control (L).

Example. I run a call center, and I have to decide staffing levels. I can bring in extra staff at any time, though (because of union issues) once they start a shift I have to pay them for the full length of the shift.

New calls arrive according to a Poisson process. If a caller is on hold for too long, she gets impatient and hangs up, and I loose a customer; if she is answered promptly, she buys the product I'm selling.

I want to maximize my long-run average profit. What policy should I use to set staffing levels? How does it depend on the number of callers currently on hold? How can I prove the policy is optimal?

Use **dynamic programming** to solve the problem over a finite horizon. When the horizon is far enough away, the answer should approximate the infinite-horizon solution, letting us compute maximum long-run average profit.

Programming is reasonably simple; understanding the infinite-horizon theory is more challenging.

9.3 Proteins comparison in bioinformatics *8 units.*

Uses elementary probability and discrete maths.

What is the difference between THATCHER and BLAIR?

THATCHER	
BLATCHER	change, change
BLAR	delete
BLAIR	insert

Each of these edits has a cost. What is the cheapest edit sequence?

An example of **dynamic programming**. Start with (R,R), then (ER,R), then (R,IR), then (ER,IR). Work backwards to (THATCHER,BLAIR).

Requires some algorithmic/programming skill, to keep track of all the possibilities.

9.4 Option pricing in mathematical finance *6 units.*

Self-contained, but IIB Optimization & Control (L) or II Stochastic Financial Models (L) may help.

Consider a stock whose price performs a random walk:

$$p_{t+1} = \begin{cases} p_t + d & \text{with probability } u \\ p_t - d & \text{with probability } 1-u \end{cases}$$

A *European call option* gives me the right to buy 1 stock at time T at price s.

At time T, the value of the option is

$$v_T(p_T) = (p_T - s)^+.$$

How much should I pay for the option at time 0? I should pay the expected return,

$$v_0(p_0) = \mathbb{E}\Big(v_T(p_T)|p_0\Big).$$

Dynamic programming. Calculate all $v_T(p_T)$, then all $v_{T-1}(p_{T-1})$, etc. Work back to $v_0(p_0)$.

Programming is reasonably simple. Questions are not deep, but require some care.

10.3 Bootstrap estimation of standard error *5 units.*

Based on IB Statistics.

We have sampled $X_1, \ldots, X_n$, from N(μ, σ^2), and we want to estimate μ. We use the estimator

$$T = \frac{X_1 + \cdots + X_n}{n}$$

Theory says that $T \sim$ N$(\mu, \sigma^2/n)$, and we can use this fact to test a hypothesis like "$\mu > 0$".

What if the X_i come from some distribution F which we don't know? We could take many samples from F, compute T for each sample, plot a histogram of values of T, hence test the hypothesis.

What if we don't know F and can't sample from it? Then sample from $X_1, \ldots, X_n$, as an approximation to sampling from F. This is the **bootstrap**.

Simple programming. Some statistical and probabilistic thought required to answer the questions.

10.9 Markov Chain Monte Carlo *6 units.*

Requires IB Statistics, and elementary Markov Chains (M).

Suppose we want to sample from a Bayesian posterior distribution:

$$\pi(\theta_1, \theta_2 | x) \propto \pi(\theta_1, \theta_2) f(x | \theta_1, \theta_2).$$

It may be hard to compute the posterior distribution exactly, if the parameter space is large.

Often we know $\pi(\theta_1 | \theta_2, x)$ and $\pi(\theta_2 | \theta_1, x)$. An approximate way to sample from the posterior distribution is: start with θ_1, θ_2 generated from $\pi(\theta_1, \theta_2)$, then

- generate a new θ_1 from $\pi(\theta_1 | \theta_2, x)$
- generate a new θ_2 from $\pi(\theta_2 | \theta_1, z)$
- Repeat until their distribution stabilizes

Programming is simple. There are questions about speed of convergence: be sure to answer them properly!

10.11 Data analysis *7 units.*

Based on IB Statistics.

You are given raw data and required to

- formulate hypotheses
- manipulate data (e.g. Excel, R)
- test hypotheses
- draw conclusions

Data set is a list of countries, together with their number of Olympic medals, population, GDP, size of state, level of democracy.

What is the data trying to tell you? Can you find the best way to let the data display your conclusions? Can you spot trends, patterns, relationships, anomalies?

19.1 Random codes *5 units.*

Requires IIA Coding & Cryptography (E)
or IIB Information Theory (M)

An n-bit code is a subset of $\{0,1\}^n$. The elements are called *codewords*. We want the codewords to be dissimilar (to be robust to bit error), yet we want there to be lots of them (to maximize the information rate).

$\{000, 111\}$ is robust but has low information rate.

$\{000, 001, 010, 011, 100, 101, 110, 111\}$ is not robust but has high information rate.

What is the tradeoff? Investigate by generating random codes.

Programs are reasonably simple, though there are some tricks needed to make them efficient. Questions involve intelligent simulation & plotting, and a little bookwork.

19.2 Information content of natural language *4 units.*

Requires IIB Information Theorem (M).

Consider a message with symbols $c_1, \ldots, c_n$, with frequencies $p_1, \ldots, p_n$. The *entropy* is

$$h = -\sum_{i=1}^{n} p_i \log p_i.$$

It measures the information content, i.e. how compressible the message is.

Take individual letters to be symbols. How compressible is a block of English text?

Take letter-pairs to be symbols. Now how compressible is it?

Needs careful programming.

20.2 Importance sampling and fast simulation *5 units.*

Requires elementary Markov chains (M).

Consider a queue. There is a random stream of customers, on average λ customers per second. The server can serve on average ν customers per second. What is

$$\mathbb{P}(\text{queue} > B)?$$

If $\lambda \ll \nu$, the probability is small, and it takes a long time to estimate small probabilities.

Instead, simulate the system at $\lambda' > \lambda$, measure

$$\mathbb{P}'(\text{queue} > B),$$

and use (elementary probability) theory to derive

$$\mathbb{P}(\text{queue} > B).$$

Simple programming, simple theory, simple technique— tremendously useful when it can be applied.

20.5 Percolation and the invasion process *7 units.*

Assumes basic probability. Some slight help from IIB Applied Probability (L). Covered more in III Percolation.

Consider an infinite square lattice. Let each edge of the lattice be present with probability p, absent otherwise, all edges independently. What is

$$\mathbb{P}(\text{there is an infinite connected component})?$$

Simulation. Requires some programming sophistication (data structures, pointers) and also clever thinking to find efficient techniques. Questions are somewhat challenging.