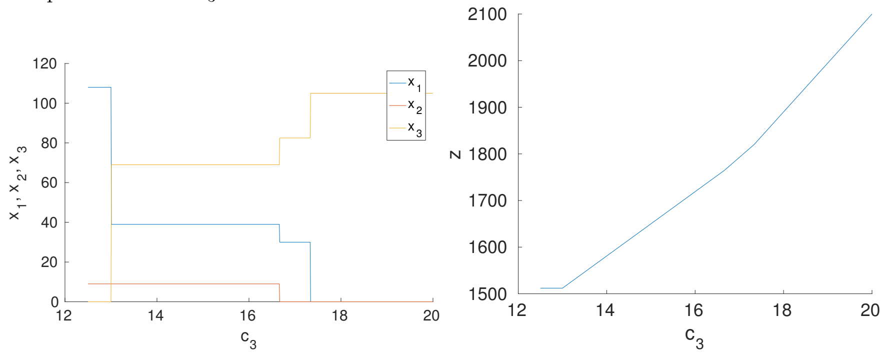
Assignment 4: Solutions

TOTAL MARKS: 20

1. (a) The following two plots show the optimal values of $\mathbf{x}$ and the total profit z as a function of the profit on a bed c_3.



[N.B. Results may be displayed in alternative forms as long as the information being displayed is clear. On the other hand, marks may be deducted for bad presentation, e.g. fonts that are too small or unlabelled axes.]

[4 marks]

(b) The solution $\mathbf{x}$ remains the same while the profit on a bed c_3 ranges between approximately 16.6 and 17.3. Outside of this range, the optimal solution changes quite significantly.

[2 marks]

[NB: note that z changes continuously in this region though!]

(c) The problem appears quite sensitive to c_3 in this range. [1 mark]

However, if the predicted profit (on a bed) were closer to 15, we would have had a much less sensitive solution.

2. Imagine a sports competition, which starts with a round-robin phase where every team plays every other team once.

There are 5 teams, and 2 playing grounds. Determine an Integer Program to find a schedule of games with the minimum number of possible rounds.

(a) The minimum number of rounds (assuming it can be attained) is calculated by noting that if each team plays each other team, there will be $n(n-1)/2$ games, and if there are two grounds, only two can be played in a round, so there will be $K = 5 * 4/4 = 5$ rounds.

We don't need the variables x_{iik} as they must always be zero, and $x_{ijk} = x_{jik}$ always, so we don't need both. So the number of variables for each round is $n(n-1)/2$, which is 10.

So the total number of variables is $5 \times 10 = 50$. [1 marks]

[N.B. It is possible to either include extra (unnecessary) variables and obtain a correct solution, but other components of your solution must be consistent.]

(b) Write mathematical constraints for the problem, *i.e.,* translate the following into mathematics.

- Assume K rounds (where we take K to be a maximum possible), then

$$\sum_{k=1}^{K} x_{ijk} = 1, \quad \forall i, j = 1, 2, \ldots, n, \text{ with } j < i.$$

[1 mark]

- No more than two games per round

$$\sum_{i=1}^{n} \sum_{j=1}^{i-1} x_{ijk} \leq 2, \quad \forall k = 1, 2, \ldots, K.$$

[1 mark]

- No team can play more than once in a particular round.

$$\sum_{j=1}^{n} x_{ijk} + \sum_{j=1}^{n} x_{jik} \leq 1, \quad \forall i = 1, 2, \ldots, n, \text{ and } \forall k = 1, 2, \ldots, K$$

[1 mark]

(c) Take the weighting for each round to be c_k, which is an increasing sequence of values to weight against using later games, then our objective is

$$\min \sum_{k=1}^{K} c_k \sum_{i=1}^{n} \sum_{j=1}^{i-1} x_{ijk}.$$

[1 marks]

Notes:

1. The above is specified assuming that we have set all $x_{ijk} = 0$ for $j \geq i$. We could have used assymetry here, however, to do things like indicate who had the home ground in each match, but that would require additional steps, for instance to indicate which grounds are "home" grounds, and potentially additionally constraints, like "no team has to play two away games in a row".
2. You do not have to solve the problem, yet. The solutions won't be unique (for instance, you should be able to see we can always permute the teams), but one solution found by AMPL is:

round	matches
1	(1,4) (2,5)
2	(1,3) (2,4)
3	(1,2) (3,5)
4	(1,5) (3,4)
5	(2,3) (4,5)
6	N/A

3. (a) The variable n starts as the length of the vector, and decreases by at least 1 each step, so the loop will always terminate.

   (b) The worst case is when n decreases by only 1, so when $x(i) > 0$ for all i.

   (c) In the worst case, the loop is invoked n times, and in each loop we perform 2 additions and/or subtractions resulting in a total of $2n + 1$ arithmetic operations. This solution is acceptable, but some students may note (correctly) that we could also include $2n$ comparison operations as well.

   One might even suggest that the "while" and "if" statements are operations that we should also include, and that each piece of arithmetic requires access to memory, but we usually don't got to this level of detail in this type of analysis.

   (d) The total number of operations is $2n + 1 = O(n)$ (or $4n + 1$, which also is $O(n)$).

[4 marks]

4. (a) All else equal, Algorithm B is preferable. Its asymptotic performance is better, because there exists constant m such that for all $n > m$ we have $n \geq (\log n)^k$ for any k, *i.e.*, at some point Algorithm A becomes slower. [1 mark]

   [Note: $O\big((\log n)^k\big)$ is called *polylogarithmic*. We might write $O\big(\text{poly}(\log n)\big)$. but remember we drop lower-order terms, and constant factors leaving just the highest power of the log. All polylogarithm functions are $O(n^\varepsilon)$ for any $\varepsilon > 0$ (in fact we can make the stronger statement that it is $o(n^\varepsilon)$).]

   (b) Three reasons why this assessment might be naive.

   - Big-O notation ignores constant factors and lower-order terms, but these might change the compute times drammatically for finite n, leading to B being better for a particular size of problem.
   - Big-O notation means $\leq$, so $O\big((\log n)^k\big) = O(n)$, *i.e.*, the $O(n)$ algorithm could also be $O\big((\log n)^k\big)$, and hence actually have as good, or even better asymptotic performance than the other algorithm.
   - Computational complexity is often measured for the worst case, but the more common cases might be much better.

[3 marks]