MA3227 Numerical Analysis II

Lab Session 8

Simon Etter, 2019/2020

1 Winning probabilities for tic-tac-toe

Tic-tac-toe is a simple paper-and-pencil game for two players. Please read the introduction of https://en.wikipedia.org/wiki/Tic-tac-toe if you have not heard of this game before. In the following, we will consider both the standard tic-tac-toe game and its generalisation to an $m \times n$ board where the goal is to occupy k squares in a row. The latter is known as the m, n, k-game, see https://en.wikipedia.org/wiki/M,n,k-game.

Our aim in this task is to compute the probabilities for the three different outcomes of the m, n, k-game assuming both players choose their moves uniformly at random from the list of all possible moves during their turn. These probabilities can be computed in two different ways.

- Play out all possible games and compute

$$P(\text{outcome}) = \sum_{\text{game} \in \text{ (set of all games)}} P(\text{game}) \begin{cases} 1 & \text{if "game" ends in "outcome".} \\ 0 & \text{otherwise.} \end{cases}$$

  If we interpret a game as a sequence of moves, i.e. game $= (m_1, m_2, m_3, \dots, m_k)$, then $P(\text{game})$ can be computed recursively using

$$P\big((m_1, \dots, m_{k+1})\big) = \frac{P\big((m_1, \dots, m_k)\big)}{\text{number of possible moves in turn } k}.$$

  I will refer to this as the recursive approach.

- Play out a few games with random moves and estimate

$$P(\text{outcome}) \approx \text{fraction of games which ended in the given outcome.}$$

  I will refer to this as the Monte Carlo approach.

Your tasks are as follows.

- Fill in the blanks in the code file for this lab session.
- Run `main()` and verify that the Monte Carlo estimate is reasonably close to the exact result computed via the recursive approach.
- Replace `m,n,k = 3,3,3` with `m,n,k = 4,3,3` and note how the runtimes change. You should observe that the runtime of Monte Carlo increases only very little while the runtime for the recursion increases substantially (from $\sim$ 0.1 seconds to $\sim$ 30 seconds on my machine).
- You may try running `recursion(4,4,3)`, but please be warned that this may take several hours. I have not done a full run of `recursion(4,4,3)` myself. Compare this with `monte_carlo(4,4,3,100_000)`, which terminates virtually instantaneously.
- Can you explain the observation in the previous two points?