

CS319: Scientific Computing (with C++)

## CS319 Lab 4: Optimized Optimization

Week 6 (15-16 Feb, 2024)

**Goal:** Compare two different methods for solving an optimization problem.

- ▶ This lab builds on Lab 3 from last week; you should complete that first.
- ▶ Submit your code and report by 5pm, Tuesday, 20th February.
- ▶ You may have to “demo” your code at one of next week’s labs before you get a grade.

## Recall: Optimization

“**Optimisation**” is the process of finding a maximum or minimum value of some function. For the purposes of this lab, it means finding the point at which a given function achieves its maximum value.

- ▶ We'll take a given function,  $f$ , which we call the **objective function**.
- ▶ We find the value of  $m$  that maximises  $f$  in a given interval,  $[a, b]$ . That is, find  $m$  such that  $a \leq m \leq b$ , and  $f(m) \geq f(x)$  for all  $x \in [a, b]$ .

In Lab 3, you used the Bisection Method to maximise

$$f(x) = e^{-2x} - 2x^2 + 4x$$

in the interval  $[-1, 3]$ ,

You should have found that 22 iterations were needed to locate the maximising value of  $x$ , subject to a tolerance of  $10^{-6}$ .

## Algorithm 2: Newton

The Bisection method from Lab 3 is quite robust: providing that the function is continuous, it will find an approximation of its maximum in the desired interval. However, there are much faster methods, the most important being *Newton's Method for Optimisation*: choose an initial guess  $x_0$ , and set

$$x_{k+1} = x_k - f'(x_k)/f''(x_k) \text{ for } k = 0, 1, 2, \dots$$

When implement this method. Note that it is different from Bisection in that one only provides a single initial guess, and also that we must provide both  $f'$  and  $f''$ .

## Assignment

- (i) Write a function that implements the **Newton Algorithm**. It should operate like the `Bisection()` function from Lab 3. Specifically, ...
  - (a) The function (and its derivative) to be optimised should be passed as an argument to your `Newton()` function.
  - (b) Unlike `Bisection()` it takes as single initial guess as input.
  - (c) It iterates until the difference between two successive estimates is less than a user-defined tolerance, which is defined as a global variable.
  - (d) The number of iterations taken should be stored in a variable that is passed by the reference.
  - (e) An argument is passed to the function that determines the maximum number of iterations allowed. It should have a default value of 10. In the `main()` function, the user is prompted for its value.
- (ii) Change the code so that it maximises

$$g(x) = x + \sin(2x) \tag{1}$$

in the interval  $[-1, 2]$ .

- (iii) In the `main()` function, the user is prompted for the maximum number of iterations.

## Assignment

- (iv) Both the Bisection and Newton optimizers should be called in `main()`, which should output the estimates they compute, and the number of iterations the used. For the Newton method, take any point in  $[-1, 2]$  as your initial guess.

Submit your code to the “Lab 4” section of 2324-CS319 on Canvas. Make sure your C++ code includes your name and ID number as comments at the top.

In addition, upload a VERY short report that includes your name, ID number, the output your code generates, and a statement as to which method is more efficient.

**Deadline: 17.00, Tuesday 20 Feb.**