Numerical Analysis 4315/5315, Assignment 2

Mathematics: Error analysis for Newton method. Experimental determination of order.
Programming: Arrays, formatting and plotting.

1- In class we defined Newton method by $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$. If $x^\star$ is the root, $f(x^\star) = 0$, then the error was shown by $e_n = x_n - x^\star$. For Newton's Method we found $e_{n+1} \approx Ce_n^2$ where $C = f''(x^\star)/2f'(x^\star)$. This is one example of a type of error behavior, in general we have an error evolution formula $e_{n+1} \approx Ce_n^p$. The question is how can we determine $C$ and $p$? You can do it by algebra, using perhaps a bit of regression, or more simply by plotting. For now assume error $e$ and $C$ are positive numbers. Take the logarithm of the error evolution formula to get $\ln(e_{n+1}) \approx \ln(C) + p\ln(e_n)$. So if we plot the logarithm of new error versus the old one the we should get almost a straight line with slope ————- (fill in the blank!) and intercept ———— . (Remember $y = mx + b$? Who is playing the role of $y$, $m$, $x$, and $b$ here?)
To plot the error evolution formula we need to keep the values of error (as opposed to what you might have done last week when you could throw them away as you used them). That means you need an array, to keep all the values in it. To keep things simple, we also work on an equation whose root we know, e.g $x^2 = 4$, or one whose root has been accurately approximated in a prior run. You need this to find the error in each stage.
So within Newton loop, with a running index such as $i$, you can add two lines to represent the following pseudo code.

```
xln(i) = log(old x - exact x)
yln(i) = log(new x - exact x)
```

when the loop is finished you have two complete vectors xln and yln. You can plot the curve by `plot(xln,yln)`. Try to find $p$ and $C$ from the graph and compare with theoretical values.
You will notice several issues or difficulties:
a- The plot is not a straight line (why?) What is the remedy? How should we find a $p$? Welcome to the world of approximations, murky math, and finite precision. We will talk about above issues at length later. But it is better if you form an idea in practice before a well-processed answer is given to you.
b- There is some redundancy in our programming approach above. Can you detect the duplications? Can you write a program that is more efficient? For a single run of Newton Method this efficiency won't be noticeable but in big projects it is an issue.
c- If you start Newton method on a different side of the root, say at $x = 1$ instead of $x = 3$, then above code will cause an error message. What steps should be taken to fix this problem.

---

2- Write a bisection algorithm to solve $x^2 = 5$, starting at $a = 2$ and $b = 3$ for the initial interval guess.

---

3- Continuation of the optional advanced project from last week.
Suppose we have non-negative numbers $Y = (y_1, y_2, \cdots, y_n)$ from which we make the generalized averages $A = (a_1, a_2, \cdots, a_n)$.
a- Observe that $a_1 \geq a_2 \geq \cdots \geq a_n$. Can you give a proof? Can you find a proof? Can you rewrite the proof from memory?
b- Assume $n$ is a large number. Half of $y_i$'s are 1 and the other half are $x$. Write a program to evaluate and plot the vector $A$ for a given $x$. Study the behavior of $a_{n/2}$ as a function of $x$.