CS6210 - Homework/Assignment-2

Arnabd Das(u1014840)

September 15, 2016

Question-1: Exercise-3

Consider the fixed point iteration $x_{K+1} = g(x_k), k = 0, 1, \dots,$ and let all the assumptions of the fixed point theorem holds. We can write the following using Taylor series expansion around the root x^*:

$$g(x_k) = g(x^*) + g'(x^*)(x_k - x^*) + \frac{g''(x^*)}{2!}(x_k - x^*)^2 + \dots + \frac{g^{(m)}(x^*)}{m!}(x_k - x^*)^m + \frac{g^{(m+1)}(\zeta)}{(m+1)!}(x_k - x^*)^{m+1}$$

Since, $g(x^*) = x^*$ and $g(x_k) = x_{k+1}$, replacing in the above equation, we get:

$$x_{k+1} = x^* + g'(x^*)(x_k - x^*) + \frac{g''(x^*)}{2!}(x_k - x^*)^2 + \dots + \frac{g^{(m)}(x^*)}{m!}(x_k - x^*)^m + \frac{g^{(m+1)}(\zeta)}{(m+1)!}(x_k - x^*)^{m+1}$$

Consider the presence of derivatives greater than equal to m'th order and the absence of 1 to (m-1)'th derivative. It means, the m'th order derivative will drive the convergence of the error function, and the higher order derivatives of order greater than m can be absorbed in the remainder term. Hence, for m being the lowest derivative that is non-zero, we can write the above expression as:

$$x_{k+1} - x^* = \frac{g^{(m)}(x^*)}{m!}(x_k - x^*)^m + \frac{g^{(m+1)}(\zeta)}{(m+1)!}(x_k - x^*)^{m+1}$$

which can be further simplified to,

$$\frac{x_{k+1} - x^*}{(x_k - x^*)^m} = \frac{g^{(m)}(x^*)}{m!} + \frac{g^{(m+1)}(\zeta)}{(m+1)!}(x_k - x^*)$$

In the limit of $k- > \infty$, the expression becomes:

$$\lim_{k->\infty} \frac{x_{k+1} - x^*}{(x_k - x^*)^m} = \frac{g^{(m)}(x^*)}{m!}$$

The right hand side is some constant that depends on the evaluation of the m'th order derivative of g at the root x^*. Consequently, the order of convergence is **m** which depends on the number of derivatives of g that vanish at $x = x^*$. (Here m is the lowest order derivative that is non-zero, indicating all the previous (m-1) order derivatives have vanished at $x = x^*$.

Now, given that $g'(x^*) = \dots = g^{(r)}(x^*) = 0$, where $r \geq 1$, we can use our previously derived expression to understand how fast the fixed point iteration converges. In our formula we considered, m'th dervative to be the first non-zero derivative in the expansion for that function at $x = x^*$. Since, here we are given $g^{(r)}(x^*) = 0$, we can consider r+1 to be the first non-zero derivative. Hence, we substitute m=r+1, then our expression becomes:

$$\frac{x_{k+1} - x^*}{(x_k - x^*)^{r+1}} = \frac{g^{(r+1)}(x^*)}{(r+1)!} + \frac{g^{(r+2)}(\zeta)}{(r+2)!}(x_k - x^*)$$

In the limit of $k- > \infty$, the expression becomes:

$$\lim_{k->\infty} \frac{x_{k+1} - x^*}{(x_k - x^*)^{(r+1)}} = \frac{g^{(r+1)}(x^*)}{(r+1)!}$$

Thus the order of convergence is **r+1** and the speed of convergence is the constant term in the rhs which corresponds to the (r+1)'th order derivative of g evaluated at $x = x^*$, such that the constant term is :

$$\frac{g^{(r+1)}(x^*)}{(r+1)!}$$

Question-2: Exercise-4

Given function:

$$g(x) = x^2 + \frac{3}{16}$$

a To find the two fixed points of this function, lets equate it to 0.

$$g(x) = x^2 + \frac{3}{16} = 0$$
$$=> 16x^2 - 16x + 3 = 0$$
$$(x - \frac{3}{4})(x - \frac{1}{4})$$

Thus the roots are at $x^* = \frac{3}{4}$ and $x^* = \frac{1}{4}$ (Answer)

(b) Consider a small interval containing the root $\frac{1}{4}$ say $(0.15, 0.3)$
$g(0.15) = 0.210000$ and $g(0.3) = 0.277500$, such that $0.15 \leq g(x) \leq 0.30$ for $x \in C[0.15, 0.30]$
Also, $g'(x) = 2x$ and $|g'(x)| < 1$ for x $\in [0.15, 0.3]$
Thus, it satisfies all criterias of Fixed Point Theorem . Hence, there is a unique root in this interval, which happens to be $x^* = \frac{1}{4}$
Also, the convergence criteria is:

$$|x_{k+1} - x_k| \leq |g'(x)||x_k - x_{k-1}|$$

Since, $|g'(x)| < 1$, this is guaranteed to converge.

Now for the other root, $x^* = \frac{3}{4}$, consider the interval $[0.7, 0.8]$
$g(0.74) = 0.735100 < 0.74$ and $g(0.76) = 0.765100 > 0.76$, thus the condition for fixed point theorem of $a \leq g(x) \leq b$ for x $\in [a, b]$ is **not satisfied**. Around this interval, $g(x)$ is approaching towards $y = x$, from the region of $y < x$, thus there **does not exists** an interval $[a, b]$ around this root, where $a \leq g(x) \leq b$ for x $\in [a, b]$ holds. Since the convergence criteria is:

$$|x_{k+1} - x_k| \leq |g'(x)||x_k - x_{k-1}|$$

The slope $g'(x)$, after x=0.5, is always greater than 1, it might not provide convergence since $|g'(x)|$ drives the convergence criteria which might diverge due to slope of greater than 1.

(c) From (b), we consider the root $x^* = \frac{1}{4}$ that is amenable to fixed point iterations. Since, for this method, the first order derivative exists, the convergence is linear. Thus the following relation holds:

$$|x_k - x^*| \approx \rho|x_{k-1} - x^*| \approx \cdots \approx \rho^k|x_0 - x^*|$$

where, $\rho = |g'(x)|$. To reduce the convergence error by a factor of 10: we have

$$|x_k - x^*| \approx \frac{1}{10}|x_0 - x^*|$$

Hence, we get,

$$\rho^k \approx 0.1$$

$$k \log_{10} \rho = -\log_{10} 10 = -1$$

At $x^* = \frac{1}{4}$; $g(x^*) = 2x^* = 0.5$, Thus

$$k = -\frac{1}{\log_{10} 0.5} = 3.32 \approx 4(Answer)$$

Question-3: Exercise 8

Let the number of iterations for Newton's Method = N1
Let the number of iterations for Secant's Method = N2
Given computation for f' takes α times the cost of computing f Newton's Formula:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Computation cost per iteration for Newton = f + αf = $(1+\alpha)f$

Secant's Formula;

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}$$

Considering we keep track of previous calculations of $f(x_{n-1})$ for reuse in evaluating x_{n+1}, there are only two function calls per iteration.
Computation cost per iteration for Secant = 2f

For Newton's method to be more efficient than Secant's Method, the following cost function inequality must hold.

Total Newton's Method Cost ≤ Total Secant's Method Cost
Assuming same initial guess qualities
,

$$(1+\alpha) \times f \times N1 \leq 2 \times f$$

$$=> \alpha \leq 2\frac{N2}{N1} - 1$$

Thus we need to figure out the expression for $\frac{N2}{N1}$. Consider the convergence expression:

$$|x_{k+1} - x^*| \approx c.|x_k - x^*|^\phi$$

$$|x_{k+1} - x^*| \approx c.||x_{k-1} - x^*|^\phi|^\phi$$

Thus, after if we keep replacing the error terms until we reach the inital error expression and ignore the constant terms since the power of the error terms drives convergence in this case, such that

$$|x_{k+1} - x^*| \approx |x_0 - x^*|^{\phi^k}$$

We write: $err_{final} = |x_{k+1} - x^*|$ and $err_{init} = |x_0 - x^*|$ For Newton's Method with N1 iterations and $\phi = 2$:

$$err_{final} \approx err_{init}^{2^{N1}}$$

For Secant's Method with N2 iterations and $\phi = 0.1618$:

$$err_{final} \approx err_{init}^{(1.618)^{N2}}$$

Since, for efficiency, we expect the err_{final} to be same for same rate of convergence when starting from the same initial guess quality: Thus,

$$err_{init}^{2^{N1}} = err_{init}^{1.618^{N2}}$$
$$=> 2^{N1} = 1.618^{N2}$$
$$\frac{N1}{N2} = \log_2 1.618$$

Plugging the above result back to the expression derived earlier for α

$$\alpha \leq \frac{2}{\log_2(1.618)} - 1$$
$$=> \alpha \leq 1.8809$$

Thus α should be less than equal to 1.88, or in other words, the cost of computation of first derivative in newton's method requires to be less than 1.88 times the cost of computing the function itself

Question-4: Exercise 10

Given function,

$$f(x) = (x-1)^2 e^x$$

(a) Deriving Newton's iteration for this function:

$$f'(x) = (x^2-1)e^x$$
$$\frac{f(x)}{f'(x)} = \frac{(x-1)^2 e^x}{e^x(x^2-1)} = \frac{x-1}{x+1}$$

Thus Newton's formulation will be,

$$x_{k+1} = x_k - \frac{x_k - 1}{x_k + 1} = \frac{x_k^2 + 1}{x_k + 1}$$

The iteration is not defined at x_k = -1 since for x_k=-1, the denominator becomes 0.

For Newton's method, error convergence is expressed as:

$$e_{k+1} = -\frac{f''(\zeta)}{2f'(x_k)} e_k^2$$

where, ζ is a point between x_k and x^*
For, $f'(x) = 0$ this formulation is undefined and for our function, we have $f'(x) = (x^2 - a)e^x$, which is zero at x=1

Let us express the iteration function of Newton's Method as below

$$\phi(x) = x - \frac{f(x)}{f'(x)}$$

such that, at root x^*, $\phi(x^*) = x^*$
Then from the Taylor series we can write upto second order derivatives(existence of second order derivative means higher order dertivatives will have negligible effect on the error term):

$$x_{k+1} = \phi(x^*) + (x_k - x^*)\phi'(x^*) + \frac{1}{2}(x_k - x^*)^2\phi''(\zeta)$$

where, ζ lies between x_k and x^*
The above form in error terms reduces to:

$$e_{k+1} = e_k\phi'(x^*) + \frac{1}{2}e_k^2\phi''(\zeta)$$

Now from the previously defined $\phi(x)$ and deriving its first derivative $\phi'(x)$, we get the expression for $\phi'(x)$ as:

$$\phi(x) = x - \frac{f(x)}{f'(x)}$$
$$\phi'(x) = \frac{f(x)f''(x)}{f'(x)^2}$$

Since we want to remove $f'(x)$ from the denominator, we apply $\lim_{x->x^*}$ and use L'Hopital's rule successively twice, such that :

$$Lim_{x->x^*}\phi'(x) = Lim_{x->x^*}\frac{f(x)f''(x)}{f'(x)^2}$$
$$= \lim_{x->x^*}\frac{f''(x)^2 + 2f'(x)f'''(x) + f(x) + f''''(x)}{2(f'(x)f'''(x) + f''(x)^2)}$$
$$= \lim_{x->x^*}\frac{f''(x)^2}{2f''(x)^2} = \frac{1}{2}$$

Plugging this result back into expression for e_{k+1} :

$$e_{k+1} = \frac{1}{2}e_k + O(e_k^2)$$

In the presence of e_k term, e_{k+1} becomes order of e_k.
Thus it is of linear convergence similar to bisection and certainly not quadratic.

b Newton's method for this problem is implemented in NewtonMethod.m in Prob4 folder. The starting guess for x_0 used is 2.0 .As explained in the previous part, we expect the convergence using newton's method in this problem to be linear. We derived the expression for e_{k+1} earlier as:

$$e_{k+1} = \frac{1}{2}e_k + O(e_k^2)$$

, which is linear such that we can write

$$e_{k+1} \approx \frac{1}{2}e_k$$

Then, if the final error is err_{final} and the initial error is err_0, and it took k iterations to reach convergence, then the following should hold:

$$e_{k+1} = \left(\frac{1}{2^k}\right)err_0$$

From the simulation, we get the following data as summarized below
$err_0 = 2.0 - 1.0000000000000002$
$err_{final} = 1.0000000000000007 - 1.0000000000000002 \approx 4.4 \times 10^-16$

Number of iterations, K = 54
If we plugin , k and err_0 in the above expression, we get the prediction for err_{final} to be approximately 0.55×10^-16, which closely matches our results.
(c) The given function $f(x) = (x-1)^2 e^x$ is always positive $\forall$ x $in\mathbb{R}$, and it just touches the x-axis as a minima without a crossing over. Thus the function never changes sign around the root, which is a necessary condition for the bisection method. Thus it will not be easy to apply the bisection method on this function for finding its root.

Question-5: Exercise 12

Given $a > 0$, we wish to compute $x = ln(a)$
(a) To compute x, we make the fixed point formulation: we can mechanize $= ln(a)$ to be rewritten as

$$e^x = a$$

$$e^x - a = 0$$

Thus we can write, $f(x) = e^x - a$
For newton's iterative method, we can write the following:

$$x_{k+1} = x_k - \frac{f(x)}{f\prime(x)}$$

$$x_{k+1} = x_k - \frac{e^{x_k} - a}{e^{x_k}}$$

$$x_{k+1} = x_k - 1 + \frac{a}{e^{x_k}}$$

(b) The error convergence expression for Newton's Method is given as:

$$e_{k+1} = -\frac{f''(\zeta)}{2f\prime(x_k)} e_k^2$$

Since $f\prime(x) \neq 0, thisformulahols$. Hence, the convergence for this function using Newton's Method is quadratic.

(c) For Secant's method, we can write the following:

$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}$$

$$x_{k+1} = x_k - \frac{(e^{x_k} - a)(x_k - x_{k-1})}{(e^{x_k} - a) - (e^{x_k} - a)}$$

$$x_{k+1} = x_k - \frac{x_k - x_{k-1}}{1 - \dfrac{e^{x_{k-1}} - a}{e^{x_k} - a}}$$

(d) We assume here that no computation is performed twice, that is once (k-1)th step gets computed , it is stored for next two iteration to be reused. Then, by our formulation, there is only a single exponential calculation in Secant's method . In Newton's method as well, there is a single evaluation of exponential in each iteration step. Thus the cost of exponential evaluation in each iteration is same for both Newton's method and Secant's method. However, since, Newton has quadratic convergence while Secant has superlinear convergence and Newton's method is more efficient than Secant's method since it converges quickly.

Question-6: Exercise 15

Given , equation for x >0

$$x + ln(x) = 0$$

(1) Show analytically that there is exactly one root, $0 \leq x^* \leq \infty$

Let us consider the continuous function: f(x) = $x + ln(x)$: whose root we need to find for $f(x) = 0$.
Suppose $g(x) = \exp^{-x}$, such that $x = g(x)$ gives a solution at the root of f(x).
Let us write the objective function as $\phi(x) = g(x) - x$.
In the interval $[0, \infty]$, $\phi(0) = 1 > 0$ and $\phi(\infty) = -\infty < 0$. Hence, by the intermediate value theorem, there is a root $0 < x^* < \infty$ such that $\phi(x^*) = 0$. Thus, $g(x^*) = x^*$, so x^* is a fixed point.

Also, $g\prime(x) = -\exp^{-x}$, hence —$g\prime(x)| < 1$, for $x > 0$
Consider an additional root in the same interval, y^* such that $g(y^*) = y^*$, then

$$|x^* - y^*| = |g(x^*) - g(y^*)| = |g\prime(\zeta)(x^* - y^*)| \leq |x^* - y^*|$$

where, ζ is an intermediate value between $x^* and y^*$. This inequality will hold for $|g\prime(\zeta)| < 1$ if and only if $x^* = y^*$
This proves there is only one unique root in that interval and it satisfies the requirements of fixed-point theorem(Answer)

(2) Figure-1 shows the plot of $(x + ln(x))$ vs x

(3.i) The bisection method is coded for the interval 0.5 to 0.6. The subroutine is available in bisection.m called from main.m under folder Prob6. The detailed prints of the iterates are printed out to the command window. Here we analyse the final solution and the iteration count.
The solution from bisection = 0.56714329030365
Number of iterations for bisection = 29

The choice of the interval is valid since there is a crossing of the x-axis as $x + ln(x)$ goes from negative to positive values for increasing values of x. It is a necessary condition for the success of bisection method that the function evaluation at the two extremes of the given interval yields opposite signs, and this interval of $[0.5, 0.6]$ satisfies that condition, hence the selection of this interval is justified.
he convergence is linear such that $|x_{k+1} - x^*| \approx \rho^k |x_0 - x^*|$
From the reported data from the subroutines, $err_{final} = 1.863e - 10$ and $err_{initial} = 0.067143$. The slope of g around the solution point is 0.606530.
Thus $err_{inital} \times g\prime(0.56714)^{29} \approx 10^{-10}$, which corresponds to linear convergence

(3.ii)

From the previous answer, we have already formulated the fixed point formulation for evaluating f(x) as

$$x = g(x) = \exp^{-x}$$

Consider the interval $[0.5, 0.66]$. Here g(0.5) = 0.606 and g(0.66) = 0.516. Thus, for x $\in C[0.5, 0.66]$, $0.5 \leq g(x) \leq 0.66$

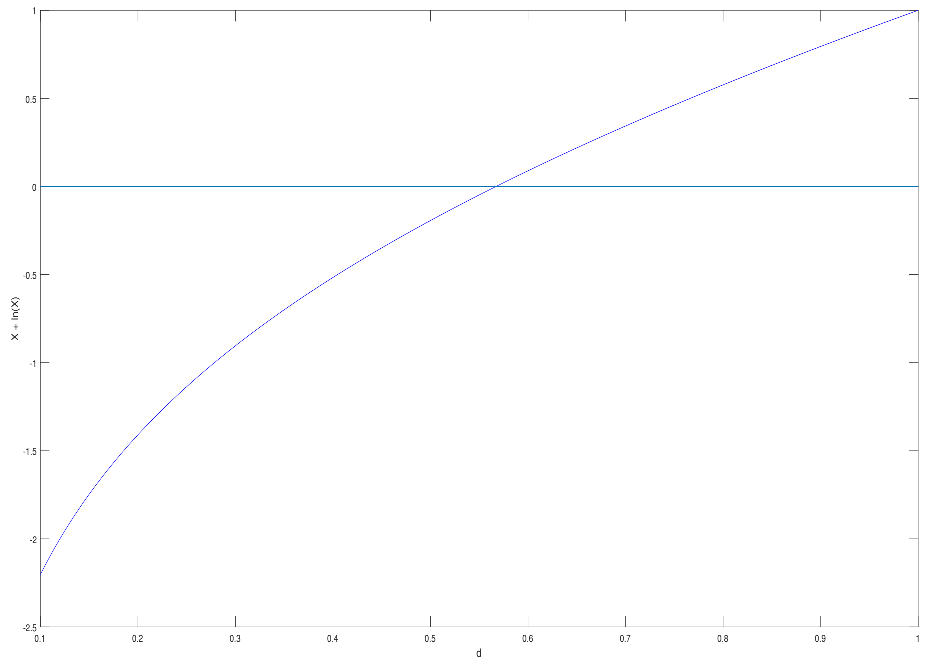

Figure 1: Plot of $x + ln(x)$ vs x $\in$ (0.1,1)

Also, $g\prime(x) = \exp^{-x}$, so $|g\prime(x)| < 1$ for $x > 0$. Thus , it satisfies all the conditions of Fixed point theorem.(answer)
So, there is a point x^* such that $0.5 < x^*0.6$, where g(x) = x^*
The implementation subroutine for fixed point is available in fixedPoint.m and called from main.m.The detailed prints of the iterates are printed out to the command window. Here we analyse the final solution and the iteration count.
The solution from fixed point iteration = 0.56714329037983
Number of iterations = 38
The convergence is linear such that $|x_{k+1} - x^*| \approx \rho^k |x_0 - x^*|$
From the reported data from the subroutines, $err_{final} = 8.277e - 11$ and $err_{initial} = 0.06129144$. The slope of g around the solution point is 0.606530.
Thus $err_{inital} \times g\prime(0.56714)^{38} \approx 10^{-10}$, which corresponds to linear convergence

(3.iii) The subroutine for Newton's method is available in newtonMethod.m and called from main.m. The detailed prints of the iterates are printed out to the command window. Here we analyse the final solution and the iteration count.
The solution from Newton method = 0.56714329040978
The number of iterations = 4
Clearly , a comparative look with the previous methods shows it converges quadraticaly

(3.iv) The subroutine for secant's method is available in secantMethod.m and called from main.m. The detailed prints of the iterates are printed out to the command window. Here we analyse the final solution and the iteration count.
The solution from Secant Method = 0.56714329040978

The number of iterations = 5
Clearly, better than linear but not as good as newton's quadratic convergence. It is superlinear.