Example Solutions for Assignment 4

Question 1

(b)

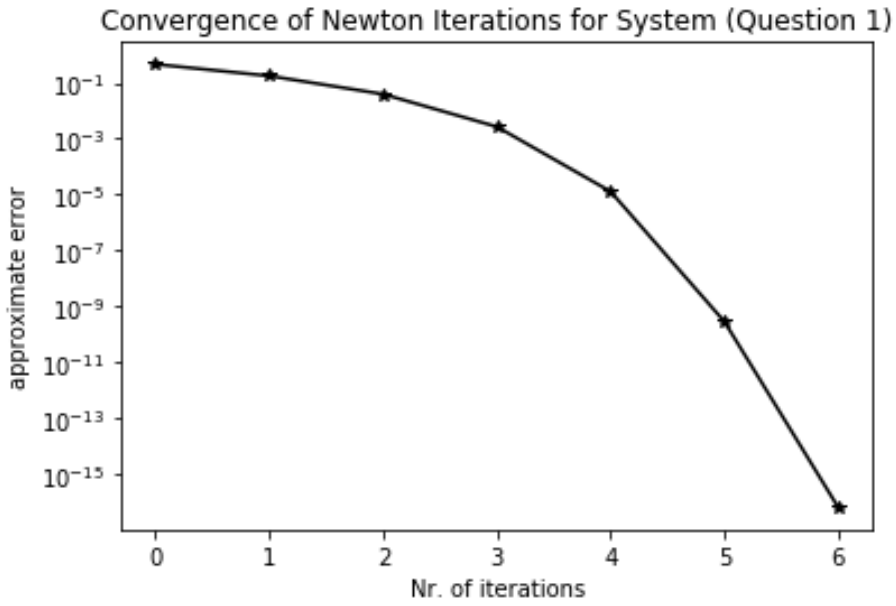

Figure 1: The (approximate) error after k steps of Newton iteration. The curve corresponds to quadratic convergence.

(c) The solution is: $\mathbf{x} = (x_1, x_2) = (0.86406221, 0.34086698)$.
Yes. Newton iteration for systems also appears to have quadratic convergence. The number of digits of accuracy of the solution appears to double at each iteration, i.e. both the approximate error and residual are *approximately* the square of their values at the previous iteration.

Question 2

(a) The interpolating polynomial is:

$$P_2(x) = 2.0 + 0.45740\ x - 0.26838\ x^2$$

(b)

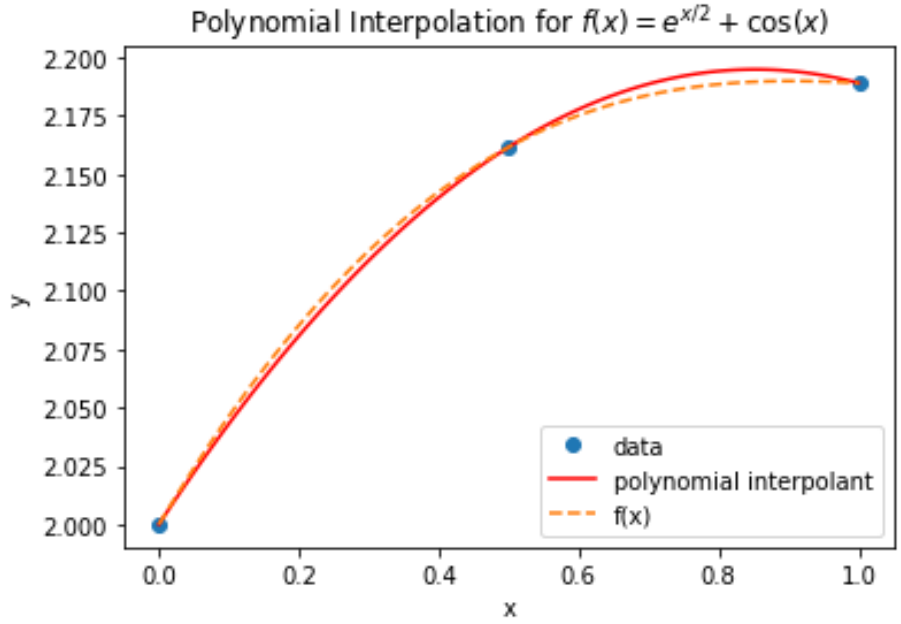

Figure 2: The function $f(x) = e^{x/2} + \cos(x)$ and the interpolating polynomial P_2 using the interpolation nodes $x_0 = 0, x_1 = 0.5, x_2 = 1$.

(c) The theorem from Lecture 16 states: $\exists \xi \in I$ such that

$$E_n(x) := f(x) - P_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{k=0}^{n} (x - x_k),$$

and thus,

$$|E_n(x)| := |f(x) - P_n(x)| \leq \max_{x \in I} \frac{\left|f^{(n+1)}(x)\right|}{(n+1)!} \prod_{k=0}^{n} |x - x_k|.$$

Here, we have $n = 2$, $I = (0, 1)$, and $x_0 = 0, x_1 = 0.5, x_2 = 1$, so

$$|E_2(x)| := |f(x) - P_2(x)| \leq \max_{x \in (0,1)} \frac{\left|f^{(3)}(x)\right|}{3!} |x||x - 0.5||x - 1|.$$

From which it follows,

$$|E_2(x)| := |f(x) - P_2(x)| \leq \frac{1}{3!} \max_{x \in (0,1)} \left|f^{(3)}(x)\right| \max_{x \in (0,1)} |x| \max_{x \in (0,1)} |x - 0.5| \max_{x \in (0,1)} |x - 1|.$$

Look at each term separately:

$$\max_{x \in (0,1)} |x| = 1, \qquad \max_{x \in (0,1)} |x - 0.5| = \frac{1}{2}, \qquad \max_{x \in (0,1)} |x - 1| = 1$$

For $\max_{x\in(0,1)} \left|f^{(3)}(x)\right|$, we use the hint that the max will occur at the end points (i.e. $x = 0$, $x = 1$), or at a local max in the interval. Local max can occur with the derivative of the function is zero (i.e. the derivative of $f^{(3)}(x)$, which is $f^{(4)}(x)$). Compute:

$$\begin{aligned} f(x) =& e^{x/2} + \cos(x) \\ f^{(1)}(x) =& \frac{1}{2}e^{x/2} - \sin(x) \\ f^{(2)}(x) =& \frac{1}{4}e^{x/2} - \cos(x) \\ f^{(3)}(x) =& \frac{1}{8}e^{x/2} + \sin(x) \\ f^{(4)}(x) =& \frac{1}{16}e^{x/2} + \cos(x) \end{aligned}$$

$f^{(4)}(x)$ is never zero in the interval $x \in (0, 1)$ and therefore the max of $f^{(3)}(x)$ has to occur at $x = 0$ or $x = 1$. We have $f^{(3)}(0) = 1.0$ and $f^{(3)}(0) = 1.04756$ and so

$$\max_{x\in(0,1)} \left|f^{(3)}(x)\right| = 1.04756$$

Collecting all our results we get:

$$|E_2(x)| := |f(x) - P_2(x)| \leq \frac{1}{3!}(1.04756)(1)(\frac{1}{2})(1)$$

and finally:

$$|E_2(x)| := |f(x) - P_2(x)| \leq 0.088$$

I have rounded up to ensure the inequality is not violated.

(d) In this case, if more interpolation nodes are added, the maximum error would decrease. In particular, $\frac{1}{n!}$ will decease significantly as n gets bigger, while $f^{(n)}(x)$ will also decrease. The factors $|x - x_j|$ will also result in a decrease since these terms will all be less than 1.

Question 3

In this part, you are asked to write a psuedo-code for building the matrix V, for a given N, and with elements specified in the problem statement. To increase efficiency, we use the previously computed value of the element in the adjacent column, which is a similar product with one less term, so we only have to multipy one more factor, i.e. $(x_k - x_j)$. This saves a lot of computation because we don't have to recompute the whole product for each element; this is a similar idea as is used in the efficient algorithm for the evaluation of the Vandermonde matrix. We also initialize M as a matrix of ones, so we do not have to modify the first column.

Input:¯ integer n and an $n+1 \times 1$ array containing the interpolating nodes

1. Initialize M as an $n+1 \times n+1$ array of ones
2. **for** $i = 2, n+1$ **do** # go through all rows
3. **for** $j = 2, i$ **do** # only have to go up to ith column
   (a) $M_{i,j} \leftarrow M_{i,j-1} * (x_i - x_{j-1})$
4. end j loop
5. end i loop

Output: M, an $n+1 \times n+1$ array of floats

Computational Complexity: Inside both the i and j **for** loops, on line 3(a), there are 2 operations: one substraction and one multiplication. Thus the total FLOPS are given by:

$$\begin{aligned} FLOPS &= \sum_{i=2}^{n+1} \sum_{j=2}^{i} 2 \\ &= 2 \sum_{i=2}^{n+1} (i-1) \\ &= 2 \left(\sum_{i=2}^{n+1} i - \sum_{i=2}^{n+1} 1 \right) \\ &= 2 \left(\frac{n(n+1)}{2} - (n) \right) \\ &= n^2 - n \\ &= O(n^2) \end{aligned}$$

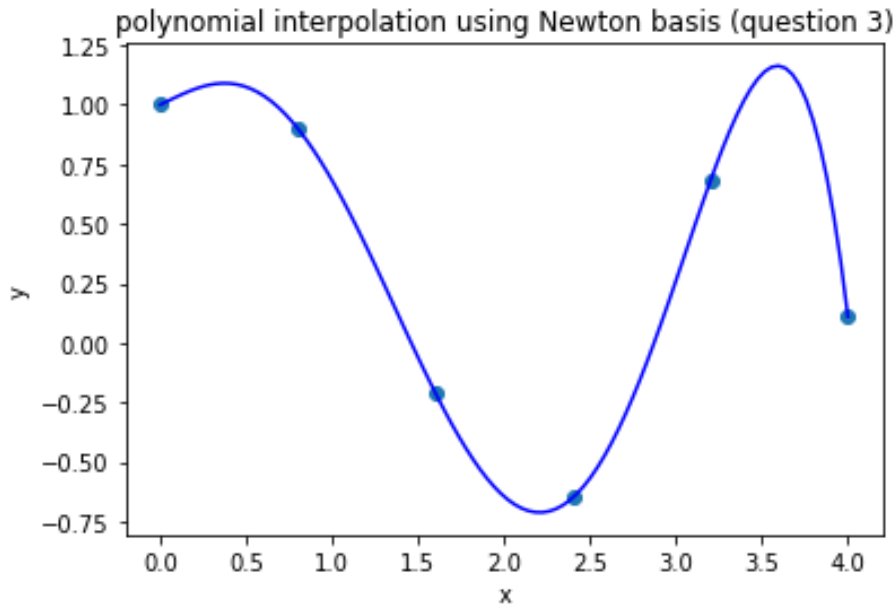

Figure 3: Polynomial interpolation using Newton Polynomial basis.