Robotics 811 - HW 2

Xiang Zhi Tan

October 1, 2015

1 Q1

This questions was done with discussion with Richard Goldstein

$$\begin{aligned}
P_2(x) &= f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) \\
&= f[x_0] + f[x_0, x_1]x - f[x_0, x_1]x_0 + f[x_0, x_1, x_2](x^2 - xx_0 - xx_1 + x_0x_1) \\
P_2'(x) &= f[x_0, x_1] + f[x_0, x_1, x_2](2x - x_0 - x_1)
\end{aligned}$$

where:

$$f[x_0, x_1] = \frac{f(x_0) - f(x_1)}{x_0 - x_1}$$

$$f[x_0, x_1, x_2] = \frac{\frac{f(x_0)-f(x_1)}{x_0-x_1} - \frac{f(x_1)-f(x_2)}{x_1-x_2}}{x_0 - x_2}$$

Now,I insert the point at $i = 1$ into $P_2'(x)$

$$\begin{aligned}
P_2'(\frac{x_0 + x_1}{2}) &= f[x_0, x_1] + f[x_0, x_1, x_2](2 \times \frac{x_0 + x_1}{2} - x_0 - x_1) \\
&= f[x_0, x_1] + f[x_0, x_1, x_2] \times 0 \\
&= f[x_0, x_1]
\end{aligned}$$

This proves $P_2'(x)$ at $i = 1$ is equal to $f[x_0, x_1]$.
Now,I insert the point at $i = 2$ into $P_2'(x)$

$$\begin{aligned}
P_2'(\frac{x_1 + x_2}{2}) &= f[x_0, x_1] + f[x_0, x_1, x_2](2 \times \frac{x_1 + x_2}{2} - x_0 - x_1) \\
&= \frac{f(x_0) - f(x_1)}{x_0 - x_1} + \frac{\frac{f(x_0)-f(x_1)}{x_0-x_1} - \frac{f(x_1)-f(x_2)}{x_1-x_2}}{x_0 - x_2}(x_0 - x_2) \times -1 \\
&= \frac{f(x_0) - f(x_1)}{x_0 - x_1} - \frac{f(x_0) - f(x_1)}{x_0 - x_1} + \frac{f(x_1) - f(x_2)}{x_1 - x_2} \\
&= \frac{f(x_1) - f(x_2)}{x_1 - x_2} \\
&= f[x_1, x_2]
\end{aligned}$$

This proves $P_2'(x)$ at $i = 2$ is equal to $f[x_1, x_2]$.

2 Q2

2(a)

Firstly, I calculate the values of $f(x_0), ...f(x_i)$ which gives us the vector $0, 0.0434, 0.1269, 0.2217, 0.3187, 0.4145, 0$
By plugging it in into the method with the following variables. $dividedDifferences(6, 2.25, xs, fxs)$.
The method will return the result:0.1737 which is equal to the ansIr given by matlab which is 0.1737.

2(b)

- Using the procedure at $x = 0.05$ with $n = 2$, gave the estimate of 4.9878.
- Using the procedure at $x = 0.05$ with $n = 4$, gave the estimate of 4.9442
- Using the procedure at $x = 0.05$ with $n = 40$, gave the estimate of 4.5872

The actual value of $f(0.05)$ is 4.5872 which is same as the estimate of 4.5872 by $n = 40$

2(c)

The Error estimate is as following:

n	E_n
2	3.4911
4	2.3584
6	3.5367
8	6.4873
10	12.9702
12	27.1445
14	58.4298
16	128.1824
18	285.0742
20	640.9705
40	2830400

The Error make sense, because I am trying to fit a polynomial on a non-polynomial function. As the interpolating polynomial will never become the function that I am trying to interpolating and it will only continue to over fitting the function causing massive error at the end of the functions

3 Q3

This questions was done with help from Reuben Aronson
Firstly, I assume the table contains the values $f(X_i), i = 0, ...n$ with $n = \frac{1}{n}$ and $x_i = -\frac{\pi}{2} + i \times h$. If $\bar{x} \in [x_{i-1}, x_i]$ then I am approximating the function with a

polynomial of degree 2, $p_1(x)$. From the notes, I know the error must be:

$$e_1(\bar{x}) = \frac{f'(xi)}{2!}(\bar{x} - x_{i-1})(\bar{x} - x_i)$$

The error of the first term $f'''(x)$ must be 1 as the function is a simple cos function. The error of the second term is estimated as following:

$$|(\bar{x} - x_{i-1})(\bar{x} - x_i)| \leq \max_{y \in [-h,0]} |(y - h)y|$$

Let us consider the function $g(y) = (y - h)y = y^2 - yh$. To find the maximum on the interval, I find the derivative of the function, $g'(y) = 2y - h$ and set it to zero $g'(y) = 0$. This means $g'(y) = 0$ if and only if $y = \frac{h}{7}2$. By inserting the value into the g(y), I find the maximum of $g(y)$. Which is:

$$\begin{aligned} g(\frac{h}{2}) &= \frac{h}{2}^2 - h(\frac{h}{2}) \\ &= -\frac{h^2}{4} \end{aligned}$$

Therefore the error for a linear interpolation is:

$$\begin{aligned} |e_1(\bar{x})| &\leq \frac{1}{2!} \times 1 \times \frac{h^2}{4} \\ &= \frac{h^2}{8} \end{aligned}$$

To get the 6 place accuracy, h need to be choose as following:

$$\begin{aligned} \frac{h^2}{8} &\leq 5 \times 10^-6 \\ h &= 0.002 \end{aligned}$$

Therefore, the number of table entries for a linear interpolation is as following:

$$\frac{\frac{pi}{2} + \frac{3\pi}{2}}{0.002} + 1 \approx 3143$$

If I am using a quadratic interpolation, the interval for $\bar{x}$ will be $[x_{i-1}, x_{i+1}]$ instead. The maximum of the error of $(\bar{x} - x_{i-1})(\bar{x} - x_i)(\bar{x} - x_{i+1})$ can be find in the notes, which is $\frac{2}{3}\frac{1}{\sqrt{3}}h^3$. The Error term for the quadratic equation is as following:

$$\begin{aligned} |e_2(\bar{x})| &\leq \frac{1}{3!} \times 1 \times \frac{2}{3}\frac{h^3}{\sqrt{3}} \\ &= \frac{h^3}{9\sqrt{3}} \end{aligned}$$

To get the 6 place accuracy, h need to be choose as following:

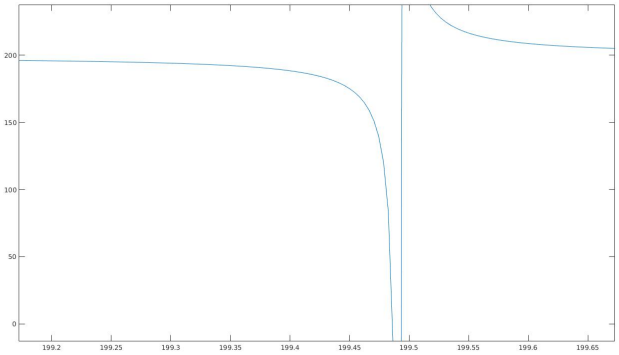
$$\frac{h^3}{9\sqrt{3}} \leq 5 \times 10^{-}6$$
$$h \approx 0.019827$$

Therefore, the number of table entries for a linear interpolation is as following:

$$\frac{\frac{pi}{2} + \frac{3\pi}{2}}{0.019827} + 1 \approx 318$$

4 Q4

This questions was done with discussion with Devin Schwab
First I observe the graph of the function $x - tan(x) = 0$ around the 200 point to find good values to be tested.



By visual inspection, I found that the root must be betIen 199.49 and 199. I tested this by inserting both values into the function and it gave us both -682.7305 and 197.1303. Since 199.49 won't be able to converge using Newton Method(implemented in the file titled *newtonMethod.m*), I first use those two values in the bisection method to give an estimate root. The code for the function is attached to this paper under the title *bisectionMethod.m*. The bisectionMethod returns 199.4861 as the root. As I plugged in the value into newton method, it gave us the same value of 199.4861. Therefore, the closet root to 200 is 199.4861

5 Q5

I received help from Reuben Aronson for this question
First, I look at the Newton Method:

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

Minus ξ from both sides will give us

$$\xi - X_{n+1} = \xi - (X_n - \frac{f(x_n)}{f'(x_n)})$$

Because $E_n = \xi - x_n$

$$E_{n+1} = E_n + \frac{f(x_n)}{f'(x_n)}$$

Let us set $\frac{f(x_n)}{f'(x_n)}$ to be $h(x)$

$$E_{n+1} = E_n + h(x)$$

Inserting $E_n = \xi - x_n$ into $h(x)$, I get

$$E_{n+1} = E_n + h(\xi - E_n)$$

$h(\xi - E_n)$ can be expressed as a Taylor series

$$h(\xi - E_n) = h(\xi) - E_n h'(\xi) + \frac{{E_n}^2}{2} h''(\xi) + ...$$

First I solve $h(\xi)$.

$$h(\xi) = \frac{f(\xi)}{f'(\xi)}$$

Because $f'(\xi) = 0$, the equation is indeterminate. Therefore, to solve this, I could apply the L'Hôpital's rule.

$$h(\xi) = \lim_{x \to \xi} \frac{f'(\xi)}{f''(\xi)} = 0$$

Now I solve $h'(\xi)$. By the notes, I know the form would as following

$$h'(\xi) = 1 - \frac{f(x)f''(x)}{[f'(x)]^2}$$

Again, the part of $\frac{f(x)f''(x)}{[f'(x)]^2}$ is indeterminate, I could apply the L'Hôpital's rule to it.

$$\lim_{x \to \xi} \frac{f(x)f''(x)}{[f'(x)]^2} = \lim_{x \to \xi} \frac{f'(x)f''(x) + f(x)f'''(x)}{2f'(x)f''(x)}$$

Because the solution is again indeterminate, I apply L'Hôpital's rule again.

$$\begin{aligned}
\lim_{x \to \xi} \frac{f'(x)f''(x) + f(x)f'''(x)}{2f'(x)f''(x)} &= \lim_{x \to \xi} \frac{f'(x)f'''(x) + f''(x)f''(x) + f'(x)f'''(x) + f(x)f''''(x)}{2f''(x)f''(x) + 2f'(x)f'''(x)} \\
&= \frac{f''(x)f''(x)}{2f''(x)f''(x)} \\
&= \frac{1}{2}
\end{aligned}$$

Plugging $\frac{1}{2}$ back into the original equation, I get

$$h'(\xi) = 1 - \frac{1}{2} = \frac{1}{2}$$

Now I plugging the value of $h'(x)$ and $h(x)$ back into $E_{n+1} = E_n + h(\xi - E_n)$

$$\begin{aligned} E_{n+1} &= E_n + h(\xi - E_n) \\ &= E_n + h(\xi) - E_n h'(\xi) + ... \quad = E_n + 0 - \frac{E_n}{2} + O({E_n}^2) = \frac{1}{2}E_n + O({E_n}^2) \end{aligned}$$

This establishes the linear convergence of Newton's Method when $f(\xi) = 0, f'(\xi) = 0$.

$$|E_{n+1}| \sim C|E_n|^1, \ where \ C = \frac{1}{2} + O({E_n}^2)$$

This proves that the method now converges linearly.
Now I change the newton's method function to $x_{n+1} = x_{n-1} - 2\frac{f(x_n)}{f'x_n}$. The only change would be that now $E_{n+1} = E_n + 2h(\xi - E_n)$. By expanding it:

$$\begin{aligned} E_{n+1} &= E_n + 2h(\xi - E_n) \\ &= E_n + 2(h(\xi) - E_n h'(\xi) + \frac{{E_n}^2}{2}h''(\xi) + O({E_n}^3) \\ &= E_n + 2(0) - 2(\frac{E_n}{2}) + {E_n}^2 h''(\xi) + 2(O({E_n}^3)) \\ &= {E_n}^2 h''(\xi) + 2(O({E_n}^3)) \end{aligned}$$

Now I only need to prove that $h''(\xi) \neq 0$. This could be done through calculation on matlab or wolframalpha. If done, it will prove that $h''(\xi) \neq 0$. Therefore,

$$|E_{n+1}| \sim C|E_n|^2, \ where \ C = O({E_n}^3)$$

This proves that the method now converges quadratically.

6 Q6

6(a)

The Muller Method was implemented based on the notes and algorithm given in the book Numerical Recipes in C on page 371. The implementation of the algorithm can be found in the file titled *mullerMethod2.m* attached to the homework.

6(b)

The initial guesses of each roots are based on the plot on the graph $x - tan(x)$. I observed where the zeros is and pick three values of x such that the root will be betIen them without going over the asymptotes. Following are the roots and

initial guesses.

x_0	x_1	x_2	roots
4	4.35	4.5	4.4934
6	6.5	7.8	7.7252
10	10.5	10.95	10.9041

6(c)

I wrote a script called *p6c.m* that runs the algorithm to find the correct k and real roots. First, I visually inspected the graph of the Bessel function at K = 100. This allow us to determine the ranges for the possible roots. Then, I set the initial k to be 3 and loop through the Muller function and slowly increasing k until the root become stable(they did not change since the last iteration). I found the roots to be stable at $k = 31$. The following is the roots found by the algorithm

x_0	x_1	x_2	roots
3	4	5	3.3817
5	6	7	7.0156
9	10	11	10.1735
12	13	14	13.3237

7 Q7

7(a)

From the two polynomials given, I can construct a Sylvester's matrix:

$$M = \begin{pmatrix} 1 & -12 & 41 & -42 & 0 \\ 0 & 1 & -23 & 41 & -42 \\ 1 & -2 & -35 & 0 & 0 \\ 0 & 0 & 1 & -2 & -35 \end{pmatrix}$$

Calculating the determinant will give us

$$det(M) = -7.1623e - 12 \approx 0$$

Because the determinant is approximately 0, this means that $p(x)$ and $q(x)$ shares a common root.

7(b)

Using the ratio method discussed in class I constructed the following equation:

$$
\begin{aligned}
x &= \frac{x^4}{x^3} \\
&= (-1)^{1+2}\frac{det(M_1)}{det(M_2)} \\
&= (-1)\frac{\begin{vmatrix} -12 & 41 & -42 & 0 \\ 1 & -12 & 41 & -42 \\ -2 & -35 & 0 & 0 \\ 1 & -2 & -35 & 0 \end{vmatrix}}{\begin{vmatrix} 1 & 41 & -42 & 0 \\ 0 & -12 & 41 & -42 \\ 1 & -35 & 0 & 0 \\ 0 & -2 & -35 & 0 \end{vmatrix}} \\
&= (-1)\frac{806736}{-115248} \\
&= 7
\end{aligned}
$$

8 Q8

8(a)

The following is the graphing of the following two functions using matlab

$$
\begin{aligned}
p(x, y) &= 2x^2 + 2y^2 - 1 \\
q(x, y) &= x^2 + y^2 + 2xy + x - y
\end{aligned}
$$

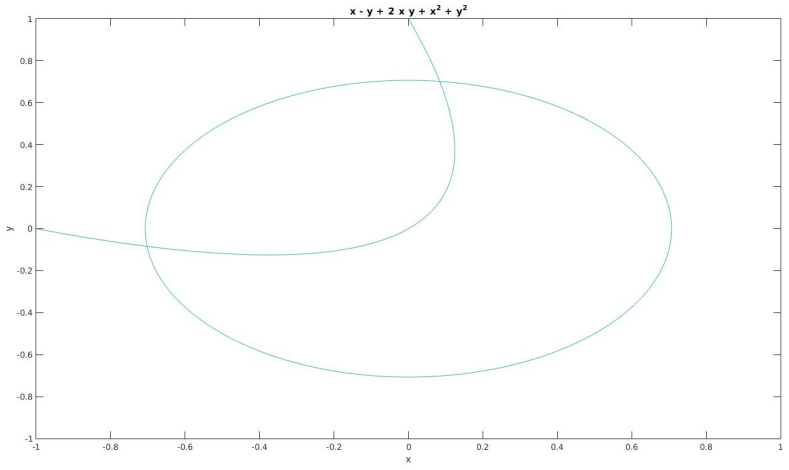


Figure 1: graphing of the two functions

8(b)

To find the roots of the equations, I first construct the resultant by eliminating y. The functions are re-written with x as constants.

$$p(y) = 2y^2 + (2x^2 - 1)$$
$$q(y) = y^2 + y(2x - 1) + (x^2 + x)$$

This then could be use to construct the Q matrix

$$Q = \begin{pmatrix} 2 & 0 & 2x^2 - 1 & 0 \\ 0 & 2 & 0 & 2x^2 - 1 \\ 1 & 2x - 1 & x^2 + x & 0 \\ 0 & 1 & 2x - 1 & x^2 + x \end{pmatrix}$$

Using the symbolic toolkit inside matlab, I could determine the determinant of the matrix, $det(Q) = 16x^4 - 16x^3 + 12x - 1$. Using the build in solver in matlab, I Ire able to solve the 4th degree polynomial, and it gave us 4 different roots (2 real roots and 2 imaginary roots).

$$0.8090 + 0.6360i$$
$$0.8090 - 0.6360i$$
$$-0.7021$$
$$0.0841$$

I plugged in the two real root into the original algorithm and it gave us the following values

$$x : -0.7021 \; y : 0.0840$$
$$x : 0.0840 \; y : 0.7021$$

8(c)

The result of the roots marked on the graph

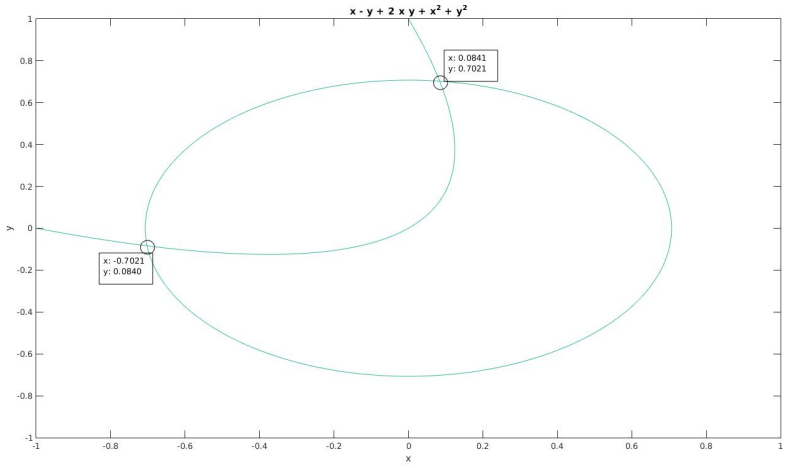


Figure 2: graphing of the two functions

9 Q9

9(a)

The linear equation would be based on Barycentric coordinate system which says that each point in a triangle could be written by three Barycentric coordinates. The three barycentric coordinates will be set as v in the linear equation.

$$\begin{pmatrix} x^{(i)} & x^{(j)} & x^{(k)} \\ y^{(i)} & y^{(j)} & y^{(k)} \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

The three values v_0, v_1 and v_2 will tells us whether the point is in the triangle. If the point lies in the triangle or on the edge for all the values, they will be $0 \geq v_i \leq 1$ for all i.

9(b)

The algorithm is implemented in the function in *interpolatePath.m*.
The Algorithm that is used to find the 3 paths are as following:

1. find the path with the closet start point to the start point given. That path will be the first path(p_1).

2. calculate the distance of all points to the start point of p_1.

3. if p_1 has $x > y$, pick two paths with the closet start point where their x is also $> y$. Those two path become p_2 and p_3.

4. if p_1 has $x <= y$, pick two paths with the closet start point where their x is also $<= y$. Those two path become p_2 and p_3.

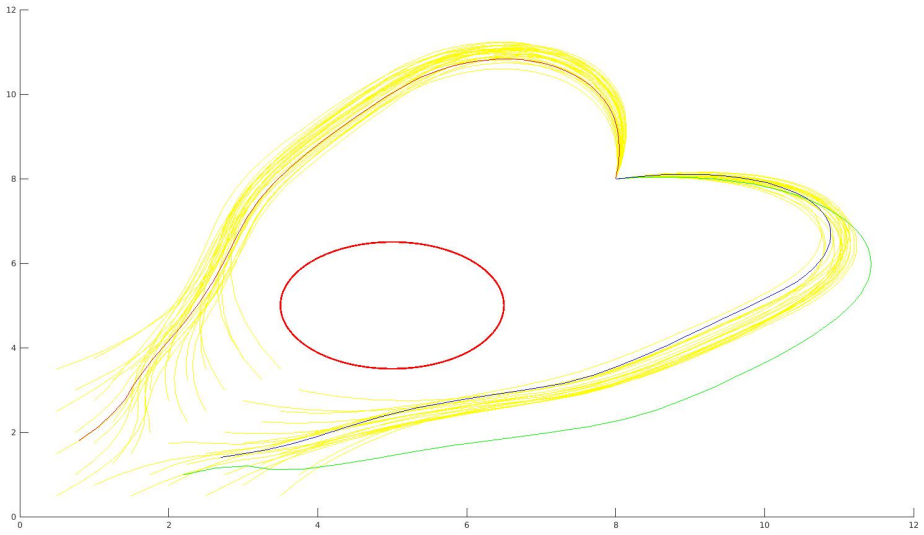
To get the alpha, I used the linear equation in $9(a)$ and plugged in the values from p_1,p_2,p_3 and the starting point. Following is the equation used.

$$\begin{pmatrix} x^{p_1} & x^{p_2} & x^{p_3} \\ y^{p_1} & y^{p_2} & y^{p_3} \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} startPoint_x \\ startPoint_y \\ 1 \end{pmatrix}$$

Currently, the time scale is 0 to 1, where the interval is decided by the user(currently is 0.001). I picked 0 to 1 because how it's commonly use in the computer graphics area as a time scale for drawing curves. I use linear interpolation for each time step.

9(c)

The following is the plot of the interpolate path. The given paths are in yellow. The path with the starting point$(0.8, 1.8)$ is the path colored red. The path with the starting point $(2.2, 1.0)$ is the path colored green. the path with the starting point $(2.7, 1.4)$ is the path colored blue.



9(d)

If you observe the graph, you will notice that certain interpolate path goes further out that the given paths. This would be a problem if there are more obstacles that are introduced near the path. One possible addition to the algorithm is to recalculate the weights and interpolation path after a certain interval.