Problem 1

(a) The solution is provided in the python script `newtonDD.py` . It implements Newton's Divided differential method of interpolating a polynomial as described in the class using a table based coefficient calculation.

(b) Given the values of e^{-x}, the interpolated value at $x = \frac{1}{3}$ according to the implementation is : $y = 0.71653$, which is accurate up to $\epsilon = 0.00005$

(c) Using the procedure in `newtonDD.py`

- The value of $f(0.05)$ with $n = 2$ is 0.99765
- The value of $f(0.05)$ with $n = 4$ is 0.99014
- The value of $f(0.05)$ with $n = 40$ is 0.96154
- The actual value of $f(0.05)$ on calculation from the function is: 0.96154

(d)
- Maximum error for n = 2 is 0.57375
- Maximum error for n = 4 is 0.3853
- Maximum error for n = 6 is 0.48888
- Maximum error for n = 8 is 0.73188
- Maximum error for n = 10 is 1.17686
- Maximum error for n = 12 is 1.96915
- Maximum error for n = 14 is 3.38075
- Maximum error for n = 16 is 5.91058
- Maximum error for n = 18 is 10.47864
- Maximum error for n = 20 is 18.75225
- Maximum error for n = 40 is 8497.58993

While it was expected that the errors reduce the interpolating polynomial degree is increased, it so happens that the max errors increases. This is because for the function given, the magnitude of the nth degree derivatives increase /oscillate rapidly as n increases. This phenomenon [1] occurs only for certain functions such as the one given above. This problem can be mitigated by choosing the spacing between interpolating points in a wise manner, such as reducing the spacing between interpolation points near the edge of the interval (Chebyshev nodes).

Problem 2

We have been given that the points are evenly spaced between an interval $[\frac{-\pi}{2}, \frac{3\pi}{2}]$, i.e., $h = \frac{2\pi}{n}$ where h is the interval size, and n is the number of points.

Now, error between an interpolating polynomial $p_n(x)$ and function $f(x)$ is given as

$$\begin{aligned} e_n(x) &= f(x) - p_n(x) \\ &= \frac{f^{n+1}(\xi)}{(n+1)!}\prod_{i=0}^{n}(x - x_i) \qquad \text{where, } \xi \in \left[\frac{-\pi}{2}, \frac{3\pi}{2}\right] \end{aligned}$$

(a) Now for linear interpolation we need the worst case error, which provides 6 digits of accuracy, therefore

$$\begin{aligned} 5 \times 10^{-7} &> \max_{\frac{-\pi}{2} \le x \le \frac{3\pi}{2}} \left| \frac{f^2(\xi)}{2!}(x - x_{i-1})(x - x_i) \right| \\ &> \max_{\frac{-\pi}{2} \le x \le \frac{3\pi}{2}} \left| \frac{f^2(\xi)}{2!} \right| \cdot \max_{\frac{-\pi}{2} \le x \le \frac{3\pi}{2}} |(x - x_{i-1})(x - x_i)| \\ &> \max_{\frac{-\pi}{2} \le x \le \frac{3\pi}{2}} \left| \frac{-\cos x}{2} \right| \cdot \max_{0 \le y \le h} |(y - h) \cdot y| \end{aligned} \tag{1}$$

since, $\frac{d^2 \cos x}{dx^2} = -\cos x$ and since points are at an interval h = $\frac{1}{n}$, doing a change of variables, we get equation 1 Now, we know that the max of $\cos x$ for any interval is bounded by +1. Therefore we have

$$5 \times 10^{-7} > \frac{1}{2} \cdot \max_{0 \le y \le h} |(y - h) \cdot y| \tag{2}$$

To find the maximum y, which maximizes $y^2 - h \cdot y$ we need to differentiate w.r.t to y and equate it to 0 i.e.,

$$\begin{aligned} \frac{\partial(y^2 - h \cdot y)}{\partial y} &= 2 \cdot y - h = 0 \\ \implies y &= \frac{h}{2} \end{aligned} \tag{3}$$

substituting in equation 2, we obtain

$$\begin{aligned} 5 \times 10^{-7} &> \frac{1}{2} \cdot \left| \frac{h^2}{4} - \frac{h^2}{2} \right| \\ \implies \frac{h^2}{8} &< 5 \times 10^{-7} \\ \implies h^2 < 4 \times 10^{-6} &\implies h < 2 \times 10^{-3} \end{aligned}$$

now, since $n = \frac{2\pi}{h} = \frac{2\pi}{2e-3} = 3141.2$ therefore, we need at least 3143 points to interpolate with the required accuracy using linear interpolation

(b) For quadratic interpolation, we need the worst case error, which provides 6 digits of accuracy, therefore

$$\begin{aligned} 5 \times 10^{-7} &> \max_{\frac{-\pi}{2} \le x \le \frac{3\pi}{2}} \left| \frac{f^3(\xi)}{3!}(x - x_{i-1})(x - x_i)(x - x_{i+1}) \right| \\ &> \max_{\frac{-\pi}{2} \le x \le \frac{3\pi}{2}} \left| \frac{f^3(\xi)}{3!} \right| \cdot \max_{\frac{-\pi}{2} \le x \le \frac{3\pi}{2}} |(x - x_{i-1})(x - x_i)(x - x_{i+1})| \\ &> \max_{\frac{-\pi}{2} \le x \le \frac{3\pi}{2}} \left| \frac{+\sin x}{6} \right| \cdot \max_{0 \le y \le h} |(y - h) \cdot y \cdot (y + h)| \end{aligned} \tag{4}$$

since $\frac{d^3 \cos x}{dx^3} = \sin x$ and for cubic interpolation, we take three consecutive points equally spaced with an interval $h = \frac{1}{n}$, we obtain equation 4.

We know that the max value of $\sin x$ is bounded by 1, therefore we have:

$$5 \times 10^{-7} > \frac{1}{6} \cdot \max_{0 \le y \le h} |(y - h) \cdot y \cdot (y + h)| \tag{5}$$

To find the maximum y, which maximizes RHS of equation 5, we will differentiate w.r.t y and equate it to 0 i.e.,

$$\begin{aligned} \frac{\partial (y - h)(y^2 + hy)}{\partial y} &= (y - h)(2y + h) + (y^2 + hy) = 0 \\ &\implies 3y^2 - h^2 = 0 \implies y = \frac{h}{\sqrt{3}} \end{aligned} \tag{6}$$

Now substituting in equation 5 we have

$$\begin{aligned} 5 \times 10^{-7} &> \frac{1}{6} \frac{h}{\sqrt{3}} \left(\frac{h}{\sqrt{3}} - h \right) \left(\frac{h}{\sqrt{3}} + h \right) \\ &> \frac{1}{6} \frac{2h^3}{3\sqrt{3}} \\ \implies h &< 0.019826 \end{aligned} \tag{7}$$

Now, this means that we need at least $n = \frac{2\pi}{h} = 316.9$ implying, we need 318 points to interpolate using quadratic interpolation for 6 digits of accuracy

Problem 3

The solution to this problem is provided in `q3.py`. The roots of the function $tan(x) - x$ is as follows:

$$(x_{low}, x_{high}) = (4.493409, 7.725252)$$

Problem 4

For newton's method, the update rule for each iteration is given as:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{8}$$

We're given that ξ is a root of $f(x)$, now we can write from Taylor's series about the root, that:

$$f(\xi + \epsilon) = f(\xi) + \epsilon f'(\xi) + \frac{\epsilon^2}{2!} f''(\xi) + \frac{\epsilon^3}{3!} f'''(\xi) \tag{9}$$

where ϵ is the error given as:

$$\begin{aligned} \epsilon_n &= \xi - x_n \\ \implies x_n &= \xi - \epsilon_n \end{aligned}$$

Replacing this in equation 8, we have

$$\begin{aligned} \epsilon_{n+1} &= \epsilon_n + \frac{f(x_n)}{f'(x_n)} \\ \epsilon_{n+1} &= \epsilon_n + \frac{f(\xi - \epsilon_n)}{f'(\xi - \epsilon_n)} \end{aligned} \tag{10}$$

We can now write equation 10 using taylor's expansion as given in equation 9 as:

$$\epsilon_{n+1} = \epsilon_n + \frac{f(\xi) - \epsilon_n f'(\xi) + \frac{\epsilon_n^2}{2!} f''(\xi) - \frac{\epsilon_n^3}{3!} f'''(\xi) + \dots}{f'(\xi) - \epsilon_n f''(\xi) + \frac{\epsilon_n^2}{2!} f'''(\xi) - \frac{\epsilon_n^3}{3!} f''''(\xi) + \dots} \tag{11}$$

We know that $f(\xi) = 0, f'(\xi) = 0, f''(\xi) \neq 0$, Substituting in equation 11:

$$\begin{aligned} \epsilon_{n+1} &= \epsilon_n + \frac{\frac{\epsilon_n^2}{2} f''(\xi) - \frac{\epsilon_n^3}{6} f'''(\xi) + \dots}{-\epsilon_n f''(\xi) + \frac{\epsilon_n^2}{2} f'''(\xi) - \frac{\epsilon_n^3}{6} f''''(\xi) + \dots} \\ \epsilon_{n+1} &= \epsilon_n \left(1 + \frac{\frac{\epsilon_n}{2} f''(\xi) - \frac{\epsilon_n^2}{6} f'''(\xi) + \dots}{-\epsilon_n f''(\xi) + \frac{\epsilon_n^2}{2} f'''(\xi) - \frac{\epsilon_n^3}{6} f''''(\xi) + \dots} \right) \end{aligned} \tag{12}$$

Since the second iteration term in equation 12 is a fraction of same order terms, we can assume it to be a constant, and therefore we have the linear convergence of newton's method

$$\epsilon_{n+1} \approx \epsilon_n \cdot C \tag{13}$$

Suppose, we have the iteration update rule now as

$$x_{n+1} = x_n - 2\frac{f(x_n)}{f'(x_n)}$$

Using a similar analysis we get

$$\begin{aligned}
\epsilon_{n+1} &= \epsilon_n + 2\frac{f(x_n)}{f'(x_n)} \\
\epsilon_{n+1} &= \epsilon_n + 2\frac{f(\xi - \epsilon_n)}{f'(\xi - \epsilon_n)} \\
\epsilon_{n+1} &= \epsilon_n + 2 \cdot \left(\frac{\frac{\epsilon_n^2}{2} f''(\xi) - \frac{\epsilon_n^3}{6} f'''(\xi) + \dots}{-\epsilon_n f''(\xi) + \frac{\epsilon_n^2}{2} f'''(\xi) - \frac{\epsilon_n^3}{6} f''''(\xi) + \dots} \right) \\
\epsilon_{n+1} &= \left(\frac{\cancel{-\epsilon_n{}^2 f''(\xi)} + \frac{\epsilon_n{}^3}{2} f'''(\xi) \cdots + \cancel{\epsilon_n{}^2 f''(\xi)} - \frac{\epsilon_n{}^3}{3} f'''(\xi) + \dots}{-\epsilon_n f''(\xi) + \frac{\epsilon^2}{2} f'''(\xi) - \frac{\epsilon^3}{6} f''''(\xi) + \dots} \right) \\
\epsilon_{n+1} &= \epsilon_n{}^2 \left(\frac{\frac{\epsilon_n}{2} f'''(\xi) \cdots - \frac{\epsilon_n}{3} f'''(\xi) + \dots}{-\epsilon_n f''(\xi) + \frac{\epsilon_n^2}{2} f'''(\xi) - \frac{\epsilon_n^3}{6} f''''(\xi) + \dots} \right)
\end{aligned} \tag{14}$$

We can now take the term in the parentheses in equation 14 as a constant, and we then have the quadratic convergence:

$$\epsilon_{n+1} \approx \epsilon_n^2 \cdot C \tag{15}$$

Problem 5

(a) Muller's method is implemented using python in `q5.py`

(b) For the polynomial given

$$p(x) = x^3 - 5x^2 + 11x - 15$$

Roots found using the Muller's method implementation are as given below:

1. $x = 3$
2. $x = 1 + 2j$
3. $x = 1 - 2j$

Problem 6

(a) Through sylvester's method, we know that there exists a common root, since for the given matrix below

$$det(\mathbf{Q}) = det \begin{pmatrix} 1 & -4 & 6 & -4 & 0 \\ 0 & 1 & -4 & 6 & -4 \\ 1 & 2 & -8 & 0 & 0 \\ 0 & 1 & 2 & -8 & 0 \\ 0 & 0 & 1 & 2 & -8 \end{pmatrix} = 0$$

(b) We can find the root for the two univariate polynomials as follows:

$$\frac{x_i}{x_j} = (-1)^{i+j} \cdot \frac{det(\mathbf{A}_i)}{det(\mathbf{A}_j)}$$

$$\frac{x^4}{x^3} = (-1)^{1+2} \cdot \frac{det(\mathbf{Q}_1)}{det(\mathbf{Q}_2)}$$

$$x = (-1) \cdot \frac{\begin{vmatrix} -4 & 6 & -4 & 0 \\ 1 & -4 & 6 & -4 \\ 2 & -8 & 0 & 0 \\ 0 & 1 & 2 & -8 \end{vmatrix}}{\begin{vmatrix} 1 & 6 & -4 & 0 \\ 0 & -4 & 6 & -4 \\ 1 & -8 & 0 & 0 \\ 0 & 2 & -8 & 0 \end{vmatrix}}$$

$$x = (-1) \cdot \frac{832}{-416} \implies x = 2$$

Problem 7

(a) The contour plot for the functions $p(x, y) = 2x^2 + 2y^2 + 4x + 4y + 3$ and $q(x, y) = x^2 + y^2 + 2xy + 3x + 5y + 4$ is generated using matplotlib in q7.py and is as given below in Fig. 1.

(b) Using the sylvester's method by considering the variable x as a constant, we have the following matrix $\mathbf{Q}$ which is required to be singular i.e., $det(\mathbf{Q}) = 0$

$$det(\mathbf{Q}) = \begin{vmatrix} 2 & 4 & 2x^2 + 4x + 3 & 0 \\ 0 & 2 & 4 & 2x^2 + 4x + 3 \\ 1 & 2x + 5 & x^2 + 3x + 4 & 0 \\ 0 & 1 & 2x + 5 & x^2 + 3x + 4 \end{vmatrix} = 0$$

$$\implies 16x^4 + 80x^3 + 144x^2 + 100x + 19 = 0$$

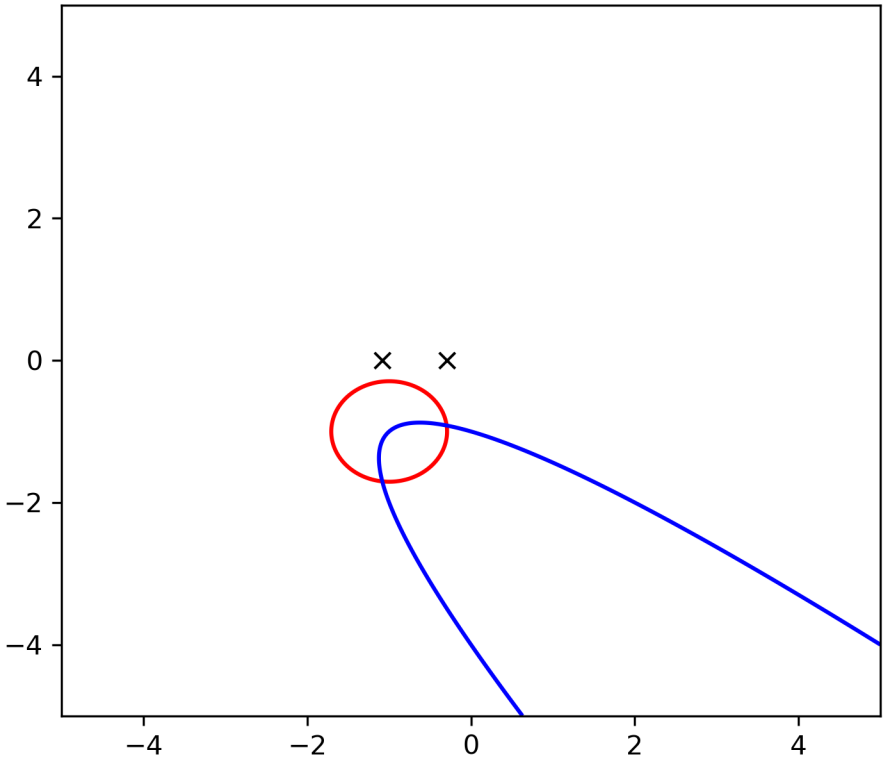

Figure 1: Contour plot for $p(x, y)$ and $q(x, y)$

The real roots of this equation are:

$$x_1 = \frac{1}{4}(-5 + \sqrt{5} - \sqrt{2\sqrt{5} - 1}) \qquad = -1.08405$$
$$x_2 = \frac{1}{4}(-5 + \sqrt{5} + \sqrt{2\sqrt{5} - 1}) \qquad = -0.29791$$

Now plugging in the values of x_1 in given $P(x, y)$ and $Q(x, y)$ we have:

$$P(y) = 2y^2 + 4y + 1.01413 \qquad \text{with root } x_1$$
$$Q(y) = y^2 + 2.83188 * y + 1.92301 \qquad \text{with root } x_1$$

We can find the common root of $P(y)$ and $Q(y)$ above using the ratio method as follows:

$$\frac{y_i}{y_j} = (-1)^{i+j} \cdot \frac{det(\mathbf{Q}_i)}{det(\mathbf{Q}_j)}$$

$$y = (-1) \cdot \frac{\begin{vmatrix} 4 & 1.01413 & 0 \\ 2 & 4 & 1.01413 \\ 2.83188 & 1.92301 & 0 \end{vmatrix}}{\begin{vmatrix} 2 & 1.01413 & 0 \\ 0 & 4 & 1.01413 \\ 1 & 1.92301 & 0 \end{vmatrix}}$$

$$y = (-1) \cdot \frac{-4.88825}{-2.87190} \implies y = -1.70209$$

Similarly plugging in the values of x_2 in the given $P(x, y)$ and $Q(x, y)$:

$$P(y) = 2y^2 + 4y + 1.98587 \qquad \text{with root } x_2$$
$$Q(y) = y^2 + 4.40419 * y + 3.19503 \qquad \text{with root } x_2$$

We can again find the other common root of P(y) and Q(y) as follows:

$$y = (-1) \cdot \frac{\begin{vmatrix} 4 & 1.98587 & 0 \\ 2 & 4 & 1.98587 \\ 4.40419 & 3.19503 & 0 \end{vmatrix}}{\begin{vmatrix} 2 & 1.98587 & 0 \\ 0 & 4 & 1.98587 \\ 1 & 3.19503 & 0 \end{vmatrix}}$$

$$y = (-1) \cdot \frac{-8.010942}{-8.74615} \implies y = -0.915939$$

Finally the common roots of the equation are

$$x_1, y_1 = -1.08405, -1.70209$$
$$x_2, y_2 = -0.29791, -0.915939$$

Problem 8

(a) We need to obtain a linear system with constraints to determine, whether a Point $P = (x, y) = \mathbf{x}$ lies within a triangle formed by 3 other points A, B and C given by $(x_i, y_i), (x_j, y_j)$ and (x_k, j_k) respectively. [2]

We can pick two edges of the triangle with A as common vertex, then we have the edges as vectors $(C-A)$and $(B-A)$. Any point inside the triangle may be now represented as starting from the vertex A and then forming a linear combination of $(B-A)$ and $(C-A)$ i.e., we have

$$P = A + (B-A)\cdot u + (C-A)\cdot v$$

Now, if $u < 0$ or $v < 0$ it means that the point could move on the negative side of triangle, which is not desired. Further, if $u > 1$ or $v > 1$ then it means that P would be larger than the lengths of the edges, resulting in point being outside triangle. Finally, if $u + v > 1$ then it would mean P is beyond the edge BC. We can now write

$$(P-A) = (B-A)\cdot u + (C-A)\cdot v$$

if we represent $(B-A) = \mathbf{v_0}, (C-A) = \mathbf{v_1}$ and $(P-A) = \mathbf{v_2}$ then we have:

$$\mathbf{v_2} = u\cdot\mathbf{v_0} + v\cdot\mathbf{v_1} \tag{16}$$

We can dot multiply both sides of the equation 16, with v_0 and v_1 to get two distinct equations as follows.

$$\mathbf{v_2}\cdot\mathbf{v_0} = u\cdot(\mathbf{v_0}\cdot\mathbf{v_0}) + v\cdot(\mathbf{v_1}\cdot\mathbf{v_0})$$
$$\mathbf{v_2}\cdot\mathbf{v_1} = u\cdot(\mathbf{v_0}\cdot\mathbf{v_1}) + v\cdot(\mathbf{v_1}\cdot\mathbf{v_1})$$

This can be represented in matrix form as follows:

$$\begin{pmatrix} \mathbf{v_0}\cdot\mathbf{v_0} & \mathbf{v_1}\cdot\mathbf{v_0} \\ \mathbf{v_0}\cdot\mathbf{v_1} & \mathbf{v_1}\cdot\mathbf{v_1} \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \mathbf{v_2}\cdot\mathbf{v_0} \\ \mathbf{v_2}\cdot\mathbf{v_1} \end{pmatrix} \tag{17}$$

Subject to $u > 0$ and $v > 0$ and $u + v < 1$

(b) The solution to this problem is given in `q8.py`

(c) The choices made are as listed below:

- **Picking triplets of points**: the start points of each path, were classified as left or right about the $x = y$ line, and then sorted according to the distance to the given starting points. The closest path starting point was kept fixed, and then pairs of points were searched for which would satisfy the triangle test. This would ensure that we first try matching triangles with shorter perimeters, by also avoiding the ring of fire, since we have classified the paths.
- **Choosing alpha**: We choose the $\alpha's$ to be inversely proportionate to the distance from the vertex of the chosen triangle. This ensures that the weighted average of the three paths, would be as close to the triangle centroid at each of the 50 timesteps. Also ensuring that we would effectively avoid the ring of fire, since all the given paths on each side of the ring of fire avoid it.

- **Timestep selection**: We chose the timestep as 0.01 arbitrarily to generate a smooth plot of the interpolated path.
- **Interpolation choice**: We chose a cubic spline interpolation method, because we needed to fit lower order polynomials in a piece-wise fashion so that the error is minimal at the interpolation points. At higher order, the polynomial would wiggle about the points causing significant error [1]. Cubic interpolation would generate very smooth curves as would be expected from a "human unicyclist"

(d) The interpolated paths are plotted as in Figure 2

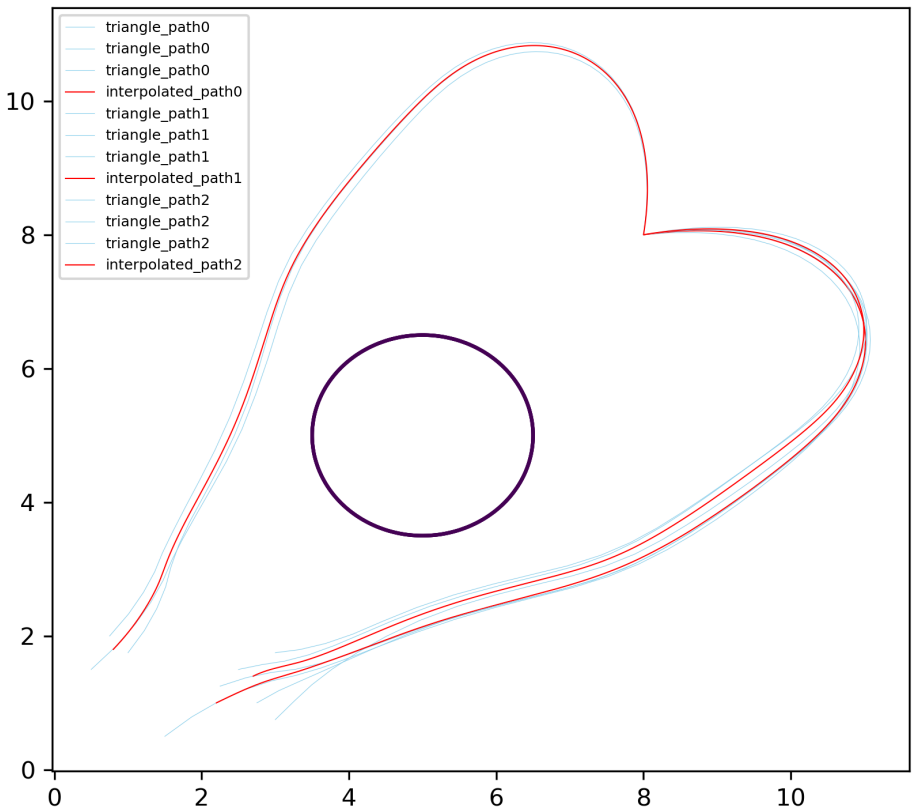

Figure 2: Plot of the paths generated by interpolating a cubic spline over the generated 50 time steps Intepolated path 1 corresponds to $(0.8, 1.8)$ path 2 corresponds to $(2.2, 1.0)$ ad path 3 corresponds to $(2.7, 1.4)$

(e) The algorithm used in this case, makes use of the structure of the environment and the data that is provided, and classifies the points about the $x = y$ line. Now if additional

obstacles were added, we would need additional good paths to interpolate from, and we could cluster the starting points into classes such that all the paths of each cluster space produce triangles with as minimal a size (perimeter) as possible and then generate a path from the same.

Discussion Citation

For this Homework set, I have discussed with *Tarasha Khurana* andrewID: `tkhurana@andrew.cmu.edu`, and *Viraj Parimi* andrewID: `vparimi@andrew.cmu.edu`

References

[1] Runge's Phenomenon,
`https://en.wikipedia.org/wiki/Runge%27s_phenomenon`

[2] Test points in a triangle,
`https://en.wikipedia.org/wiki/Barycentric_coordinate_system`