16-811 hw 2

Erica Weng

October 29, 2020

1

To run the implementation, run `python3 code/q1.py`

1b

interpolated $cos(\pi x)$ at $x = 3$: 0.588

1c

n	p(0.07)
2	1.991180000
4	1.966875065
40	1.915525327
Actual	1.915525333

1d

n	E_n
2	0.94
4	0.60
6	0.63
8	0.77
10	1.00
12	1.35
14	1.87
16	2.64
18	3.78
20	5.47
40	290.10

yes the error estimates make sense. they get bigger as n increases, because the polynomial gets more “spiky" (high-frequency) at the non-interpolated points. I use an error estimate interval of $2/1000 = 0.005$, thus for $n = 40$, the interpolating interval is only 0.05, so in between adjacent interpolated points, the error gets really huge.

2

linear interpolation:

$$e_1(x) = \frac{f^{(2)}(\xi)}{2!}(x+h)(x)$$

$$\max_{x\in[0,2\pi]} \frac{f^{(2)}(\xi)}{2!} = \max_{x\in[0,2\pi]} \Big[-\sin(x) \Big] = \frac{1}{2}$$

$$\max_{x\in[0,h]} \Big|(x-h)(x)\Big| \to x = \frac{h}{2}; \max_{x\in[0,h]} \Big|(x-h)(x)\Big| = \frac{h^2}{4}$$

$$\max_{x\in[0,2\pi]} \Big|e_1(x)\Big| = \frac{1}{2}\cdot\frac{h^2}{4} \to h = 2\sqrt{2e_1}$$

$$e_1 < 0.0000005 \to h < 2\sqrt{2\cdot 0.0000005} \approx \boxed{0.002}$$

$$N = \frac{2\pi}{h} \approx \boxed{3142}$$

quadratic interpolation:

$$e_2(x) = \frac{f^{(3)}(\xi)}{3!}(x+h)(x)$$

$$\max_{x\in[0,2\pi]} \frac{f^{(3)}(\xi)}{3!} = \max_{x\in[0,2\pi]} \Big[-\sin(x) \Big] = \frac{1}{6}$$

$$\max_{x\in[-h,h]} \Big|(x-h)(x)(x+h)\Big| \to x = \frac{h}{\sqrt{3}}$$

$$\max_{x\in[-h,h]} \Big|(x-h)(x)(x+h)\Big| = \frac{2h^3}{3\sqrt{3}}$$

$$\max_{x\in[0,2\pi]} \Big|e_1(x)\Big| = \frac{1}{6}\cdot\frac{2h^3}{3\sqrt{3}} \to h = \sqrt[3]{9\sqrt{3}e_2}$$

$$e_2 < 0.0000005 \to h < \boxed{0.019827}$$

$$N = \frac{2\pi}{h} \approx \boxed{316.9}$$

3

To run the implementation, run `python3 code/q3.py`
two roots on either side of 15: 14.07 and 17.22

4

4 part 1

the newton's method update rule is:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

let's do a substitution:

$$x_i = \xi + \epsilon_i$$

where ξ is the true root and ϵ_i is the error on the ith iteration of Newton's method. so, substituting we get:

$$\xi + \epsilon_{i+1} = \xi + \epsilon_i - \frac{f(\xi + \epsilon_i)}{f'(\xi + \epsilon_i)}$$
$$\epsilon_{i+1} = \epsilon_i - \frac{f(\xi + \epsilon_i)}{f'(\xi + \epsilon_i)}$$

using taylor's expansion to expand out $\frac{f(\xi+\epsilon_i)}{f'(\xi+\epsilon_i)}$:

$$\frac{f(\xi + \epsilon_i)}{f'(\xi + \epsilon_i)} = \frac{f(\xi) + \epsilon_i f'(\xi) + \frac{\epsilon_i^2}{2} f''(\xi) + \frac{\epsilon_i^3}{6} f'''(\xi) + \dots}{f'(\xi) + \epsilon_i f''(\xi) + \frac{\epsilon_i^2}{2} f'''(\xi) + \frac{\epsilon_i^3}{6} f''''(\xi) + \dots}$$

So Newton's update rule becomes:

$$\epsilon_{i+1} = \epsilon_i - \left[\frac{f(\xi) + \epsilon_i f'(\xi) + \frac{\epsilon_i^2}{2} f''(\xi) + \frac{\epsilon_i^3}{6} f'''(\xi) + \dots}{f'(\xi) + \epsilon_i f''(\xi) + \frac{\epsilon_i^2}{2} f'''(\xi) + \frac{\epsilon_i^3}{6} f''''(\xi) + \dots} \right] \tag{1}$$

Cancel out 0 terms:

$$\epsilon_{i+1} = \epsilon_i - \left[\frac{\frac{\epsilon_i^2}{2} f''(\xi) + \frac{\epsilon_i^3}{6} f'''(\xi) + \dots}{\epsilon_i f''(\xi) + \frac{\epsilon_i^2}{2} f'''(\xi) + \frac{\epsilon_i^3}{6} f''''(\xi) + \dots} \right] \tag{2}$$

Do some rearranging and cancel out an ϵ_i from top and bottom:

$$\begin{aligned}
\epsilon_{i+1} &= \epsilon_i - \epsilon_i \left[\frac{\frac{\epsilon_i}{2} f''(\xi) + \frac{\epsilon_i^2}{6} f'''(\xi) + \dots}{\epsilon_i f''(\xi) + \frac{\epsilon_i^2}{2} f'''(\xi) + \frac{\epsilon_i^3}{6} f''''(\xi) + \dots} \right] \\
&= \epsilon_i \left(1 - \left[\frac{\frac{\epsilon_i}{2} f''(\xi) + \frac{\epsilon_i^2}{6} f'''(\xi) + \dots}{\epsilon_i f''(\xi) + \frac{\epsilon_i^2}{2} f'''(\xi) + \frac{\epsilon_i^3}{6} f''''(\xi) + \dots} \right] \right) \\
&= \epsilon_i \left(1 - \left[\frac{\frac{1}{2} f''(\xi) + \frac{\epsilon_i}{6} f'''(\xi) + \dots}{f''(\xi) + \frac{\epsilon_i}{2} f'''(\xi) + \frac{\epsilon_i^2}{6} f''''(\xi) + \dots} \right] \right) \\
\epsilon_{i+1} &= \epsilon_i \cdot C
\end{aligned} \tag{3}$$

thus newton's method converges linearly

4 part 2

this part is basically the same as the update rule from equation 2 (with zero-elements eliminated) except with an additional coefficient of 2:

$$\epsilon_{i+1} = \epsilon_i - 2\left[\frac{\frac{\epsilon_i^2}{2}f''(\xi) + \frac{\epsilon_i^3}{6}f'''(\xi) + \dots}{\epsilon_i f''(\xi) + \frac{\epsilon_i^2}{2}f'''(\xi) + \frac{\epsilon_i^3}{6}f''''(\xi) + \dots}\right] \tag{4}$$

Simplifying, rearranging, and cancelling out terms:

$$\begin{aligned}
\epsilon_{i+1} &= \epsilon_i - \epsilon_i\left[\frac{\epsilon_i f''(\xi) + \frac{\epsilon_i^2}{3}f'''(\xi) + \dots}{\epsilon_i f''(\xi) + \frac{\epsilon_i^2}{2}f'''(\xi) + \dots}\right] \\
&= \epsilon_i - \epsilon_i\left[\frac{f''(\xi) + \frac{\epsilon_i}{3}f'''(\xi) + \dots}{f''(\xi) + \frac{\epsilon_i}{2}f'''(\xi) + \dots}\right] \\
&= \epsilon_i\left(1 - \left[\frac{f''(\xi) + \frac{\epsilon_i}{3}f'''(\xi) + \dots}{f''(\xi) + \frac{\epsilon_i}{2}f'''(\xi) + \frac{\epsilon_i^2}{6}f''''(\xi) + \dots}\right]\right) \\
&= \epsilon_i\left[\frac{\frac{\epsilon_i}{6}f'''(\xi) + \dots}{f''(\xi) + \frac{\epsilon_i}{2}f'''(\xi) + \frac{\epsilon_i^2}{6}f''''(\xi) + \dots}\right] \\
&= \epsilon_i^2\left[\frac{\frac{1}{6}f'''(\xi) + \dots}{f''(\xi) + \frac{\epsilon_i}{2}f'''(\xi) + \frac{\epsilon_i^2}{6}f''''(\xi) + \dots}\right] \\
\epsilon_{i+1} &= \epsilon_i^2 \cdot C
\end{aligned} \tag{5}$$

thus newton's method converges quadratically

5

5a

code in q5.py

5b

roots: $0.341 \pm 1.162j$, -0.682

6

6a

code used in q6.py

if the determinant of our Q resultant matrix (constructed by multiplying our given equations the by powers of x), then our given equations have a shared common root:

$$Q = \begin{pmatrix} 1 & -3 & 1 & -3 & 0 \\ 0 & 1 & -3 & 1 & -3 \\ 1 & 1 & -12 & 0 & 0 \\ 0 & 1 & 1 & -12 & 0 \\ 0 & 0 & 1 & 1 & -12 \end{pmatrix}$$

$det(Q) = 0$, so yes there's common root

6b

using the ratio method learned in class, the root $= \frac{-detQ_1}{detQ_2}$.

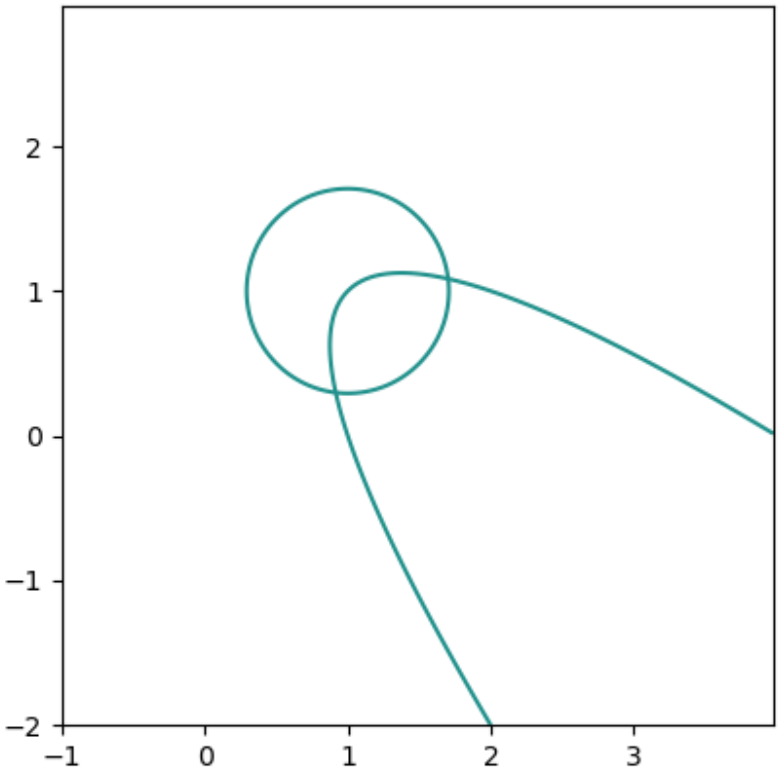
$$Q_1 = \begin{pmatrix} -3 & 1 & -3 & 0 \\ 1 & -3 & 1 & -3 \\ 1 & -12 & 0 & 0 \\ 1 & 1 & -12 & 0 \end{pmatrix}$$

$$Q_2 = \begin{pmatrix} 1 & 1 & -3 & 0 \\ 0 & -3 & 1 & -3 \\ 1 & -12 & 0 & 0 \\ 0 & 1 & -12 & 0 \end{pmatrix}$$

so the root $x = \frac{-detQ_1}{detQ_2} = 3$

7

7a



7b

$$p(y) = 2x^2 - 4x + (2y^2 - 4y + 3)$$

$$q(y) = x^2 + (2y - 5)x + (y^2 - 3y + 4)$$

$$det\ Q = det \begin{pmatrix} 2 & -4 & 2y^2 - 4y + 3 & 0 \\ 0 & 2 & -4 & 2y^2 - 4y + 3 \\ 1 & 2y - 5 & y^2 - 3y + 4 & 0 \\ 0 & 1 & 2y - 5 & y^2 - 3y + 4 \end{pmatrix} = 16y^4 - 80y^3 + 144y^2 - 100y + 19 = 0$$

$$y = \frac{5 - \sqrt{5} \pm \sqrt{2(\sqrt{5} - 1)}}{4}$$

plugging y into $p(x, y)$ to get x:

$$x = \frac{4 \pm \sqrt{2\left(2 \mp \sqrt{2(\sqrt{5}-1)} \pm \sqrt{10(\sqrt{5}-1)}\right)}}{4}$$

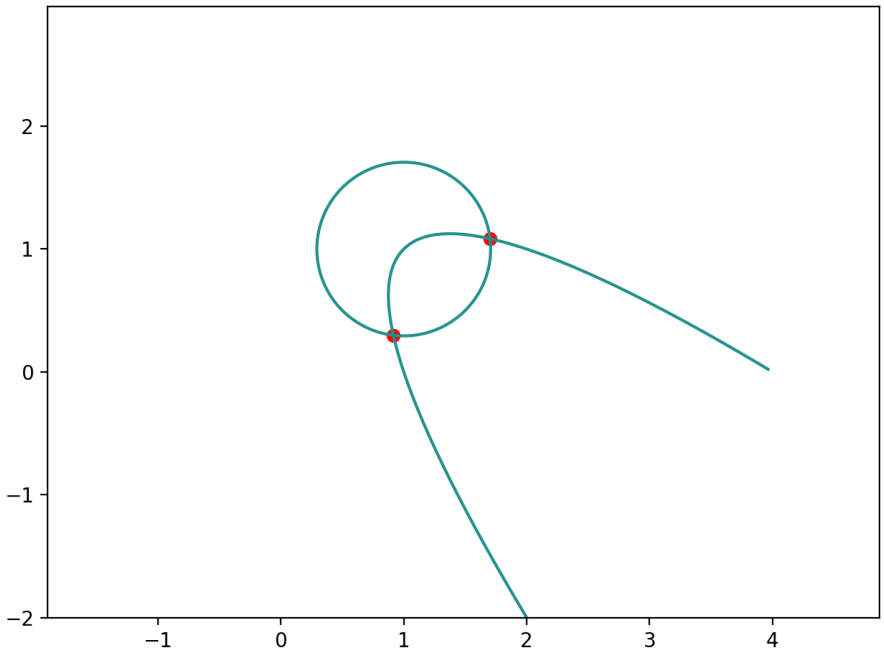
thus, our two simultaneous zeros:

$$\left(\frac{4 + \sqrt{2\left(2 - \sqrt{2(\sqrt{5}-1)} + \sqrt{10(\sqrt{5}-1)}\right)}}{4}, \frac{5 - \sqrt{5} + \sqrt{2(\sqrt{5}-1)}}{4}\right)$$

and

$$\left(\frac{4 - \sqrt{2\left(2 + \sqrt{2(\sqrt{5}-1)} - \sqrt{10(\sqrt{5}-1)}\right)}}{4}, \frac{5 - \sqrt{5} - \sqrt{2(\sqrt{5}-1)}}{4}\right)$$

7c



verified

8

8a

Let's consider the vectors from the origin defined by the three points $(x^{(i)}, y^{(i)})$, $(x^{(j)}, y^{(j)})$, and $(x^{(k)}, y^{(k)})$: call them v_i, v_j, and v_k respectively. any interpola-

tion v_p of these three vectors, $v_p = \alpha_i v_i + \alpha_j v_j + \alpha_k v_k$, with $\alpha_i + \alpha_j + \alpha_k = 1$ and $\alpha_i, \alpha_j, \alpha_k \geq 0$, will be a vector from the origin whose endpoint lies within the triangle defined by the endpoints of v_i, v_j, and v_k.

So we have the linear system $Av = b$:

$$A = \begin{pmatrix} x^{(i)} & x^{(j)} & x^{(k)} \\ y^{(i)} & y^{(j)} & y^{(k)} \\ 1 & 1 & 1 \end{pmatrix}; \quad v = \begin{pmatrix} \alpha_i \\ \alpha_j \\ \alpha_k \end{pmatrix}; \quad b = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

with the constraint $\alpha_i, \alpha_j, \alpha_k \geq 0$.

8b

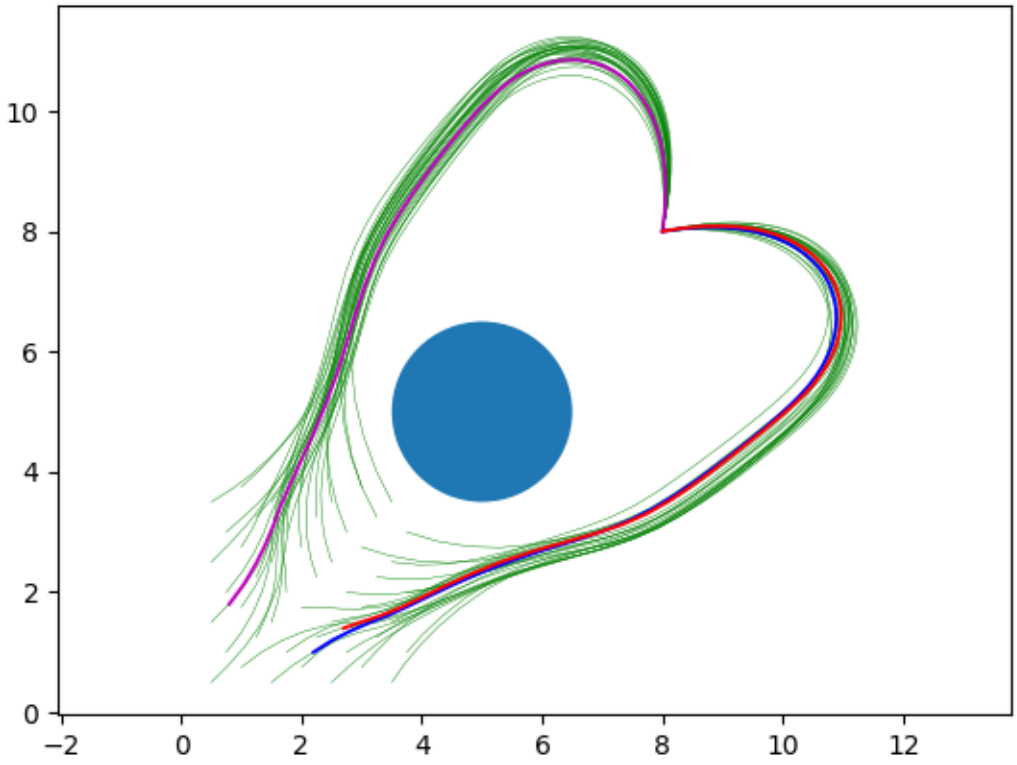
run `code/q8.py`. change the value of `point` in the `main` function to test different starting points. the path will be plotted

Note: I used sklearn for calculating distance

8c

I picked the triple by sorting the paths by distance to the given starting point, and successively iterated thorough all triples from closest to furthest until I found one that worked. "worked" means: 1. starting point must be within the triangle defined by the initial triple of points, according to the system defined in 8a 2. for all t, the interpolated path did not go through the ring of fire (no linear interpolation between any two points generated from the interpolation of three paths can intersect the circle: i used distance formula between line and point to check this `https://math.stackexchange.com/questions/275529/check-if-line-intersects-with-circles-perimeter`

8d



8e

i would need to not only check for interpolated path intersection with the ring of fire, but also with any other obstacles that are introduced