Introduction to Numerical Methods and Applications Homework #1

Student ID: B0930940003 Name: 鄭璟翰

1. (5%)By 3^{th} Order Taylor Approximation

$$f(x_{i+1}) = f(x_i) + f'(x_i)h + \frac{f''(x_i)h^2}{2!} + R_3^{(1)}..........(1)$$
$$f(x_{i-1}) = f(x_i) + f'(x_i)(-h) + \frac{f''(x_i)(-h)^2}{2!} + R_3^{(2)}...(2)$$

by (1)-(2),

$$f(x_{i+1}) - f(x_{i-1}) = f'(x_i)2h + (R_3^{(1)} - R_3^{(2)})$$
$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h} - \frac{R_3^{(1)} - R_3^{(2)}}{2h}$$
$$R_3^{(1)} = f^{(3)}(\xi_1)\frac{h^3}{3!} \text{ and } R_3^{(2)} = f^{(3)}(\xi_2)\frac{(-h)^3}{3!}, \text{ so } \frac{R_3^{(1)} - R_3^{(2)}}{2h} = \overline{O}(h^2)$$

Thus,

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h} + O(h^2)$$

2. a. (10%)Graphical method to guess all roots : -2.08, -1.00, -0.35, 0.30, 1.00.

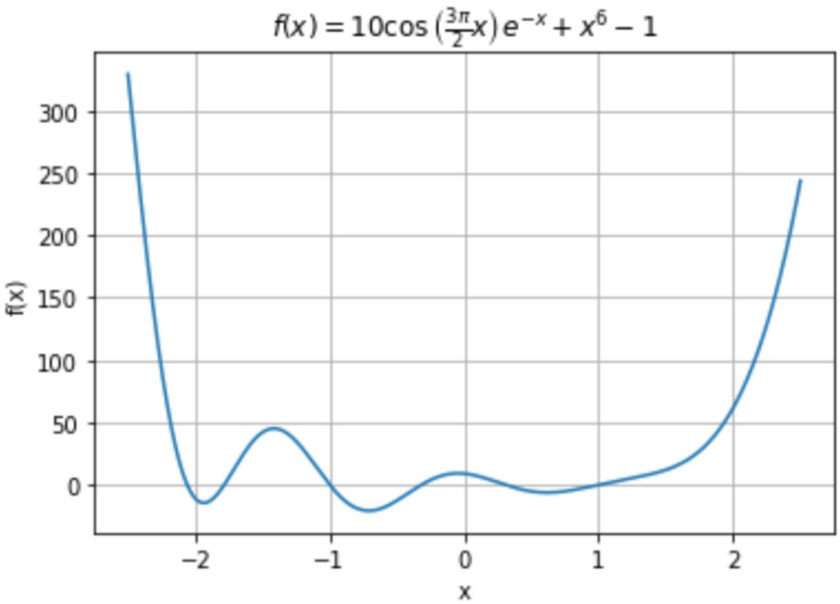

Figure 1: $f(x) = 10\cos\left(\frac{3\pi}{2}x\right)e^{-x} + x^6 - 1$

In $(2.5, \infty)$, $\lim_{x\to\infty} 10\cos\left(\frac{3\pi}{2}x\right)e^{-x} = 0$ ($|\cos x| <= 1$) and $\lim_{x\to\infty} x^6 = \infty$. So $\lim_{x\to\infty} f(x) = \infty$, no other root in $(2.5, \infty)$; In $(-\infty, -2.5)$, $\lim_{x\to-\infty} x^6 = 0$, $\lim_{x\to-\infty} e^{-x} = \infty$ and $\lim_{x\to-\infty} 10\cos\left(\frac{3\pi}{2}x\right)$ is oscillatory, so $10\cos\left(\frac{3\pi}{2}x\right)e^{-x}$ may close to 0. Thus, there exits other roots in $(-\infty, -2.5)$.

b. (10%) Bisection Approximation: x=0.3042

Iteration	xl	xu	xr	er(%)
1	0.0000	0.5000	0.2500	100.000
2	0.2500	0.5000	0.3750	33.333
3	0.2500	0.3750	0.3125	20.000
4	0.2500	0.3125	0.2812	11.111
5	0.2812	0.3125	0.2969	5.263
6	0.2969	0.3125	0.3047	2.564
7	0.2969	0.3047	0.3008	1.299
8	0.3008	0.3047	0.3027	0.645
9	0.3027	0.3047	0.3037	0.322
10	0.3037	0.3047	0.3042	0.161

Figure 2: Each iteration result using bisection in $[0, 0.5]$

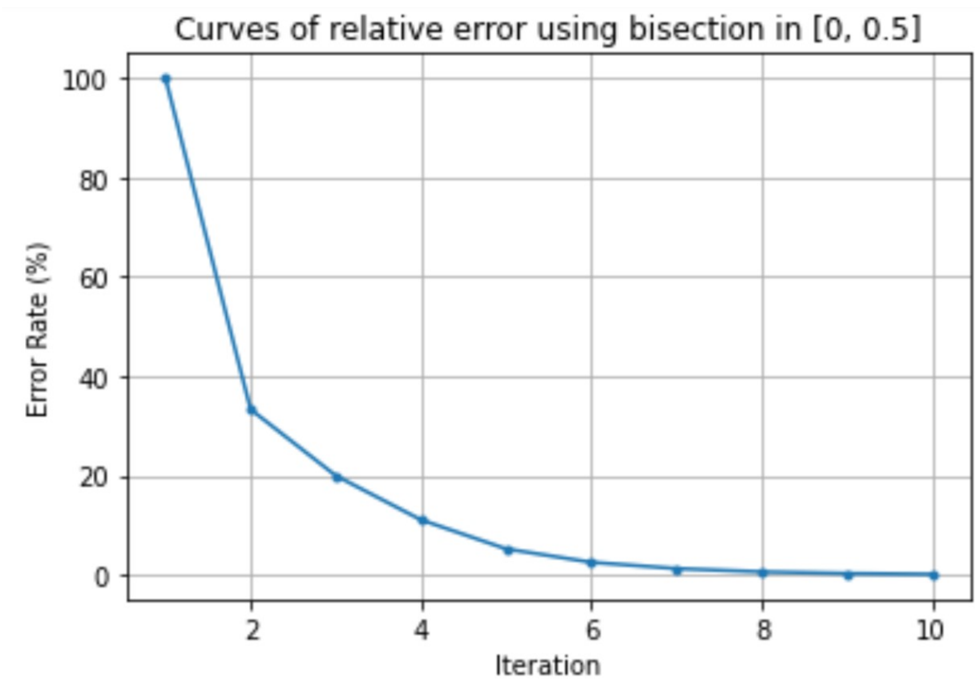

Figure 3: Curves of relative error using bisection in $[0, 0.5]$

False Position Approximation: x=0.3045

Iteration	xl	xu	xr	er(%)
1	0.0000	0.5000	0.3153	100.000
2	0.0000	0.3153	0.3025	4.211
3	0.3025	0.3153	0.3045	0.647
4	0.3025	0.3045	0.3045	0.005

Figure 4: Each iteration result using false position in $[0, 0.5]$

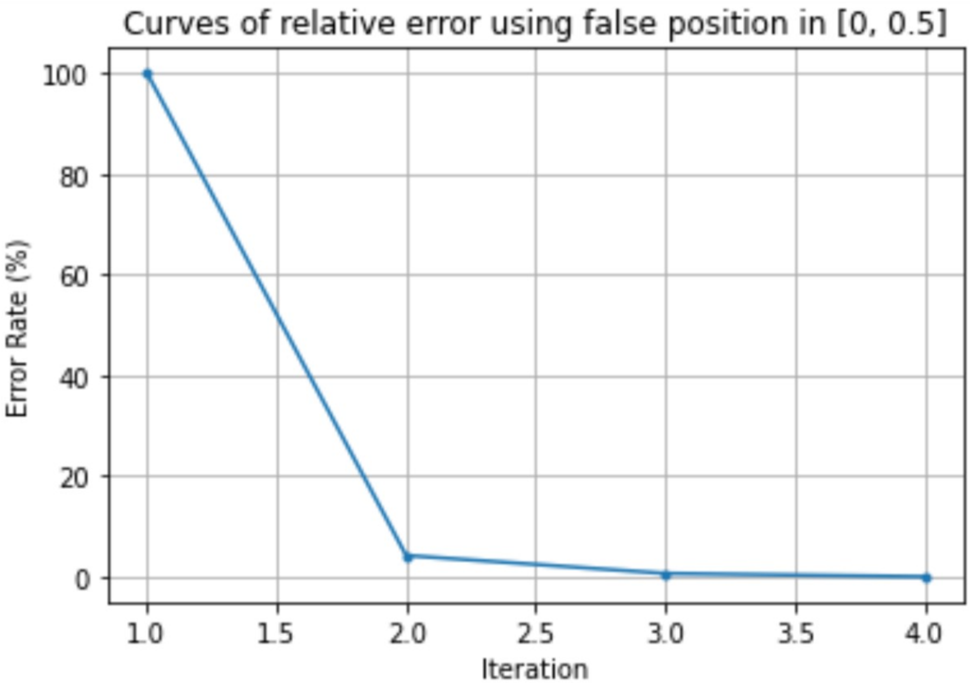

Figure 5: Curves of relative error using false position in $[0, 0.5]$

c. (10%) Bisection Approximation: x=0.9995

Iteration	xl	xu	xr	er(%)
1	0.5000	2.0000	1.2500	100.000
2	0.5000	1.2500	0.8750	42.857
3	0.8750	1.2500	1.0625	17.647
4	0.8750	1.0625	0.9688	9.677
5	0.9688	1.0625	1.0156	4.615
6	0.9688	1.0156	0.9922	2.362
7	0.9922	1.0156	1.0039	1.167
8	0.9922	1.0039	0.9980	0.587
9	0.9980	1.0039	1.0010	0.293
10	0.9980	1.0010	0.9995	0.147

Figure 6: Each iteration result using bisection in $[0, 0.5]$

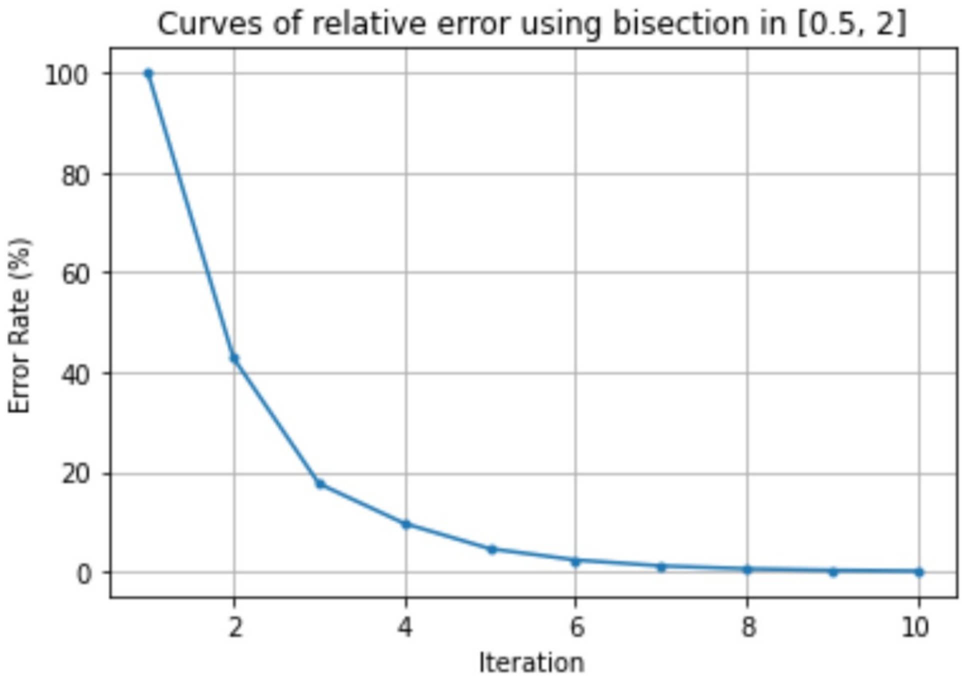

Figure 7: Curves of relative error using bisection in $[0, 0.5]$

False Position Approximation: x=0.9969

Iteration	xl	xu	xr	er(%)
1	0.5000	2.0000	0.6182	100.000
2	0.6182	2.0000	0.7443	16.948
3	0.7443	2.0000	0.8432	11.719
4	0.8432	2.0000	0.9060	6.934
5	0.9060	2.0000	0.9433	3.961
6	0.9433	2.0000	0.9655	2.299
7	0.9655	2.0000	0.9789	1.363
8	0.9789	2.0000	0.9870	0.822
9	0.9870	2.0000	0.9920	0.501
10	0.9920	2.0000	0.9950	0.308
11	0.9950	2.0000	0.9969	0.190

Figure 8: Each iteration result using false position in $[0, 0.5]$

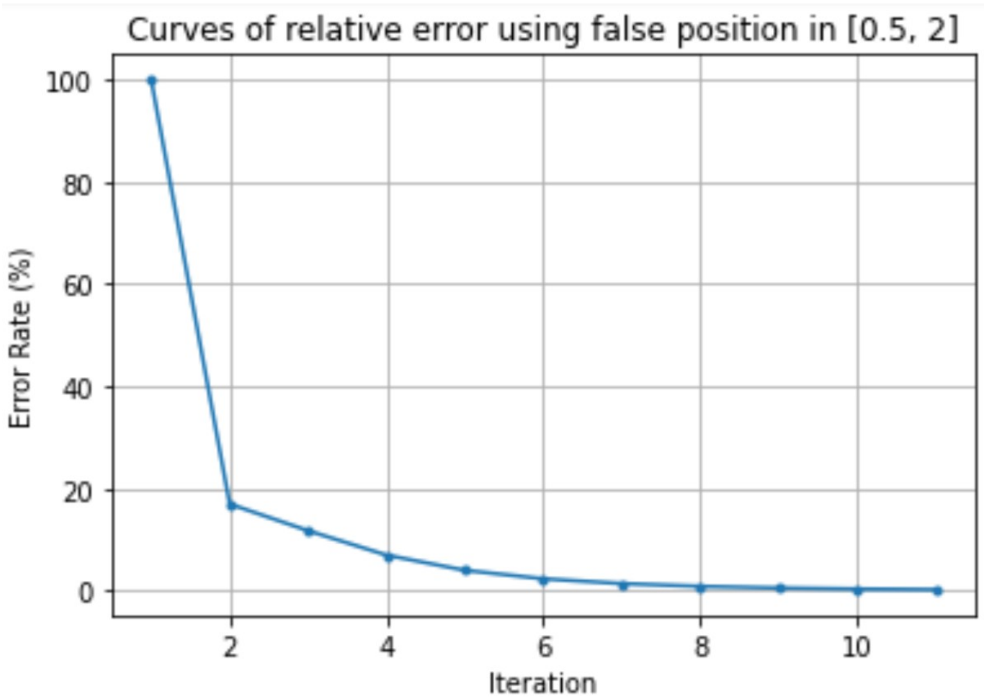

Figure 9: Curves of relative error using false position in $[0, 0.5]$

3. a. for XY plane rotation:

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

for +z-axis to +y-axis(around +x-axis):

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & \cos(-\theta) & -\sin(-\theta) \\ 0 & \sin(-\theta) & \cos(-\theta) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}$$

c. (20%)

a) Roation Matrix

```
✓ [26] def RotationMatrix(vector, theta):
0          return np.array([[0, 0, 0], [0, cos(theta*pi/180), sin(theta*pi/180)], [-sin(theta*pi/180), 0, cos(theta*pi/180)]]).dot(vector)
秒
```

b) Magnitude Modulation

```
✓ [27] def magnitudeModulation(t, vector, T1, T2):
0          return np.array([vector[0]*exp(-t/T2), vector[1]*exp(-t/T2), vector[2]*exp(-t/T1)]) + np.array([[0], [0], [1-exp(-t/T1)]])
秒
```

c) MRI function

$S^+ = f(\theta, t, S^-, T1, T2)$

```
✓ [28] def MRIfunction(theta, t, vector, T1, T2):
0          S = magnitudeModulation(t, RotationMatrix(vector, theta), T1, T2)
秒         return (S[0]**2+S[1]**2)**0.5, np.array([[0], [0], S[2]])
```

e. (10%) Graphical method to estimate θ_{opt}: 31°

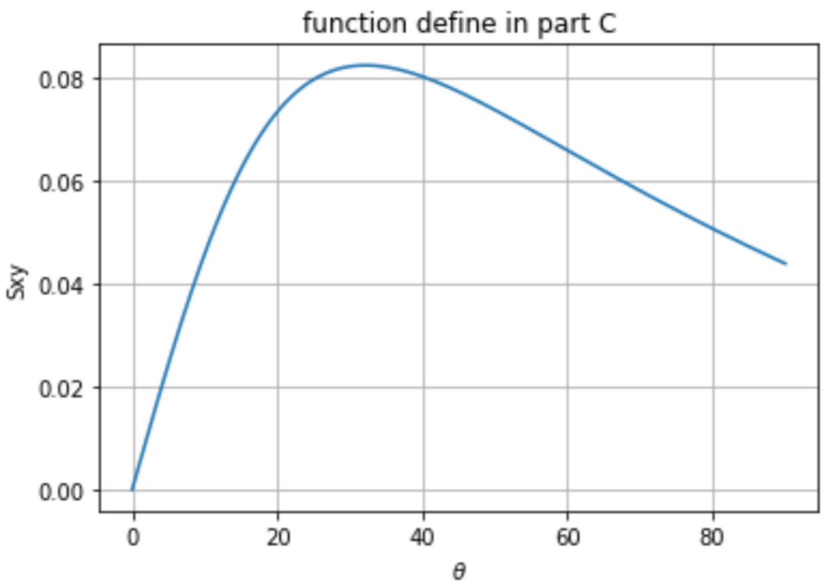

Figure 10: MRI function with 500 iterations in different θ (thick function)

f. (15%) approximate solution of $g(\theta)$ is 32.2 with h = 0.1

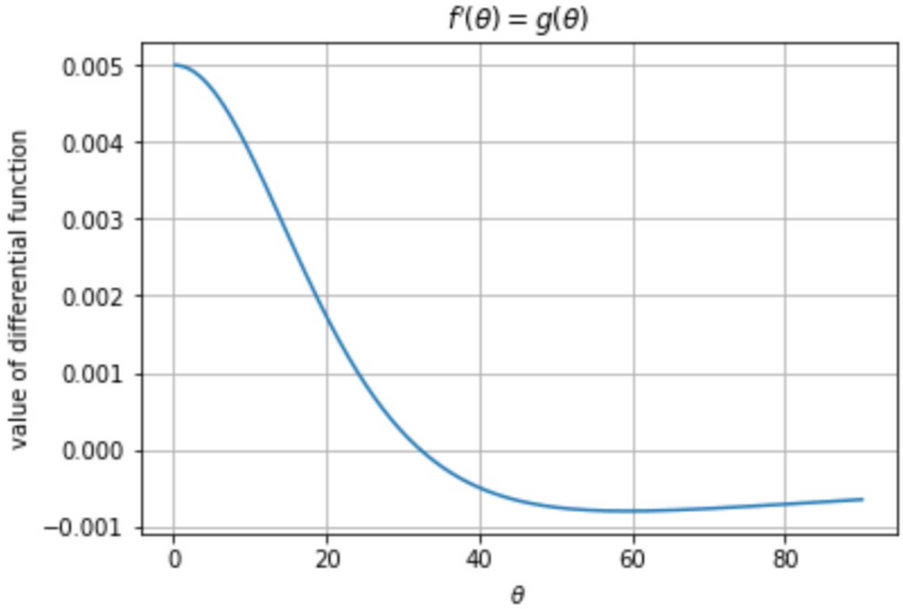

Figure 11: centered finite-difference approach $g(\theta) = f'(\theta)$

g. (15%) Approximate solution of golden-section is 31.1157 (or 32.2884)

Iteration	xl	xu	er
1	0.0000	90.0000	1.000
2	0.0000	55.6231	0.618
3	21.2461	55.6231	0.191
4	21.2461	42.4922	0.236
5	29.3614	42.4922	0.083
6	29.3614	37.4767	0.090
7	29.3614	34.3769	0.059
8	31.2772	34.3769	0.022
9	31.2772	33.1929	0.023
10	31.2772	32.4612	0.014
11	31.7294	32.4612	0.005
12	32.0089	32.4612	0.003
13	32.0089	32.2884	0.003
14	32.1157	32.2884	0.001

Figure 12: Each iteration result using Golden-section

h. (5%) Solution by analytical one is roughly 32.16896, and approximate solution from e, f and g is 31, 32.2 and 32.1157 (or 32.2884). Part e(Graphical method), use smell range of to approach may x. Part f(centered finite-difference approach), use smeller h (0.01, 0.001) to approach. Part g(Golden-section), let ϵ_s lower(e.g. 10^{-4}) to increase # of iterations.