AM205 HW3 Writeup

Jiawen Tong

Problem 1
(a)
See plot in Figure 1.

Figure 1: Problem 1: (a)

By composite trapezoid rule, for each evenly spaced interval $[x_{i-1}, x_i]$, we have

$$\int_{x_{i-1}}^{x_i} f(x)dx = \frac{h}{2}[f(x_{i-1}) + f(x_i)]$$

$$h = |x_i - x_{i-1}|$$

.

Therefore, the integral on the entire interval [a, b], we have

$$\int_a^b f(x)dx = \frac{h}{2}[f(x_0) + f(x_1) + f(x_1) + f(x_2) + ... + f(x_{n-1}) + f(x_n)]$$

$$= h[\frac{1}{2}f(a) + \frac{1}{2}f(b) + f(x_1) + ... + f(x_{n-1})]$$

Plugging in a=0, b=$\frac{\pi}{3}$, n=1,2,3...,50, $h = \frac{(b-a)}{n}$, we can calculate the composite integral using trapezoid rule on each evenly spaced small interval.

As the log-log plot shows, the log-scale absolute error has a linear relation with the log-scale h. This indicates that the absolute error scales like h^m for some fixed m. In addition, the numerically computed log-scale absolute error is always smaller than the error bound from lecture

$$E(h) = \frac{h^2\pi}{36}||f^{''}||_\infty$$

where $f^{''}(x)$ is:

$$f^{''}(x) = \frac{-\frac{5}{4}cosx + sin^2x + 1}{(\frac{5}{4} - cosx)^3}$$

The rightmost point at n=1, h=(b-a) is missing. This is because by coincidence h=(b-a) gives a very accurate numerical result for this integral, giving absolute error as 0.

(b)

See plot in Figure 2.

Using the same procedure as in part(a), I got the log-scale absolute error grows **non-linearly** with regards to log-scale h. This indicates that the absolute error of this integration is **not bounded by** h^m for some m. If we take infinitely small h, the log-scale error has infinitely large slope, which could not be bounded by a certain m.

Problem 2

(a)

Given the cubic **Legendre polynomial** $P_3(x) = \frac{1}{2}(5x^2 - 3)$, its roots over [-1, 1] are calculated as:

$$P_3(x) = \frac{1}{2}(5x^2 - 3) = 0$$

$$x_1 = -\sqrt{\frac{3}{5}}, \quad x_2 = 0, \quad x_3 = \sqrt{\frac{3}{5}}$$

Then, to get the quadrature weights, we use the roots as the fitting points to integrate the **Lagrange polynomial**.

$$L_0(x) = \frac{(x-0)(x-\sqrt{\frac{3}{5}})}{(-\sqrt{\frac{3}{5}}-0)(-\sqrt{\frac{3}{5}}-\sqrt{\frac{3}{5}})} = \frac{5}{6}x(x-\sqrt{\frac{3}{5}})$$

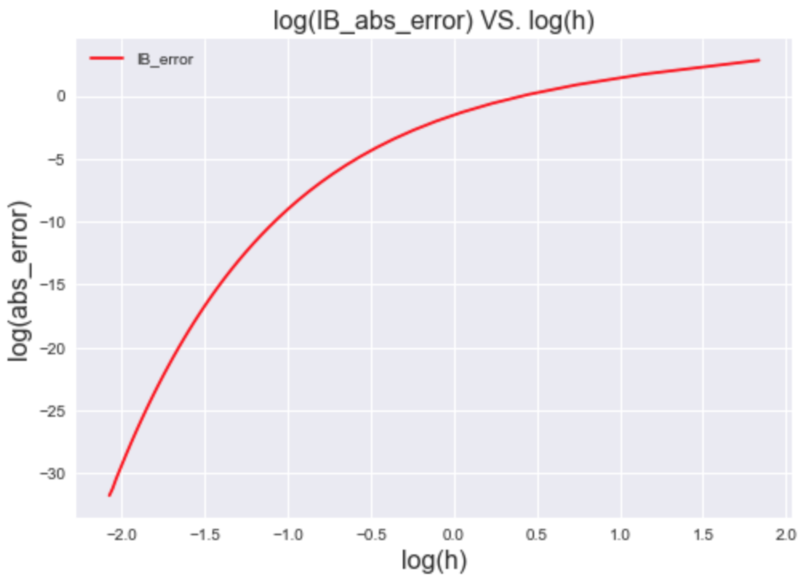

Figure 2: Problem 1: (b)

$$L_1(x) = \frac{(x+\sqrt{\frac{3}{5}})(x-\sqrt{\frac{3}{5}})}{\sqrt{\frac{3}{5}} * (-\sqrt{\frac{3}{5}})} = -\frac{5}{3}x^2 + 1$$

$$L_2(x) = \frac{(x+\sqrt{\frac{3}{5}})(x-0)}{2\sqrt{\frac{3}{5}} * \sqrt{\frac{3}{5}}} = \frac{5}{6}x(x+\sqrt{\frac{3}{5}})$$

$$w_i = \int_{-1}^{1} L_i(x)dx, \quad i = 0, 1, 2$$

$$w_0 = \frac{5}{9}, w_1 = \frac{8}{9}, w_2 = \frac{5}{9}$$

It could be demonstrated that $Q(f) = \Sigma_{k=0}^{2} f(x_k)$ can integrate polynomials up to the **expected degree = 5** exactly. We have 3 parameters for quadrature points and 3 parameters for weights. Thus, 6 parameters can integrate polynomials up to degree 5.

Let an arbitrary polynomial upto degree 5 be denoted as:

$$f(x) = a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$$

$$\int_{-1}^{1} f(x)dx = \int_{-1}^{1} (a_4x^4 + a_2x^2 + a_0)dx = \frac{2}{5}a_4 + \frac{2}{3}a_2 + 2a_0$$

$$Q(f) = \Sigma_{k=0}^{2} w_k f(x_k) = \frac{5}{9}[f(-\sqrt{\frac{3}{5}}) + f(\sqrt{\frac{3}{5}})] + \frac{8}{9}a_0$$

$$= \frac{5}{9} * 2[a_4(\sqrt{\frac{3}{5}})^4 + a_2(\sqrt{\frac{3}{5}})^2 + a_0] + \frac{8}{9}a_0 = \frac{2}{5}a_4 + \frac{2}{3}a_2 + 2a_0$$

Hence, we can exactly integrate any polynomial f upto degree 5.

$$\Sigma_{k=0}^{2} w_k f(x_k) = \int_{-1}^{1} f(x)dx$$

(b)

Using the recursive adoptive routine to sum up integrals over sub-intervals, we can get the value of the **integral I, number of intervals, total error** for functions $f(x) = x^m - x^2 + 1$ (m = 4,5,6,7,8) over $[-1, \frac{5}{4}]$ as reported in Figure 3. Before applying the Adaptive 3-point Gauss Quadrature, we need to transform the integral region to be within [-1, 1]. See detailed implementation in code.

$$z = \frac{b-a}{2}x + \frac{b+a}{2} \qquad \int_{a}^{b} f(x)dx = \frac{b-a}{2}\int_{-1}^{1} f(z)dz$$

$$z_i = \frac{b-a}{2}x_i + \frac{b+a}{2} \qquad \int_{a}^{b} f(x)dx \sim \frac{b-a}{2}\Sigma_{i=1}^{n} w_i f(z_i)$$

Reference: Gauss Quadrature - Change of interval (`https://en.wikipedia.org/wiki/Gaussian_quadrature`)

```
m = 4: I = 2.0759765625, n_intervals = 1, total_error = 0.00000000000000000000
m = 5: I = 1.7347412109, n_intervals = 1, total_error = 0.00000000000000000000
m = 6: I = 2.0896776854, n_intervals = 8, total_error = 0.00000039150764707951
m = 7: I = 1.8856830810, n_intervals = 11, total_error = 0.00000059840882827775
m = 8: I = 2.2045781013, n_intervals = 13, total_error = 0.00000029345507567002
```

Figure 3: Problem 2: (b)

(c)

For $\int_{-1}^{1}|x|dx, \quad \int_{-1}^{2}|x|dx, \int_{0}^{1} x^{3/4} sin\frac{1}{x}dx$, report **integral I, number of intervals, total error** in Figure 4.

Problem 3

(a)

```
I(|x|) over [-1, 1]:
I = 1.0000000000, n_intervals = 2, total_error = 0.00000000000000000000

I(|x|) over [-1, 2]:
I= 2.5000000001, n_intervals = 16, total_error = 0.00000000007205526794

I(x**(3/4)*sin(1/x)) over [0, 1]:
I = 0.4070268679, n_intervals = 194326, total_error = 0.00000021294988755797
```

Figure 4: Problem 2: (c)

Plot $g(x; \frac{1}{3})$ in Figure 5.

(b)
See plot of **integral I, number of intervals** for

$$I(\phi) = \int_{-1/2}^{1/2} g(x; \phi) dx, \quad \phi \in (0, 1)$$

in Figure 6. Approximately $\phi \in$ **range (0.4, 0.5)** requires the most intervals.

Problem 4
(a)

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f(t, y(t)) dt$$

$$y^{'}(t) = f(t, y(t))$$

$$y^{'}(t_k + \frac{h}{2}) = f(t_k + \frac{h}{2}, y(t_k + \frac{h}{2})) \sim f(t_k + \frac{h}{2}, \frac{y_k + y_{k+1}}{2})$$

Update each y_k using mid-point method:

$$y_{k+1} = y_k + hf(t_k + \frac{h}{2}, \frac{y_k + y_{k+1}}{2}) = y_k + hy^{'}(t_k + \frac{h}{2})$$

The truncation error at step k is:

$$\begin{aligned} T_k &= \frac{y(t_{k+1}) - y(t_k)}{h} - f(t_k + \frac{h}{2}, \frac{y_k + y_{k+1}}{2}) \\ &= \frac{y(t_{k+1}) - y(t_k)}{h} - y^{'}(t_k + \frac{h}{2}) \end{aligned}$$

Let $t_m = t_k + \frac{h}{2}$ denote the mid-point, and $L = \frac{h}{2}$, we have

$$t_{k+1} = t_m + L, \quad t_k = t_m - L$$

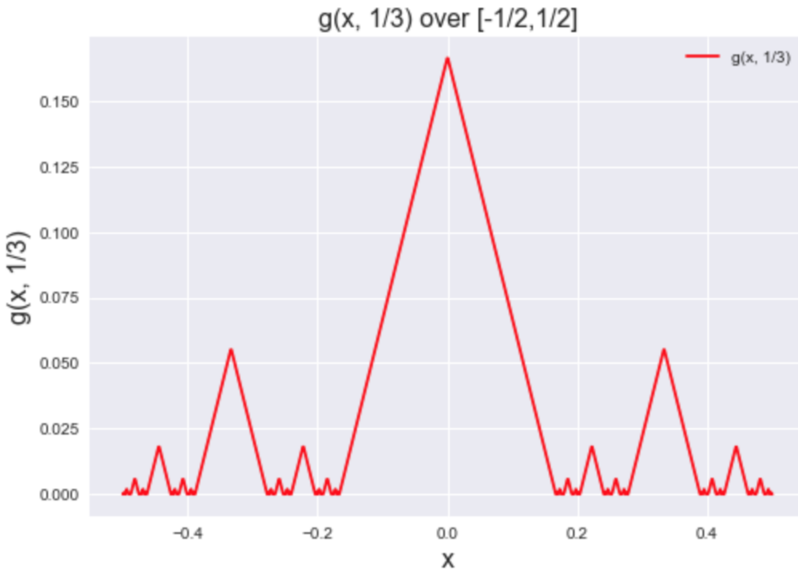

Figure 5: Problem 3: (a)

$$T_k = \frac{y(t_m + L) - y(t_m - L)}{2L} - y^{'}(t_m)$$

$$y(t_m + L) = y(t_m) + Ly^{'}(t_m) + \frac{L^2}{2}y^{''}(t_m) + \frac{L^3}{6}y^{'''}(\theta_1)$$

$$y(t_m - L) = y(t_m) - Ly^{'}(t_m) + \frac{L^2}{2}y^{''}(t_m) - \frac{L^3}{6}y^{'''}(\theta_2)$$

$$T_k = \frac{L^2}{12}[y^{'''}(\theta_1) + y^{'''}(\theta_2)] = \frac{h^2}{48}[y^{'''}(\theta_1) + y^{'''}(\theta_2)]$$

$$\theta_1, \theta_2 \in [t_k, t_{k+1}]$$

Hence, we prove that the accuracy of this method is 2.

(b)

$$y_{k+1} = y_k + h\lambda y^{'} = y_k + h\lambda y_{t_k + \frac{h}{2}} \sim y_k + h\lambda \frac{y_k + y_{k+1}}{2}$$

$$(2 - h\lambda)y_{k+1} = (2 + h\lambda)y_k$$

$$y_{k+1} = \frac{2 + h\lambda}{2 - h\lambda}y_k$$

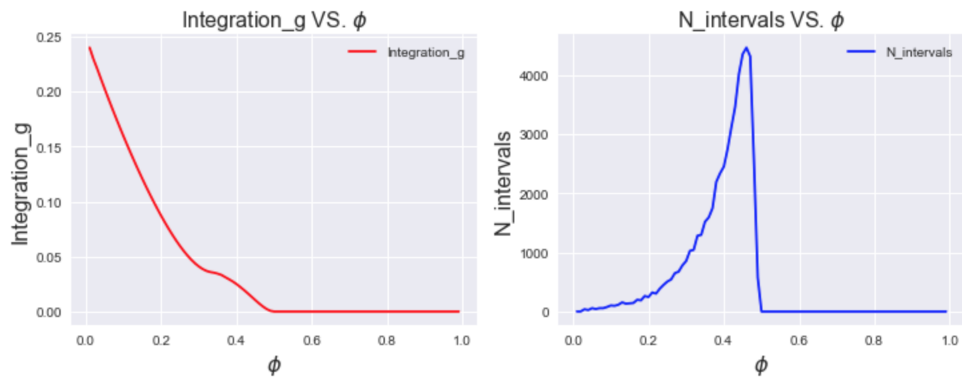

Figure 6: Problem 3: (b)

Thus, we get the amplification factor as $(\frac{2+h\lambda}{2-h\lambda})$. The stability region is given by restricting the absolute amplification factor to be no more than 1.

Let $\bar{h} = h\lambda = a + bi$,

$$\frac{|2+a+bi|}{|2-a-bi|} \leq 1$$

$$|2+a+bi| \leq |2-a-bi|$$

$$(2+a+bi)(2+a-bi) \leq (2-a-bi)(2-a+bi)$$

$$a \leq 0$$

Hence, the stability region of this method is $a = Re(\bar{h}) \leq 0$.

Problem 6

(a)

Expanding the partial derivatives of the Jacobi integral gives the system of ODEs as follows:

$$x^{'} = -\frac{1}{2}[-2u] = u$$

$$y^{'} = -\frac{1}{2}[-2v] = v$$

$$u^{'} = v + x - \mu + \frac{(\mu-1)x}{(x^2+y^2)^{\frac{3}{2}}} + \frac{\mu(1-x)}{((x-1)^2+y^2)^{\frac{3}{2}}}$$

$$v^{'} = -u + y + \frac{(\mu-1)y}{(x^2+y^2)^{\frac{3}{2}}} - \frac{\mu y}{((x-1)^2+y^2)^{\frac{3}{2}}}$$

(b)

To shorten the running time for this routine, I discussed the idea for this question with Xiaohan Wu.

To check if a line segment through (x_0, y_0), (x_1, y_1), I calculate the intersection points of the entire line with the unit circle centered at (0, 0), with radius = 1. If at least one intersection point exists and it lies on the segment, then we say the segment intersects with the unit circle.

Thus, to detect if a line segment goes through the Earth (0, 0) $r_{earth} = 0.02$ or the Moon (1, 0) $r_{moon} = 0.005$, we apply this routine by parsing radius as an additional argument. With moon, we also need to right shift the x-coordinate by 1. See detailed implementation in codes

(c)

Adding random noise to observed initial states, we get

$$x(0), y(0) = x_{obs}(0) + E_x, y_{obs}(0) + E_y$$

$$x(0.02), y(0.02) = x_{obs}(0.02) + E_x, y_{obs}(0.02) + E_y$$

$$u(0), v(0) = \frac{x(0.02) - x(0)}{0.02}, \frac{y(0.02) - y(0)}{0.02}$$

Running 10 times to get 10 initial states and using python's odeint routine to solve the ODEs will give 10 trajectories in Figure 7.

(d)

Run 2500 times and count the number of collisions with the Earth, the Moon, or both. Results are summarized in Figure 8.

Plotted in Figure 9 are the first 10 trajectories colliding with the Earth, all collisions with the Moon, and all collisions with both.

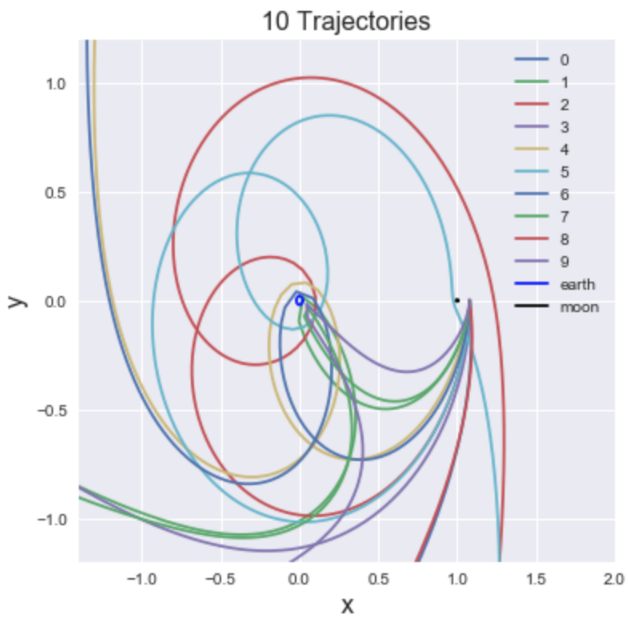

Figure 7: Problem 6: (c)

```
The number of collision with
- Earth: 436
- Moon: 8
- Both: 2
The probability of colliding with
- Earth: 0.1744000
- Moon: 0.0032000
- Both: 0.0008000
```

Figure 8: Problem 6: (d) Collision Probabilities

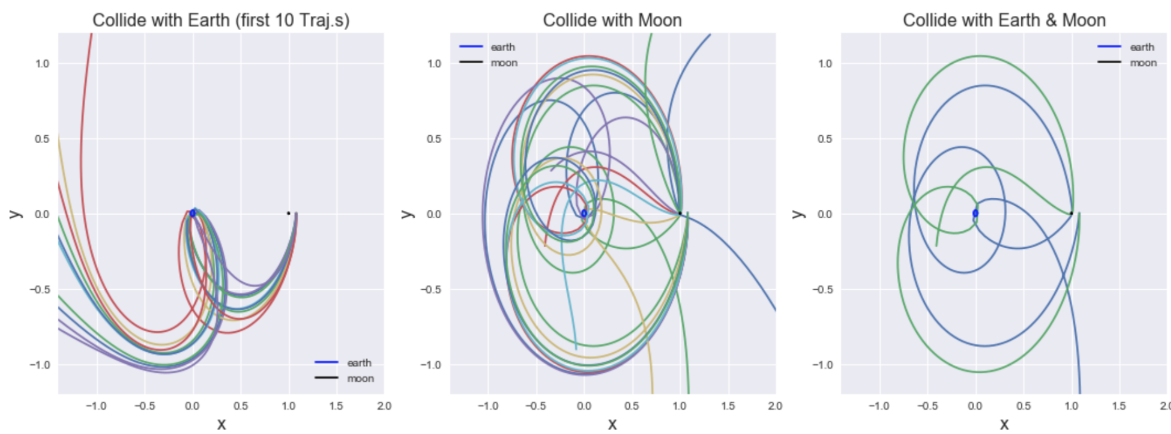

Figure 9: Problem 6: (d) Collision Trajectories