HW 6

AMATH 585 Tianbo Zhang

1. Prove that the ODE IVP

$$y'(t) = \frac{1}{t^2 + 2y(t)^2}, \quad t \geq 1,$$

$$y(1) = \eta,$$

has a unique solution for all $t \geq 1$. Find a Lipschitz constant for this problem.

① Prove for continuity:

Let $f(t,y) = (t^2+2y^2)^{-1}$

Then note $f(t,y)$ is discontinuous for $t^2+2y^2=0$

Note since we have $t \geq 1$ and also:

$2y^2 \geq 0$ (By definition)

$\Rightarrow f(t,y)$ is continuous for $t \geq 1$, $-\infty < y < \infty$

② Lipschitz condition in y:

Let $y_1, y_2 \in y$

Then note we have:

$$|f(y_1)-f(y_2)| = |(t^2+2y_1^2)^{-1} - (t^2+2y_2^2)^{-1}|$$

$$= \left| \frac{2y_2^2 - 2y_1^2}{(t^2+2y_1^2)(t^2+2y_2^2)} \right|$$

$$= \left| \frac{2(y_2^2-y_1^2)}{t^4+2t^2y_1^2+2t^2y_2^2+4y_1^2y_2^2} \right| = \left| \frac{2(y_2+y_1)}{(t^2+2y_1^2)(t^2+2y_2^2)} \right| |y_2-y_1|$$

Then let $r \geq 2$, then we have:

$$|f(y_1)-f(y_2)| = \left| \frac{2(y_2+y_1)}{(t^2+2y_1^2)(t^2+2y_2^2)} \right| |y_2-y_1| \leq \left| \frac{r(y_2+y_1)}{(t^2+2y_1^2)(t^2+2y_2^2)} \right| |y_2-y_1|$$

Hence by Theorem 11.1.3 we have:

The IVP has an unique solution for $t \geq 0$

with a Lipschitz constant of:

$$L = \left| \frac{r(y_2+y_1)}{(t^2+2y_1^2)(t^2+2y_2^2)} \right|, \quad r \geq 2$$

2. Consider the equation of harmonic motion:

$$u'' = -ku, \quad u(t_0) = u_0, \quad u'(t_0) = v_0.$$

Here $u(t)$ represents the distance from equilibrium and $k > 0$ is a spring constant. Write this as a system of two first-order differential equations, and show that the right-hand side of your system satisfies a Lipschitz condition on $\mathbf{R}^2$. Determine the (smallest possible) Lipschitz constant for the 1-norm, the 2-norm, and the ∞-norm.

① Write as a system of two 1st order differential equations:

Let $v = u'$, then we have:

$v' = u'' = -ku$

$\Rightarrow \begin{cases} v' = -ku \\ u' = v \end{cases}$

② To show $f(u,v)$ satisfies a Lipschitz condition on $\mathbb{R}^2$ and find smallest L

Let $f(u,v) = (v, -ku)$

Then let $u_1, u_2 \in U$, $v_1, v_2 \in V$

(1) 1-norm

$\|f(u_2, v_2) - f(u_1, v_1)\|_1 = \|(v_2 - v_1, -k(u_2 - u_1))\|$

$= |v_2 - v_1| + |k||u_2 - u_1|$

$\|(u_2, v_2) - (u_1, v_1)\|_1 = |v_2 - v_1| + |u_2 - u_1|$

$\boxed{\begin{array}{l} \text{Let } L = \max(1, |k|) \\ \text{Then } \|f(u_2, v_2) - f(u_1, v_1)\|_1 \leq L\|(u_2, v_2) - (u_1, v_1)\|_1 \end{array}}$

(2) 2-norm

$\|f(u_2, v_2) - f(u_1, v_1)\|_2 = \left[(v_2 - v_1)^2 + (-k(u_2 - u_1))^2\right]^{\frac{1}{2}}$

$= \left[(v_2 - v_1)^2 + k^2(u_2 - u_1)^2\right]^{\frac{1}{2}}$

$\|(u_2, v_2) - (u_1, v_1)\|_2 = \left[(u_2 - u_1)^2 + (v_2 - v_1)^2\right]$

Then note:

k^2 is by definition greater than equal to 0.

$\Rightarrow (1+k^2)(v_2 - v_1)^2 + (1+k^2)(u_2 - u_1)^2 \geq (v_2 - v_1)^2 + k^2(u_2 - u_1)^2$

$\boxed{\begin{array}{l} \text{Thus let } L = \sqrt{1+k^2} \\ \Rightarrow \text{Then } \|f(u_2, v_2) - f(u_1, v_1)\|_2 \leq L\|(u_2, v_2) - (u_1, v_1)\|_2 \end{array}}$

(3) ∞-norm

$\|f(u_2, v_2) - f(u_1, v_1)\|_\infty = \max(|v_2 - v_1|, |k||u_2 - u_1|)$

$\|(u_2, v_2) - (u_1, v_1)\|_\infty = \max(|u_2 - u_1|, |v_2 - v_1|)$

$\boxed{\begin{array}{l} \text{Then let } L = \max(1, |k|) \\ \Rightarrow L \geq |k|, \ L \geq 1 \\ \Rightarrow \|f(u_2, v_2) - f(u_1, v_1)\|_\infty \leq L\|(u_2, v_2) - (u_1, v_1)\|_\infty \end{array}}$

3. Consider the one-step method

$$y_{n+1} = y_n + h[\theta f(t_n, y_n) + (1-\theta) f(t_{n+1}, y_{n+1})],$$

where $\theta \in [0,1]$ is given. Note that this method is *explicit* if $\theta = 1$ and otherwise it is implicit. Show that the local truncation error is $O(h^2)$ if $\theta = 1/2$ and otherwise it is $O(h)$.

First note we have:

$$y_{n+1} - y_n - h[\theta f(t_n, y_n) + (1-\theta) f(t_{n+1}, y_{n+1})] = 0$$

Then note if we substitute the true solution $y(t_n)$ we have:

$$y(t_{n+1}) - y(t_n) - h[\theta f(y(t_n)) + (1-\theta) f(y(t_{n+1}))] = 0$$

Then note by Taylor's Theorem we have:

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + \frac{h^2}{2} y''(t_n) + \frac{h^3}{6} y'''(t_n) + O(h^4)$$

Also note that we have:

$$f(t_{n+1}, y_{n+1}) = y'(t_n) + hy''(t_n) + \frac{h^2}{2} y'''(t_n) + O(h^3)$$

$$f(t_n, y_n) = y'(t_n)$$

Then plug in these expressions we have:

$$\cancel{y(t_n)} + \cancel{hy'(t_n)} + \frac{h^2}{\cancel{2}} \cancel{y''(t_n)} + \frac{h^3}{6} y'''(t_n) + O(h^4) - \cancel{y(t_n)}$$
$$-h\left[\theta \cancel{y'(t_n)} + (1-\theta)[\cancel{y'(t_n)} + h\cancel{y''(t_n)} + \frac{h^2}{2} y'''(t_n)] + O(h^3)\right]$$

$$= h(1-\theta-1+\theta) y'(t_n) + h^2(\tfrac{1}{2} - 1 + \theta) y''(t_n) + (\tfrac{\theta}{2} - \tfrac{1}{3}) h^3 y'''(t_n) + O(h^4)$$

$$= (\theta - \tfrac{1}{2}) h^2 y''(t_n) + (\tfrac{\theta}{2} - \tfrac{1}{3}) h^3 y'''(t_n) + O(h^4)$$

Then note from lecture we have that LTE is $1/h$ times the first neglected term.

$$\Rightarrow \text{LTE} = (\theta - \tfrac{1}{2}) h y''(t_n) + (\tfrac{\theta}{2} - \tfrac{1}{3}) h^2 y'''(t_n) + O(h^3)$$

$$\boxed{\begin{array}{l} \text{Then note:} \\ \text{(1) If } \theta = \tfrac{1}{2} \text{ then we have:} \\ \quad \text{the local truncation error: } O(h^2) \\ \text{(2) If } \theta \neq \tfrac{1}{2} \text{ then we have:} \\ \quad \text{the local truncation error: } O(h) \end{array}}$$

4. Compute the leading term in the local truncation error of Heun's method:

$$\begin{aligned}\tilde{y}_{n+1} &= y_n + hf(t_n, y_n),\\ y_{n+1} &= y_n + \frac{h}{2}\left[f(t_{n+1}, \tilde{y}_{n+1}) + f(t_n, y_n)\right].\end{aligned}$$

First note we have:

$$y_{n+1} - y_n - \frac{h}{2}\left[f(t_{n+1}, \tilde{y}_{n+1}) + f(t_n, y_n)\right] = 0$$

Then note after substituting the true solution we have:

$$LTE: y(t_{n+1}) - y(t_n) - \frac{h}{2}\left[f(t_{n+1}, \tilde{y}(t_{n+1})) + f(t_n, y(t_n))\right]$$

Then note by Tylor's Theorem we have:

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n) + O(h^3)$$

Then note we have:

$$f(t_n, y(t_n)) = y'(t_n)$$

Then we have:

$$\tilde{y}(t_{n+1}) = y(t_n) + hy'(t_n)$$

$$\Rightarrow f(t_{n+1}, \tilde{y}(t_{n+1})) = y'(t_n) + hy''(t_n) + O(h^3)$$

$$\Rightarrow LTE = \left[\cancel{y(t_n)} + h\cancel{y'(t_n)} + \frac{h^2}{\cancel{2}}\cancel{y''}(t_n) + O(h^3) - \cancel{y(t_n)} - \frac{h}{2}\left[\cancel{y'(t_n)} + h\cancel{y''}(t_n) + O(h^3) + \cancel{y'(t_n)}\right]\right] \cdot \frac{1}{h}$$

$$= \frac{h^2}{3!}y'''(t_n) - \frac{1}{2} \cdot \frac{h^3}{3!}y'''(t_n) + O(h^4)$$

$$\Rightarrow \boxed{\text{The leading term is: } \frac{h^2}{3!}y'''(t_n)}$$

5. The initial value problem

$$y'(t) = y(t)^2 - \sin(t) - \cos^2(t), \quad y(0) = 1$$

has the solution $y(t) = \cos(t)$. Write a computer code to solve this problem up to time $T = 8$ with various different time steps $h = T/N$, with

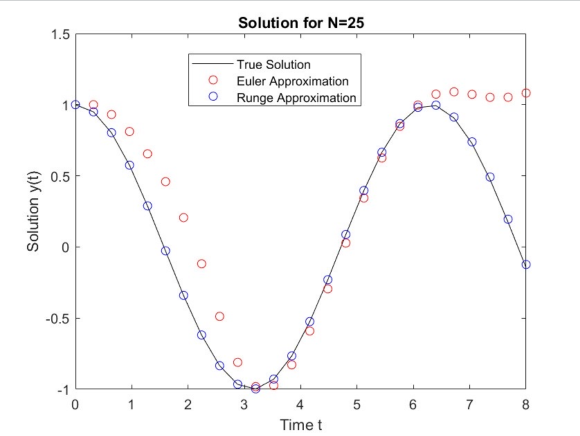
$$N = 25,\ 50,\ 100,\ 200,\ 400,\ 800,\ 1600.$$

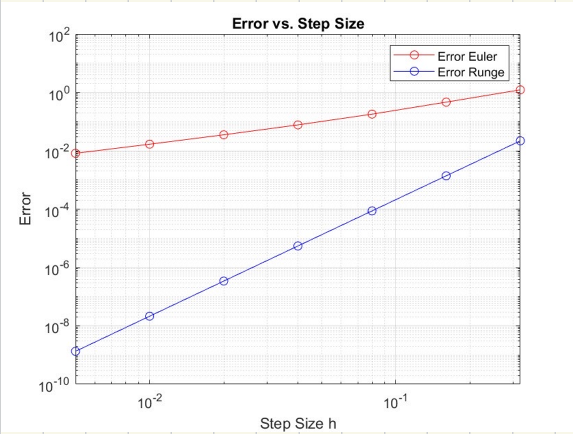
Do this using two different methods:

(a) Forward Euler.

(b) The classical fourth-order Runge-Kutta method.

Plot the true solution (with a solid line) and the approximate solution (with o's) for $N = 25$. Make a log-log plot of the errors for each method vs. the stepsize h.





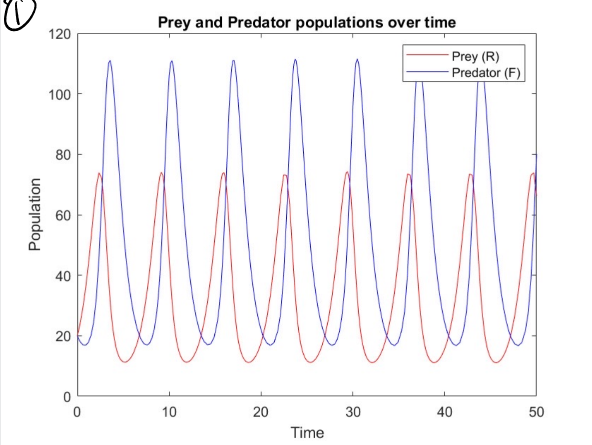
```matlab
% AMATH 585 HW6 Problem 5
% Tianbo Zhang 1938501
% Solve the IVP problem with Forward Euler, Classiscal Fourth-order
% Runge-kutta method

% Set up base conditions
T = 8;
y0 = 1;
N_values = [25, 50, 100, 200, 400, 800, 1600];
error_euler = zeros(1, length(N_values));
error_runge = zeros(1, length(N_values));

for i = 1:length(N_values)
    N = N_values(i);
    h = T/N;
    t = 0:h:T;

    %Forward Euler Method
    y_euler = zeros(1, N+1);
    y_euler(1) = y0;
    for j = 1:N
        y_euler(j+1) = y_euler(j) + h * dy_dt(t(j), y_euler(j));
    end

    % Classical Fourth-order Runge-kutta method
    y_runge = zeros(1, N+1);
    y_runge(1) = y0;
    for j = 1:N
        q1 = dy_dt(t(j), y_runge(j));
        q2 = dy_dt(t(j)+h/2, y_runge(j) + h*q1/2);
        q3 = dy_dt(t(j)+h/2, y_runge(j) + h*q2/2);
        q4 = dy_dt(t(j)+h, y_runge(j) + h*q3);
        y_runge(j+1) = y_runge(j) + (h/6)*(q1 + 2 * q2 + 2 * q3 + q4);
    end

    % Compute Errors
    y_true = y(t);
    error_euler(i) = max(abs(y_true - y_euler));
    error_runge(i) = max(abs(y_true - y_runge));

    % Plot for N+25
    if N == 25
        figure;
        plot(t, y_true, 'k-');
        hold on
        plot(t, y_euler, 'ro');
        hold on
        plot(t, y_runge, 'bo');
        hold off
        legend('True Solution', 'Euler Approximation', 'Runge Approximation');
        title('Solution for N=25');
        xlabel('Time t');
        ylabel('Solution y(t)');
    end
end

% Plot errors
h_values = T./N_values;
figure;
loglog(h_values, error_euler, 'r-o', h_values, error_runge, 'b-o');
legend('Error Euler', 'Error Runge');
title('Error vs. Step Size');
xlabel('Step Size h');
ylabel('Error');
grid on;

% Calculate y'
function dy = dy_dt(t, y)
    dy = y^2 - sin(t) - cos(t)^2;
end

% Calculate true solution
function y_true = y(t)
    y_true = cos(t);
end
```

6. Use a method of your choice or try `ode45` in Matlab to solve the Lotka-Volterra predator-prey equations:

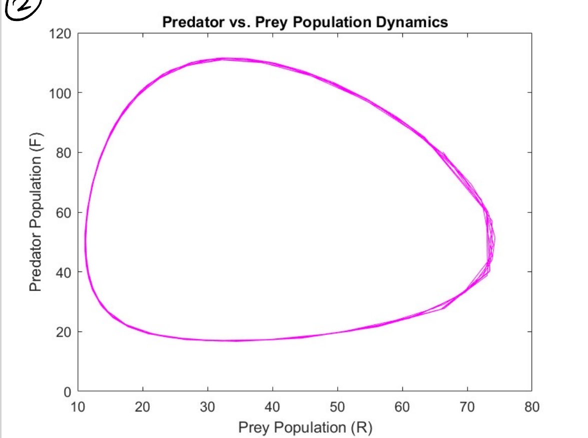
$$\begin{aligned} R' &= (1 - .02F)R, \\ F' &= (-1 + .03R)F, \end{aligned}$$

starting with $R_0 = F_0 = 20$. Plot $R(t)$ and $F(t)$ vs. t on the same plot, labeling each curve. Present the solution in another way with a plot of F vs. R.

①



②



First note from figure ① we can see that:

(1) The graph shows a periodic system which makes sense since: As the population of prey decreases which means that there're less food for the predator which leads to the decrease for the predator's population. (Reverse otherwise)

(2) Then note the trajectory of the predators is shift a bit to the right. One possibility is that the predators needs time to realize the prey population has increase or decrease.

Then from figure 2 we can see that:

① The closed loops indicate the system is indeed periodic.

② Then note it also supports the fact that increase in the prey population leads to a growth to the predator populaion.

③ Then growth in predator populaton also leads to a decay in prey population.

```matlab
% AMATH 585 HW6 Problem 6
% Tianbo Zhang 1938501
% Use ode45 to solve the Lotka-Volterra predator-prey equations

% Set up initial equation
R0 = 20;
F0 = 20;
y0 = [R0; F0];

% Define time span
t_span = [0, 50]

[T, Y] = ode45(@lotka_volterra, t_span, y0);

% Plot R(t) and F(t) vs. t
figure;
plot(T, Y(:,1), 'r-', T, Y(:,2), 'b-');
xlabel('Time');
ylabel('Population');
legend('Prey (R)', 'Predator (F)');
title('Prey and Predator populations over time');

% Plot F vs. R
figure;
plot(Y(:,1), Y(:,2), 'm-');
xlabel('Prey Population (R)');
ylabel('Predator Population (F)');title('Predator vs. Prey Population Dynamics');

function dydt = lotka_volterra(t, y)
    dydt =[(1 - 0.02 * y(2)) * y(1); (-1 + 0.03 * y(1)) * y(2)];
end
```