Assignment 5

學號：111550129　姓名：林彥亨

---

1. (20%) Solve $y' = \sin(x) + y$, $y(0) = 2$ by the modified Euler method to get $y(0.1)$ and $y(0.5)$. Use a value of h small enough to be sure that you have five digits correct.

Through the Euler method, we get the answer is y(0.1) = 2.21550 and y(0.5) = 3.44326.

- Here is the matlab code.

```matlab
% Define the differential equation
f = @(x, y) sin(x) + y;

% Initial conditions
x0 = 0;
y0 = 2;

% Step size
h = 0.01;

% Number of steps to reach x = 0.5
n_steps = 0.5 / h;

% Initialize arrays to store the values
x = zeros(1, n_steps + 1);
y = zeros(1, n_steps + 1);

% Set initial values
x(1) = x0;
y(1) = y0;

% Apply the modified Euler method
for i = 1:n_steps
    x(i + 1) = x(i) + h;
    k1 = f(x(i), y(i));
    y_predict = y(i) + h * k1;
    k2 = f(x(i + 1), y_predict);
    y(i + 1) = y(i) + (h / 2) * (k1 + k2);
end

% Display the results
fprintf('y(0.1) = %.5f\n', y(floor(0.1/h) + 1));
fprintf('y(0.5) = %.5f\n', y(end));
```

- Here is the output.

```
>> Q1
y(0.1) = 2.21550
y(0.5) = 3.44326
```

2. (20%) Derive the formula for the second-order Adams method. Use the method of undetermined coefficients.

$$y_{n+1} = y_n + h\,(a f_n + b f_{n-1})$$

$$P(x) = f_n + \frac{f_n - f_{n-1}}{h}(x - x_n)$$

$$\int_{x_n}^{x_{n+1}} P(x)\,dx = \int_{x_n}^{x_{n+1}} \left(f_n + \frac{f_n - f_{n-1}}{h}(x - x_n)\right)dx$$

$$= h\left(f_n + \frac{f_n - f_{n-1}}{h}\cdot\frac{h}{2}\right)$$

$$= h\left(f_n + \frac{f_n - f_{n-1}}{2}\right)$$

$$y_{n+1} = y_n + h\,(a f_n + b f_{n-1}) \Rightarrow y_{n+1} - y_n = h\left(\frac{3}{2} f_n - \frac{1}{2} f_{n-1}\right) \#$$

$$a = 1 + \frac{1}{2} = \frac{3}{2}$$

$$b = -\frac{1}{2}$$

3. (30%) For the third-order equation

$$y''' + ty' - 2y = t, \; y(0) = y''(0) = 0, \; y'(0) = 1$$

(a) Solve for $y(0.2)$, $y(0.4)$, $y(0.6)$ by RKF.

(b) Advance the solution to $t = 1.0$ with the Adams-Moulton method.

Part(a)

Using Runge-Kutta method, firstly we defined :

$y_1 = y$

$y_2 = y'$

$y_3 = y''$

Then the system becomes:

$y_1' = y_2$

$y_2' = y_3$

$y_3' = t - t y_2 + 2y_1$

then we use matlab to solve the problem

- Here is the code

```matlab
function dydt = odesystem(t, y)
    dydt = zeros(3,1);
    dydt(1) = y(2);
    dydt(2) = y(3);
    dydt(3) = t - t * y(2) + 2 * y(1);
end

% Initial conditions
t0 = 0;
y0 = [0; 1; 0]; % y(0) = 0, y'(0) = 1, y''(0) = 0

% Time points where solution is sought
tspan = [t0 0.6];

% Solve using Runge-Kutta-Fehlberg method
[t, y] = ode45(@odesystem, tspan, y0);

% Interpolating the results to get y(0.2), y(0.4), and y(0.6)
y_at_02 = interp1(t, y(:,1), 0.2);
y_at_04 = interp1(t, y(:,1), 0.4);
y_at_06 = interp1(t, y(:,1), 0.6);

% Display the results
fprintf('y(0.2) = %.5f\n', y_at_02);
fprintf('y(0.4) = %.5f\n', y_at_04);
fprintf('y(0.6) = %.5f\n', y_at_06);
```

- Here is the result.

```
>> Q3_1
y(0.2) = 0.20013
y(0.4) = 0.40214
y(0.6) = 0.61078
```

Part(2)

In part(b), we using

- Here is the matlab code.

```
function dydt = odesystem(t, y)
    dydt = zeros(3,1);
    dydt(1) = y(2);
    dydt(2) = y(3);
    dydt(3) = t - t * y(2) + 2 * y(1);
end

% Initial conditions
t0 = 0;
y0 = [0; 1; 0]; % y(0) = 0, y'(0) = 1, y''(0) = 0

% Time span and step size
t_final = 1.0;
h = 0.1;
tspan = t0:h:t_final;

% Initial values
y = zeros(length(tspan), 3);
y(1, :) = y0';

% First few steps using RKF to initialize Adams-Moulton
[t_rkf, y_rkf] = ode45(@odesystem, [t0, t0+2*h], y0);

% Copy initial values
y(2,:) = y_rkf(2,:);
y(3,:) = y_rkf(3,:);

% Adams-Moulton Method (Second-order implicit)
for i = 3:length(tspan)-1
    t_prev = tspan(i);
    y_prev = y(i, :)';
    t_next = tspan(i+1);

    % Predictor (using Adams-Bashforth)
    y_predict = y(i, :)' + h * (23/12 * odesystem(t_prev, y_prev) - 16/12 * odesystem(tspan(i-1), y(i-1,:)') + 5/12 * odesystem(tspan(i-2), y(i-2,:)'));

    % Corrector (using Adams-Moulton)
    y(i+1, :) = y(i, :) + h * (5/12 * odesystem(t_next, y_predict) + 8/12 * odesystem(t_prev, y_prev) - 1/12 * odesystem(tspan(i-1), y(i-1,:)'))';
end

% Display the results at specific points
fprintf('y(0.2) = %.5f\n', y(tspan == 0.2, 1));
fprintf('y(0.4) = %.5f\n', y(tspan == 0.4, 1));
fprintf('y(0.6) = %.5f\n', y(tspan == 0.6, 1));
fprintf('y(1.0) = %.5f\n', y(tspan == 1.0, 1));
```

- Here is the result

```
>> Q3_2
y(0.2) = 0.00010
y(0.4) = 0.20017
y(0.6) = 0.40212
y(1.0) = 0.83353
```

4. (30%) Solve through finite differences with four subintervals:

$$\frac{d^2y}{dx^2} + y = 0,\ \ y'(0) + y(0) = 2,$$

$$y'(\frac{\pi}{2}) + y(\frac{\pi}{2}) = -1$$

Firstly, we defined the boundry is from 0 to pi/2 and have four subintervals.

$x_0 = 0,$

$x_1 = pi/8,$

$x_2 = pi/4,$

$x_3 = 3pi/8,$

$x_4 = pi/2$

The boundry condition:

At $x_0 = 0$

y1+(1+h)y0=2h.

At $x_4 = 2\pi/x_4$

y4(1+h)−y3=−h.

- Here is the code.

```
% Number of intervals
n = 4;

% Step size
h = pi / (2 * n);

% Coefficient matrix
A = zeros(n+1, n+1);
b = zeros(n+1, 1);

% Boundary conditions
A(1,1) = 1 + h;
A(1,2) = 1;
b(1) = 2 * h;

A(n+1,n) = -1;
A(n+1,n+1) = 1 + h;
b(n+1) = -h;

% Interior points using finite differences
for i = 2:n
    A(i, i-1) = 1;
    A(i, i) = -(2 + h^2);
    A(i, i+1) = 1;
end

% Solve the system of equations
y = A \ b;

% Display the results
x = linspace(0, pi/2, n+1);
disp('x values:');
disp(x');
disp('y values:');
disp(y);
```

- Here is the result.

```
>> Q4
x values:
         0
    0.3927
    0.7854
    1.1781
    1.5708


y values:
    0.4230
    0.1963
    0.0000
   -0.1963
   -0.4229
```