```
% AE740 HW1 akshatdy
clc;
close all;

% 1.a equations in state variable form
```

$$\dot{x_1} = x_2$$

$$\dot{x_2} = -\frac{g}{l} sin(x_1 + \frac{u}{ml^2})$$

$$y = x_1$$

```
% 1.b (see the function below after all the script code)

% 1.c equilibrium at u = 20
ueq = 20;
xeq = findPendEq(ueq);
fprintf('1.c equilibrium at u = 20\n')
fprintf('xeq = [%f; %f]\n', xeq(1), xeq(2));

% 1.d symbolically linearized model at u = 20 and x = xeq
[A, B] = symLin(xeq, ueq);
fprintf('\n1.d symbolically linearized model at u = 20 and x = [%f; %f]\n',
xeq(1), xeq(2));
fprintf('A = \n[%f, %f;\n%f, %f]\n', A(1, 1), A(1, 2), A(2, 1), A(2, 2));
fprintf('B = \n[%f;\n%f]\n', B(1), B(2));

% 1.e numerically linearized model at u = 20 and x = xeq
[A, B] = cdLin(xeq, ueq);
fprintf('\n1.e numerically linearized model at u = 20 and x = [%f; %f]\n',
xeq(1), xeq(2));
fprintf('A = \n[%f, %f;\n%f, %f]\n', A(1, 1), A(1, 2), A(2, 1), A(2, 2));
fprintf('B = \n[%f;\n%f]\n', B(1), B(2));

% 1.f simulate over 10 seconds
odefun = @(t, x) pendModel(x, ueq + uIn(t));
tspan = [0:0.01:10];
x0 = [0; 0];
[Tode, Xode] = ode45(odefun, tspan, x0);
figure(1);
plot(Tode, Xode(:, 1));
title('1.f pendulum angle over time using ode45');
xlabel('time (s)');
ylabel('angle (rad)');

% 1.g simulate over 10 seconds using forward euler
figure(2);
title('1.g pendulum angle over time using forward euler');
xlabel('time (s)');
ylabel('angle (rad)');
hold on;
% 0.01 time step
```

```
[T, X] = eulerF(odefun, tspan, x0);
plot(T, X(:, 1));
% 0.1 time step
tspan = [0:0.1:10];
[T, X] = eulerF(odefun, tspan, x0);
plot(T, X(:, 1));
plot(Tode, Xode(:, 1));
legend('0.01 time step', '0.1 time step', 'ode45', 'Location','southwest');
hold off;
fprintf('\n1.g if we use a smaller time step, the accuracy of the model is
closer to what we get from ode45\n');

% 1.h simulate using model linearlized via central difference
[Acd, Bcd] = cdLin(xeq, ueq);
odeLinSim = @(t, x) Acd*x + Bcd*uIn(t);
tspan = [0:0.01:10];
x0 = [0; 0];
[T1, X1] = ode45(odeLinSim, tspan, x0(:)-xeq(:));
figure(3);
hold on;
title('1.h pendulum angle over time using central difference linearlized
model');
xlabel('time (s)');
ylabel('angle (rad)');
plot(T1, X1(:, 1)+xeq(1));
plot(Tode, Xode(:, 1));
legend('linearized', 'ode45', 'Location','southwest');
hold off;
fprintf('\n1.h the accuracy of the linearized model is good, it is very close
to the simulation done using ode45\n');

% functions need to be at the bottom??
% 1.b pendulum model
function xdot = pendModel(x, u)
    g = 9.81;
    m = 1;
    l = 10;
    xdot = [x(2); -g/l*sin(x(1)) + u/(m*l^2)];
end

% 1.c find equilibrium point
function xeq = findPendEq(ueq)
    xeq = fsolve(@(x) pendModel(x, ueq), [0; 0]);
end

% 1.d linearize pendulum model symbolically
function [A, B] = symLin(x0, u0)
    syms x1 x2 u
    x = [x1; x2];
    f = pendModel(x, u);
    A = jacobian(f, x);
    A = subs(A, [x; u], [x0; u0]);
    B = jacobian(f, u);
    B = subs(B, [x; u], [x0; u0]);
```

```
end

% 1.e linearize pendulum model numerically
function [A, B] = cdLin(x0, u0)
    epsilon = 1e-6;

    A = zeros(2, 2);
    for i = 1:2
        x = x0;
        x(i) = x(i) + epsilon;
        f1 = pendModel(x, u0);
        x = x0;
        x(i) = x(i) - epsilon;
        f2 = pendModel(x, u0);
        A(:, i) = (f1 - f2)/(2*epsilon);
    end

    B = zeros(2, 1);
    u = u0;
    u = u + epsilon;
    f1 = pendModel(x0, u);
    u = u0;
    u = u - epsilon;
    f2 = pendModel(x0, u);
    B = (f1 - f2)/(2*epsilon);
end

% 1.f simulate input over time
function u = uIn(t)
    u = 2*sin(t);
end

% 1.g simulate using forward euler
function [T, X] = eulerF(odefun, tspan, x0)
    T = tspan;
    time_step = tspan(2) - tspan(1);
    X = zeros(length(tspan), length(x0));
    X(1, :) = x0;
    for i = 2:length(tspan)
        % use the (') to transpose output from odefun so it is 1x2
        X(i, :) = X(i-1, :) + time_step*odefun(T(i-1), X(i-1, :))';
    end
end
```

*Equation solved.*

*fsolve completed because the vector of function values is near zero*
*as measured by the value of the function tolerance, and*
*the problem appears regular as measured by the gradient.*

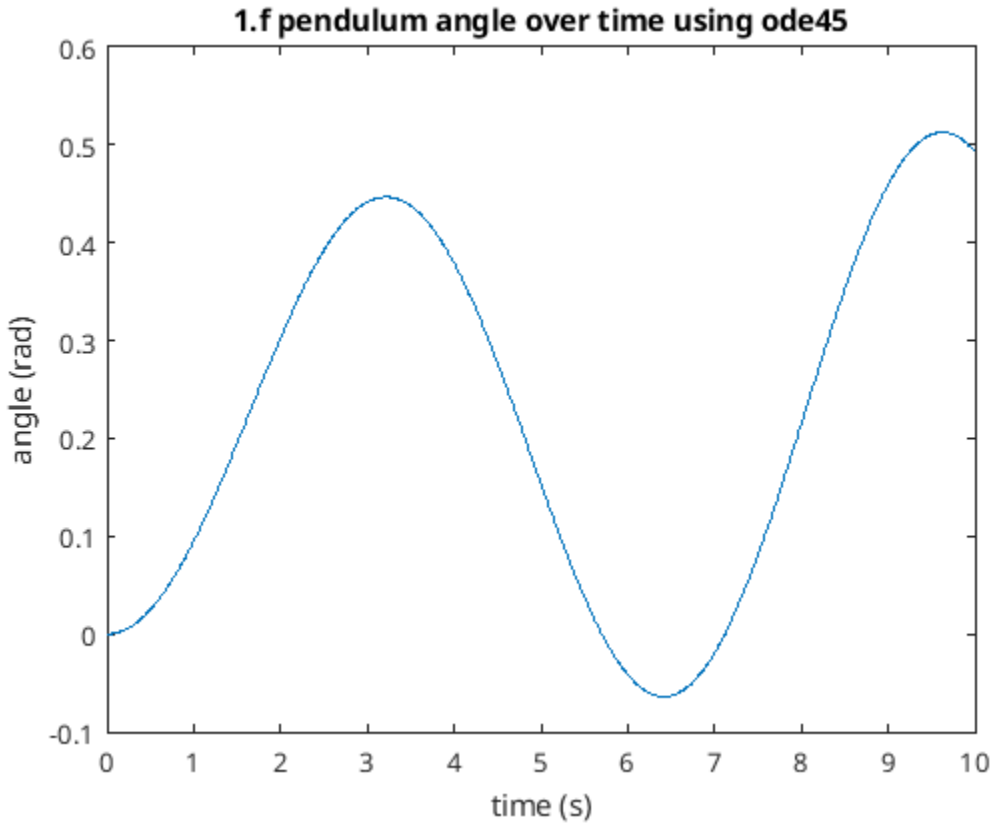
*1.c equilibrium at u = 20*
*xeq = [0.205313; 0.000000]*

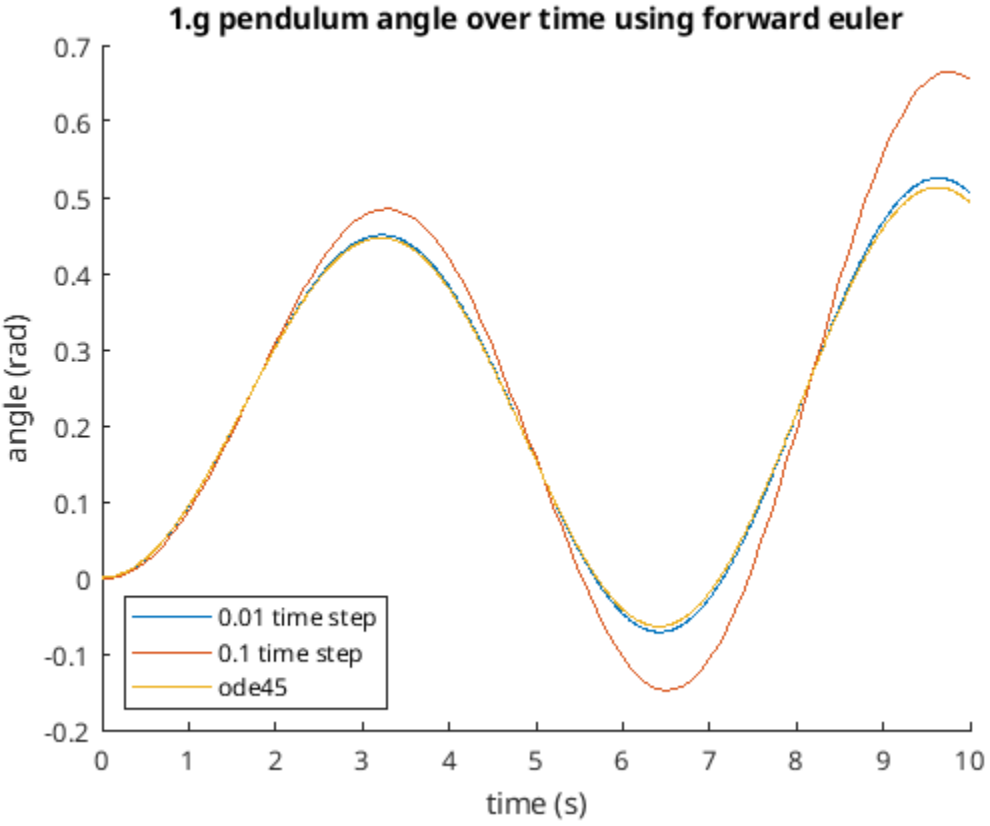
*1.d symbolically linearized model at u = 20 and x = [0.205313; 0.000000]*
*A =*
*[0.000000, 1.000000;*
*-0.960396, 0.000000]*
*B =*
*[0.000000;*
*0.010000]*

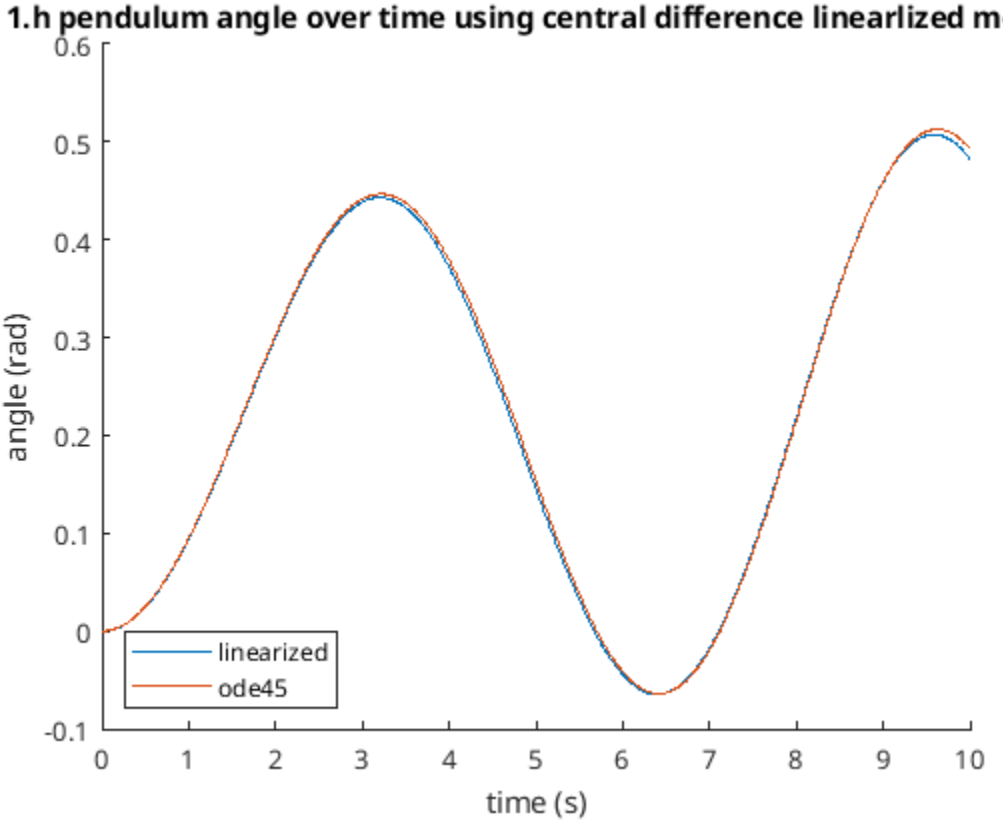
*1.e numerically linearized model at u = 20 and x = [0.205313; 0.000000]*
*A =*
*[0.000000, 1.000000;*
*-0.960396, 0.000000]*
*B =*
*[0.000000;*
*0.010000]*

*1.g if we use a smaller time step, the accuracy of the model is closer to what we get from ode45*

*1.h the accuracy of the linearized model is good, it is very close to the simulation done using ode45*




1.g pendulum angle over time using forward euler
angle (rad)
time (s)
0.01 time step
0.1 time step
ode45


1.h pendulum angle over time using central difference linearlized model
angle (rad)
time (s)
linearized
ode45

*Published with MATLAB® R2023b*