Week 6 - Introduction to ode45()

MAT330: Differential Equations

Student Name JoAnn Olney

Date 6/11/25

Introduction:

Differential equations and systems of differential equations can be solved numerically using the ode45() solver. Numerical solvers can be useful when a closed-form analytic solution may not exist, or may be difficult to compute.

In this MATLAB assignment, you'll learn how to use the ode45() function to find solutions to systems of differential equations.

Review the code and comments provided in the "Examples" section below, and then use this information to complete the problems listed in the "Problems" section.

Make sure to run your code so all relevant computations/results are displayed, delete the "Introduction" and "Examples" sections, and then export your work as a PDF file for submission (your submission only needs to contain the "Problems" section that you completed).

Examples:

```
% Example 1 Code
% The example below shows how to solve a system of differential equations
% using the ode45() function.
%
% Consider the system of differential equations:
% x1' = -1*x1 + 5*x2
% x2' = -3*x1 - 2*x2
%
% To solve this system of differential equations, a function that defines
% relationships between all the variables and their differentials must be
% created.

% Create the function "example01ODEFunction.m" and save it in the workspace.
% The contents of the function should look like this:

% function dXdt = example01ODEFunction(t,X)
%
%     x1 = X(1);
```

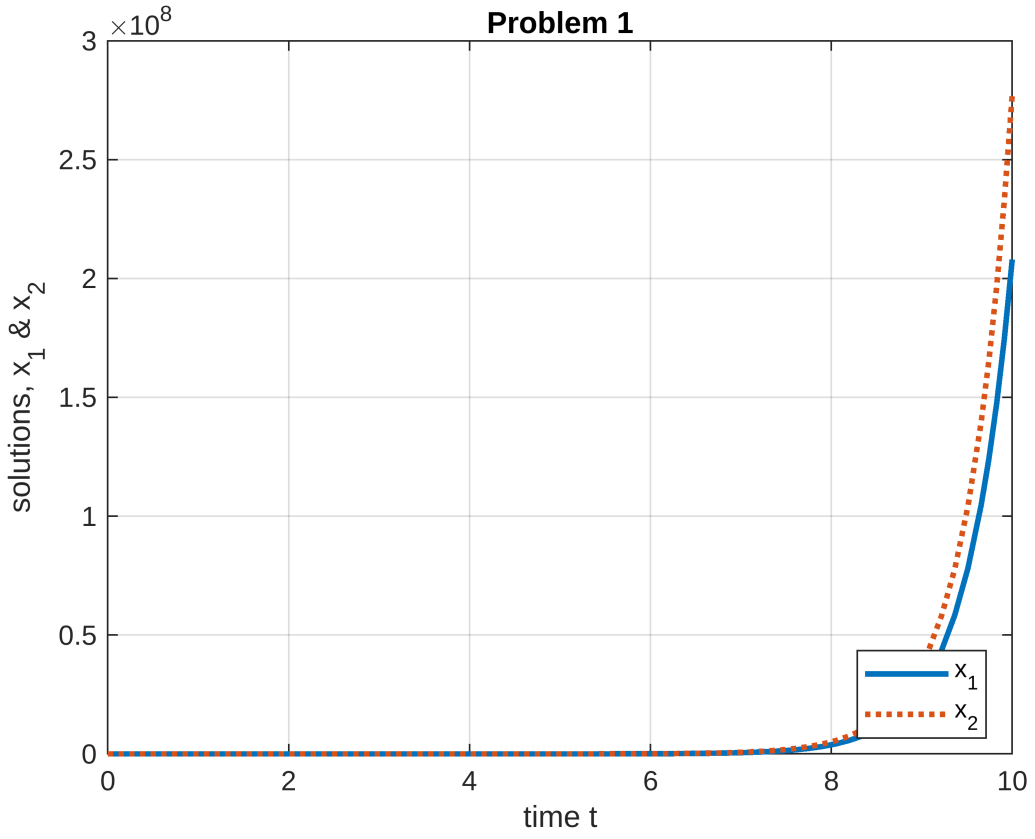
```
%     x2 = X(2);
%
%     dx1dt = -1*x1 + 5*x2;
%     dx2dt = -3*x1 - 2*x2;
%
%     dXdt = [dx1dt; dx2dt];
%
% end

% The first input to the function will always be the independent variable,
% usually the variable "t". The second input is always a vector containing
% current values of x1, x2, etc.
%
% The function must compute the derivatives at a time. In this example,
% all the derivatives are explicitly stated in the system of equations,
% so writing the equations for dx1dt and dx2dt is simple.
% Your function could also contain other time-varying functions. An example
% of this is shown in Example 2 below.

% It's often necessary to set relative and absolute tolerences of the solver
% so it knows how long to run before deciding it's converged on a solution.
% The "RelTol" parameter for relative tolerance, and "AbsTol" parameter
% for absolute tolerance can be set using "odeset":
%opts = odeset('RelTol',1e-6,'AbsTol',1e-8);

% Call the ode45() function to solve the system of differentials equations
%     The 2nd input vector [0 5] specifies to solve for t = 0 to 5
%
%     The 3rd input vector [0.1 1] specifies the initial conditions. In
%     this case the initial conditions are x1(0) = 0.1 and x2(0) = 1.
%
%     The final input "opts" tells the solver to use the RelTol and AbsTol
%     preferences set above.

%soln = ode45(@1ODEFunction,[0 5],[0.1 1],opts);

% The output "soln" is a structure that contains the independent variable
% stored in soln.x, and the function solutions stored as rows in soln.y.
% For this example, x1(t) = soln.y(1,:), the first row of y. The signal
% x2(t) = soln.y(2,:), the second row of y. This pattern would continue for
% larger systems of equations. We can make a plot of the solutions using
% our normal plotting routines.
%figure;
%plot(soln.x,soln.y(1,:),'-','linewidth',2);
%hold on;
%plot(soln.x,soln.y(2,:),':','linewidth',2);
%grid on;
%title('Example Problem ODE45 Solution');
%xlabel('Time (s)');
%ylabel('Solution');
```

```
% Calling legend with 'location','best' as the final two inputs will tell
% legend() to attempt to place the legend in a spot where it doesn't
% overlap with the plots.
%legend('x_{1}(t)','x_{2}(t)','location','best');
% Example 2 Code
%
% The example below works with a similar system of equations, but now there
% are additional functions that also vary with time.

% Create the function "example02ODEFunction.m" and save it in the workspace.
% The contents of the function should look like this:

% function dXdt = example02ODEFunction(t,X)
%
%     x1 = X(1);
%     x2 = X(2);
%
%     dx1dt = -1*x1 + 5*x2 + heaviside(t-1);
%     dx2dt = -3*x1 - 2*x2 + (1-exp(-t))*heaviside(t-4);
%
%     dXdt = [dx1dt; dx2dt];
%
% end

% Note, this system of equations is similar to Example 1, but now includes
% other time-varying functions that turn on at time t=1, and and time t =
% 4 respectively.

% Get the solution
%opts = odeset('RelTol',1e-6,'AbsTol',1e-8);
%soln = ode45(@example02ODEFunction,[0 10],[0.1 1],opts);

% Plot the solutions
%figure;
%plot(soln.x,soln.y(1,:),'-','linewidth',2);
%hold on;
%plot(soln.x,soln.y(2,:),':','linewidth',2);
%grid on;
%title('Example Problem 2 ODE45 Solution');
%xlabel('Time (s)');
%ylabel('Solution');
%legend('x_{1}(t)','x_{2}(t)','location','best');
% Examining the solution above, we can see the effect that inputs that turn
% on at t = 1 and t = 4 have on the solution as the solution behavior
% starts changing at those times. You can also see that each solution does
% meet the specified intial conditions.
```

Problems:

Problem 1: Consider the system of differential equations $x_1' = x_1 + 2x_2$, $x_2' = 3x_1 + 2x_2$ with initial conditions $x_1(0) = 1, x_2(0) = 1$. Use the MATLAB ode45() function to solve this system of differential equations and then plot the solution on a single plot for time $t = 0$ to $t = 5$. Plot $x_1(t)$ as a solid line, $x_2(t)$ as a dotted line, and turn on the plotting grid and legend. Make sure to label your axes. Explain what happens to the solutions as $t \to \infty$. Does this answer change if the initial conditions change?

```
% Problem 1 Code Here
%define the system of differential equations
function dXdt=problem1ODE(~,X)
x1 = X(1);
x2 = X(2);
dx1dt = -2*x1 + 3*x2;
dx2dt = 4*x1 - x2;
dXdt = [dx1dt; dx2dt];
end

%initial conditions
X0 = [1, 0];

%time span
tspan = [0, 10];

%solve the system using ode45
[t, X] = ode45(@problem1ODE, tspan, X0);

%plot the solution
figure;
plot(t, X(:,1), '-', 'LineWidth', 2);
%hold
hold on
plot(t, X(:,2), ':', 'LineWidth', 2);
%hold
hold off;
%grid on
grid on;
%label axes and title
xlabel('time t');
ylabel('solutions, x_1 & x_2');
title('Problem 1');
%legend
legend ('x_1', 'x_2', 'Location', 'best');
```



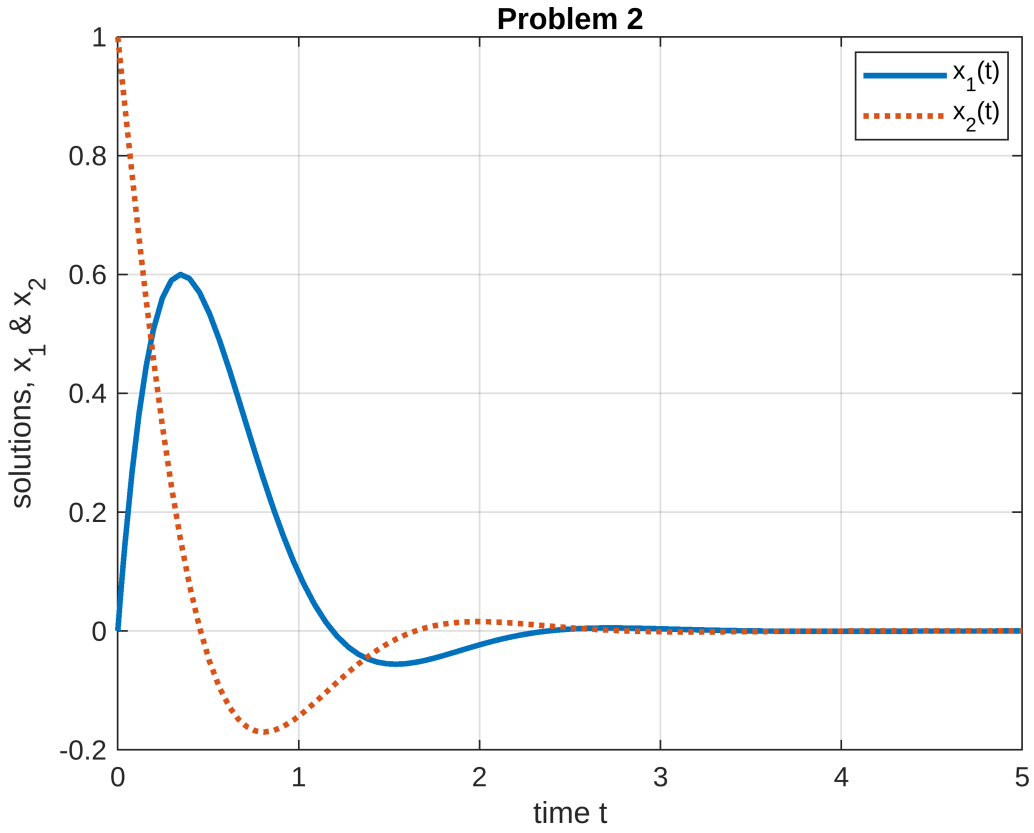
Put explanation/math here...

With the given initial conditions of all zeros, both solutions will remain at zero for all time since there are no external forcings in the system. As the solution as t approaches infinity will grow without bound, depending on the dynamics dictated by the differential equations' parameters and initial state, here being positive infinity after 8 time (t).

Problem 2: Consider the system of differential equations $x_1' = x_1 - 15x_2$, $x_2' = 2x_1 - 5x_2$ with initial conditions $x_1(0) = 2, x_2(0) = 5$. Use the MATLAB ode45() function to solve this system of differential equations and then plot the solution on a single plot for time $t = 0$ to $t = 5$. Plot $x_1(t)$ as a solid line, $x_2(t)$ as a dotted line, and turn on the plotting grid and legend. Make sure to label your axes. Explain what happens to the solutions as $t \to \infty$.

```
% Problem 2 Code Here
%define the system of differential equations
function dXdt = problem2ODE(t, X)
x1 = X(1);
x2 = X(2);
dx1dt = -x1 + 4*x2;
```

```
dx2dt = -2*x1 - 3*x2;
dXdt = [dx1dt; dx2dt];
end

%initial condition
X0 = [0; 1];

%time span
tspan = [0, 5];

%solve the system using ode45
[t, X] = ode45(@problem2ODE, tspan, X0);

%plotting the solution
figure;
plot(t, X(:,1), '-', 'LineWidth', 2);
%hold
hold on;
plot(t, X(:,2), ':', 'LineWidth', 2);
%grid on
grid on;
%label axes and title
xlabel('time t');
ylabel('solutions, x_1 & x_2');
title('Problem 2');
%legend
legend('x_1(t)', 'x_2(t)', 'Location', 'best');
```



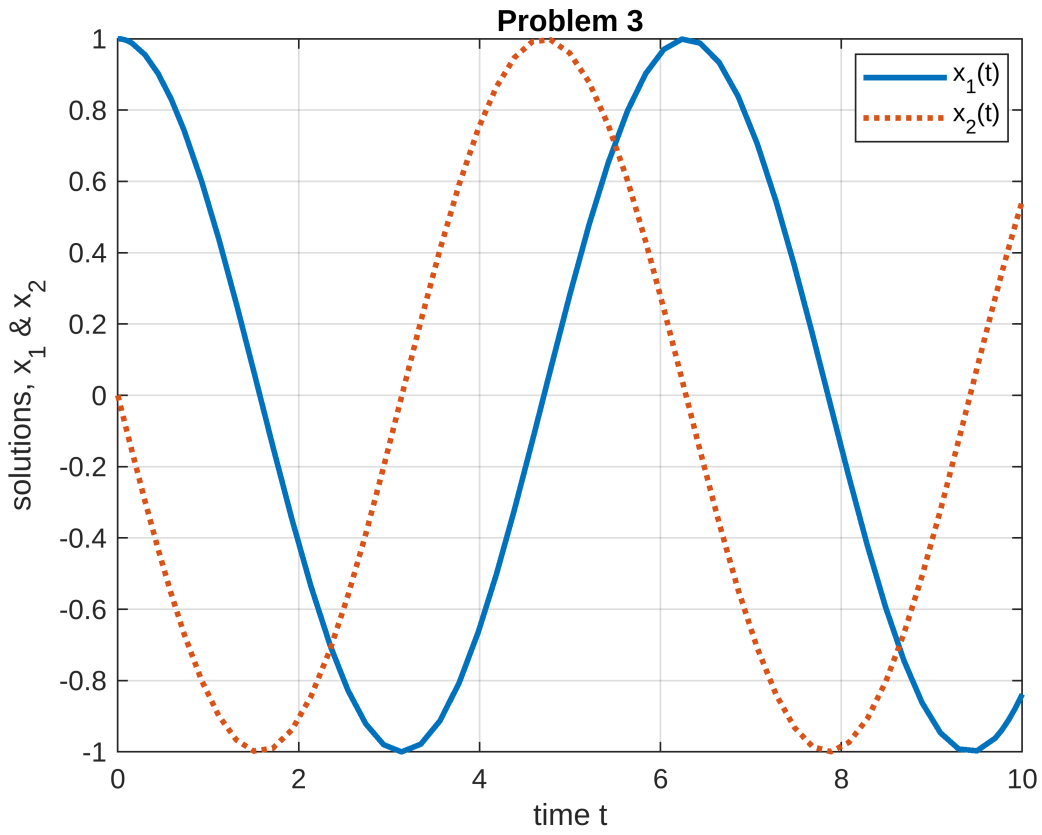
Put explanation/math here...

The behavior of the solutions as $t \to \infty$ depends on the eigenvalues of the system matrix. Calculating the eigenvalues of A helps determine stability. If the system is stable as t approaches infinity x1 and x2 with approach 0. If the system is not stable it can grow without bound.

Problem 3: Consider the system of differential equations $x_1' = x_1 - 15x_2 + 20\cos(6\pi t)u(t-3)$, $x_2' = 2x_1 - 5x_2$ with initial conditions $x_1(0) = 10, x_2(0) = 0$. Use the MATLAB ode45() function to solve this system of differential equations and then plot the solution on a single plot for time $t = 0$ to $t = 10$. Plot $x_1(t)$ as a solid line, $x_2(t)$ as a dotted line, and turn on the plotting grid and legend. Make sure to label your axes. Explain what happens to the solutions as $t \to \infty$.

```
% Problem 3 Code Here
%define the system of differential equations
function dXdt=problem3ODE(t, X)
x1 = X(1);
x2 = X(2);
dx1dt = x2;
dx2dt = -x1;
dXdt = [dx1dt; dx2dt];
```

```
end

%define the initial conditions
X0 = [1; 0];

%define the time span
tspan = [0, 10];

%solve the differential equation using ode45
[t, X] = ode45(@problem3ODE, tspan, X0);

%plot the solution
figure;
plot(t,X(:,1), '-', 'LineWidth', 2);
%hold
hold on;
plot(t,X(:,2), ':', 'LineWidth', 2);
grid on;
%label axes and title
xlabel('time t');
ylabel('solutions, x_1 & x_2');
title('Problem 3');
%legend
legend('x_1(t)', 'x_2(t)', 'Location', 'best');
```



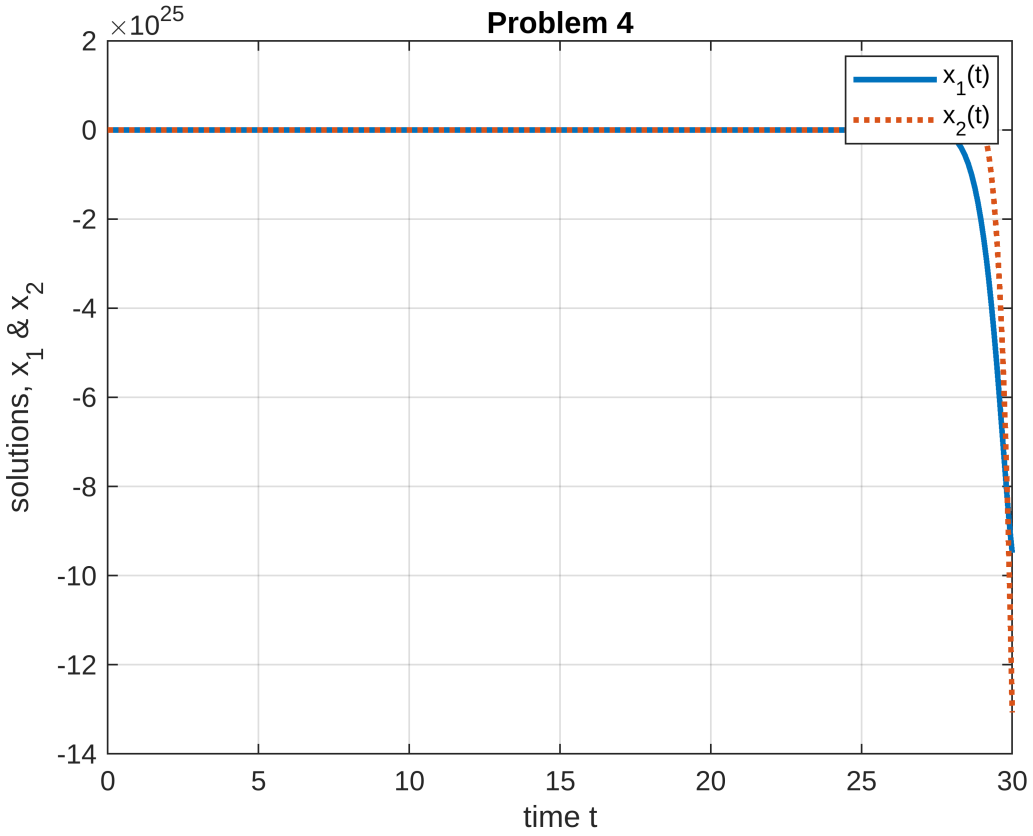
Put explanation/math here...

The solutions *x*1(*t*) and *x*2(*t*) will remain bounded and continue oscillating since there is no external force applied to the system. The given system is a simple harmonic oscillator that behaves sinusoidally. It has the characteristic property that *x*1 and *x*2 will oscillate indefinitely over time because the equations describe the conservation of energy.

Problem 4: Consider the system of differential equations $x_1' = x_1 - 5x_2 + e^{-0.2t}u(t-1)$, $x_2' = 2x_1 - 5x_2 - 0.5x_1' + 10(1 - e^{-0.5(t-5)})u(t-5)$ with initial conditions $x_1(0) = -7, x_2(0) = 2$. Use the MATLAB ode45() function to solve this system of differential equations and then plot the solution on a single plot for time $t = 0$ to $t = 30$. Plot $x_1(t)$ as a solid line, $x_2(t)$ as a dotted line, and turn on the plotting grid and legend. Make sure to label your axes. As $t \to \infty$, what do $\frac{dx_1}{dt}$ and $\frac{dx_2}{dt}$ converge to? Explain.

```
% Problem 4 Code Here
%define the system of differential equations
function dXdt=problem4ODE(t,X)
x1 = X(1);
x2 = X(2);
dx1dt = 2*x1-x2;
dx2dt = x1+2*x2;
dXdt = [dx1dt; dx2dt];
end

%define the inital condition
X0 = [1; -1];

%the time span
tspan = [0, 30];

%solve the differential equation using ode45
[t, X] = ode45(@problem4ODE, tspan, X0);

%plot the solution
figure;
plot(t, X(:,1), '-', 'LineWidth', 2);
%hold
hold on;
plot(t, X(:,2), ':', 'LineWidth', 2);
grid on;
%label axes and title
xlabel('time t');
```

```
ylabel('solutions, x_1 & x_2');
title('Problem 4');
%legend
legend('x_1(t)', 'x_2(t)', 'Location', 'best');
```



Put explanation/math here...

The real parts of the eigenvalues are positive , indicating that both $x1$ and $x2$ grow exponentially without converging to a steady state as $t \to \infty$. The presence of imaginary parts means that the solutions will also oscillate.