Course Notes February 12, 2026

Tutorial 1: An Overview of MATLAB

Problem 3

Suppose that $x = 5$ and $y = 2$. Use MATLAB to compute the following, and check the results with a calculator.

a. $(1 - \frac{1}{x^5})^{-1}$

b. $3\pi x^2$

c. $\frac{3y}{4x-8}$

d. $\frac{4(y-5)}{3x-6}$

```
clear; clc;
x = 5;
y = 2;

% a. (1 - 1/x^5)^-1
result_a = (1 - 1/x^5)^-1;

% b. 3 * pi * x^2
result_b = 3 * pi * x^2;

% c. (3*y) / (4*x - 8)
result_c = (3*y) / (4*x - 8);

% d. (4*(y - 5)) / (3*x - 6)
result_d = (4*(y - 5)) / (3*x - 6);

% Display results
disp(table(result_a, result_b, result_c, result_d));
```

Problem 5

Assuming that the variables a, b, c, d, and f are scalars, write MATLAB statements to compute and display the following expressions. Test your statements for the values $a = 1.12$, $b = 2.34$, $c = 0.72$, $d = 0.81$ and $f = 19.83$.

- $x = 1 + \frac{a}{b} + \frac{c}{f^2}$
- $r = \frac{1}{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}+\frac{1}{d}}$
- $s = \frac{b-a}{d-c}$
- $y = ab\frac{1}{c}\frac{f^2}{2}$

```
clear; clc;
a = 1.12; b = 2.34; c = 0.72; d = 0.81; f = 19.83;

x = 1 + a/b + c/f^2;
r = 1 / (1/a + 1/b + 1/c + 1/d);
s = (b - a) / (d - c);
y = a * b * (1/c) * (f^2/2);

disp(['x = ', num2str(x)]);
disp(['r = ', num2str(r)]);
disp(['s = ', num2str(s)]);
disp(['y = ', num2str(y)]);
```

Problem 9

The functions `realmax` and `realmin` give the largest and smallest possible numbers that can be handled by MATLAB. Suppose you have variables $a = 3 \times 10^{150}$, $b = 5 \times 10^{200}$.

a. Use MATLAB to calculate $c = ab$.

b. Supposed $d = 5 \times 10^{-200}$ use MATLAB to calculate $f = d/a$.

c. Use MATLAB to calculate the product $x = abd$ two ways.

```
% Check limits
realmax
realmin

a = 3e150;
b = 5e200;

% a. Calculate c = a*b (Expect Overflow)
c = a * b

```

```
% b. d = 5e-200, calculate f = d/a (Expect Underflow)
d = 5e-200;
f = d / a

% c. Calculate x = abd in two ways
x1 = a * b * d; % Risk of intermediate overflow
y = b * d;
x2 = a * y;      % Safer calculation

disp(['Method 1: ', num2str(x1)]);
disp(['Method 2: ', num2str(x2)]);
```

Problem 22

Use MATLAB to calculate:

a. $e^{(-2.1)^3} + 3.47\log(14) + \sqrt[4]{287}$

b. $(3.4)^7 \log(14) + \sqrt[4]{287}$

c. $\cos^2(\frac{4.12\pi}{6})$

d. $\cos(\frac{4.12\pi}{6})^2$

```
% Note: Source likely implies log base 10 for "log(14)" in standard notation,
% but MATLAB's log() is natural log. Using log10() for base 10.
ans_a = exp((-2.1)^3) + 3.47 * log10(14) + nthroot(287, 4);
ans_b = (3.4)^7 * log10(14) + nthroot(287, 4);
ans_c = cos((4.12 * pi) / 6)^2;
ans_d = cos(((4.12 * pi) / 6)^2);
```

Problem 27

Use MATLAB to plot the function $T = 7\ln t - 8e^{0.3t}$ over the interval $1 \leq t \leq 3$.

```
t = 1:0.01:3;
T = 7 .* log(t) - 8 .* exp(0.3 .* t);

plot(t, T);
title('Temperature vs Time');
xlabel('Time (min)');
ylabel('Temperature (C)');
grid on;
```

Problem 30

A cycloid is described by $x = r(\phi - \sin\ \phi)$ and $y = r(1 - \cos\ \phi)$. Plot for $r = 10$ and $0 \leq \phi \leq 4\pi$.

```
r = 10;
phi = 0 : 0.01 : 4*pi;
x = r .* (phi - sin(phi));
y = r .* (1 - cos(phi));

plot(x, y);
title('Cycloid Plot (r=10)');
xlabel('x'); ylabel('y');
axis equal;
```

Problem 34

Develop a procedure for computing the length of side c_2 of the two-triangle figure given sides b_1, b_2, c_1 and angles A_1, A_2. Test with $b_1 = 200$, $b_2 = 1801$, $c_1 = 1201$, $A_1 = 120°$, $A_2 = 100°$.

```
% Inputs
b1 = 200; b2 = 1801; c1 = 1201;
A1_deg = 120; A2_deg = 100;
A1 = deg2rad(A1_deg); A2 = deg2rad(A2_deg);

% 1. Find common side 'a' (Top Triangle Law of Cosines)
a_sq = b1^2 + c1^2 - 2*b1*c1*cos(A1);
a = sqrt(a_sq);

% 2. Find c2 (Bottom Triangle) solving quadratic:
% c2^2 - (2*b2*cos(A2))*c2 + (b2^2 - a^2) = 0
coeff_A = 1;
coeff_B = -2 * b2 * cos(A2);
coeff_C = b2^2 - a_sq;

possible_c2 = roots([coeff_A, coeff_B, coeff_C]);
c2 = possible_c2(possible_c2 > 0); % Filter positive

disp(['Side c2: ', num2str(c2')]);
```

Problem 35

Write a script to compute the three roots of $x^3 + ax^2 + bx + c = 0$.

```
a = input('Enter a: ');
b = input('Enter b: ');
c = input('Enter c: ');
disp(roots([1, a, b, c]));
```

Tutorial 2: Numeric, Cell and Structure Arrays

Problem 10

Consider the array $A = \begin{bmatrix} 1 & 4 & 2 \\ 2 & 4 & 100 \\ 7 & 9 & 7 \\ 3 & \pi & 42 \end{bmatrix}$ and $B = \ln(A)$.

Write MATLAB expressions to do the following:

a. Select just the second row of B.

b. Evaluate the sum of the second row of B.

c. Multiply the second column of B and the first column of A element by element.

d. Evaluate the maximum value in the vector resulting from element-by-element multiplication of the second column of B with the first column of A.

e. Use element-by-element division to divide the first row of A by the first three elements of the third column of B. Evaluate the sum of the elements of the resulting vector.

```
% Define Matrix A
A = [1, 4, 2;
     2, 4, 100;
     7, 9, 7;
     3, pi, 42];

% Define Matrix B (Natural log is log() in MATLAB)
B = log(A);

% a. Select second row of B
part_a = B(2, :);

% b. Sum of second row of B
part_b = sum(B(2, :));

% c. Multiply 2nd col of B and 1st col of A element-wise
part_c = B(:, 2) .* A(:, 1);

% d. Max value of result from c
part_d = max(part_c);
```

```
% e. Divide 1st row of A by first 3 elements of 3rd col of B
% Note: A(1,:) is 1x3. B(1:3, 3) is 3x1.
% We must transpose B's slice to match dimensions.
vec_e = A(1, :) ./ B(1:3, 3)';
part_e = sum(vec_e);

disp(['Sum (Part b): ', num2str(part_b)]);
disp(['Max (Part d): ', num2str(part_d)]);
disp(['Sum (Part e): ', num2str(part_e)]);
```

Problem 11

Create a three-dimensional array D whose three "layers" are matrices A, B, and C. Use MATLAB to find the largest element in each layer of D and the largest element in D.

```
A = [3, -2, 1; 6, 8, -5; 7, 9, 10];
B = [6, 9, -4; 7, 5, 3; -8, 2, 1];
C = [-7, -5, 2; 10, 6, 1; 3, -9, 8];

% Create 3D array D
D(:, :, 1) = A;
D(:, :, 2) = B;
D(:, :, 3) = C;

% Largest element in each layer
max_layer_1 = max(max(D(:, :, 1)));
max_layer_2 = max(max(D(:, :, 2)));
max_layer_3 = max(max(D(:, :, 3)));

% Largest element in D
max_total = max(D(:));

disp(['Max Total: ', num2str(max_total)]);
```

Problem 15

Given matrices A, B, and C, verify the associative and commutative laws for addition.

```
A = [-7, 11; 4, 9];
B = [4, -5; 12, -2];
C = [-3, -9; 7, 8];

% a. A + B + C
res_a = A + B + C;

```

```
% b. A - B + C
res_b = A - B + C;

% c. Verify Associative Law: (A+B)+C = A+(B+C)
check_assoc = isequal((A+B)+C, A+(B+C));

% d. Verify Commutative Law: A+B+C = B+C+A = A+C+B
term1 = A + B + C;
term2 = B + C + A;
term3 = A + C + B;
check_comm = isequal(term1, term2) && isequal(term2, term3);

if check_assoc && check_comm
    disp('Laws Verified');
else
    disp('Verification Failed');
end
```

Problem 19

Plot the function $f(x) = \frac{4\cos x}{x+e^{-0.75x}}$ over the interval $-2 \le x \le 16$.

```
x = -2 : 0.05 : 16; % Smooth interval
f = (4 .* cos(x)) ./ (x + exp(-0.75 .* x));

plot(x, f);
title('Plot of f(x)');
xlabel('x');
ylabel('f(x)');
grid on;
```

Problem 22

A ship travels on a straight line course described by $y = (200 - 5x)/6$. The ship starts when $x = -20$ and ends when $x = 40$. Calculate the distance at closest approach to a lighthouse located at the origin (0,0) without using a plot.

```
% Define path range
x = -20 : 0.01 : 40;
y = (200 - 5 .* x) ./ 6;

% Distance formula d = sqrt(x^2 + y^2)
distances = sqrt(x.^2 + y.^2);

% Find minimum distance
```

```
min_dist = min(distances);

disp(['Closest approach distance: ', num2str(min_dist), ' km']);
```

Problem 23

Calculate work done $W = FD$ for five segments of a path given force and distance data. Find (a) work for each segment and (b) total work.

```
% Data vectors
Force = [400, 550, 700, 500, 600]; % Newtons
Distance = [3, 0.5, 0.75, 1.5, 5]; % Meters

% a. Work per segment (Element-wise multiplication)
Work_segments = Force .* Distance;

% b. Total work
Work_total = sum(Work_segments);

disp('Work per segment (J):');
disp(Work_segments);
disp(['Total Work (J): ', num2str(Work_total)]);
```

Problem 27

Calculate compression x and potential energy $PE = \frac{1}{2}kx^2$ for five springs given Force $F = kx$ and spring constant k.

```
% Data
F = [11, 7, 8, 10, 9];           % Force (N)
k = [1000, 600, 900, 1300, 700]; % Constant (N/m)

% a. Compression x = F / k
x = F ./ k;

% b. Potential Energy PE = 0.5 * k * x^2
PE = 0.5 .* k .* (x.^2);

% Display results table
disp(table(F', k', x', PE', 'VariableNames', {'Force','k','Compression','PE'}));
```

Problem 41

Solve the following system using the left-division method.

$$
\begin{aligned}
6x - 3y + 4z &= 41 \\
12x + 5y - 7z &= -26 \\
-5x + 2y - 6z &= 16
\end{aligned}
$$

```
% Coefficient Matrix A
A = [ 6, -3,  4;
     12,  5, -7;
     -5,  2, -6];

% Constant Vector B
B = [41; -26; 16];

% Solve for X = [x; y; z] using left division
Solution = A \ B;

disp('Solution [x; y; z]:');
disp(Solution);
```

Tutorial 3: Functions

Problem 10

An object thrown vertically with a speed v_0 reaches a height h at time t, where $h = v_0 t - \frac{1}{2}gt^2$. Write and test a function that computes the time t required to reach a specified height h, for a given value of v_0. The function's inputs should be h, v_0, g. Test for $h = 100$ m, $v_0 = 50$ m/s, $g = 9.81 m/s^2$.

```
% --- Main Script ---
h = 100; v0 = 50; g = 9.81;

% Call the function
t_solutions = compute_time(h, v0, g);

disp('Times to reach 100m (seconds):');
disp(t_solutions);
% Interpretation: The object reaches 100m twice.
% Once on the way up, and once on the way down.

% --- Function Definition ---
function t = compute_time(h, v0, g)
    % Solves 0.5*g*t^2 - v0*t + h = 0
    % Using quadratic formula: ax^2 + bx + c = 0
    % a = 0.5*g, b = -v0, c = h

    roots_vec = roots([0.5*g, -v0, h]);
    t = roots_vec;
end
```

Problem 17

The volume and paper surface area A of a conical paper cup are given by $V = \frac{1}{3}\pi r^2 h$ and $A = \pi r\sqrt{r^2 + h^2}$.

a. Eliminate h to obtain A as a function of r and V.

b. Create a function for A and use **fminbnd** to find r that minimizes A for $V = 10$ in^3.

```
% --- Main Script ---
global V
V = 10; % Volume constraint

% Minimize Area function between r=0.1 and r=10
[r_min, A_min] = fminbnd(@cone_area, 0.1, 10);

% Calculate corresponding h
h_min = 3 * V / (pi * r_min^2);

disp(['Optimal r: ', num2str(r_min)]);
disp(['Optimal h: ', num2str(h_min)]);
disp(['Minimum Area: ', num2str(A_min)]);

% --- Function Definition ---
function A = cone_area(r)
    global V
    % Eliminate h: h = 3V / (pi*r^2)
    h = 3 * V ./ (pi .* r.^2);
    % Substitute into A
    A = pi .* r .* sqrt(r.^2 + h.^2);
end
```

Problem 18

A torus with inner radius a and outer radius b has volume $V = \frac{1}{4}\pi^2(a+b)(b-a)^2$ and surface area $A = \pi^2(b^2 - a^2)$.

a. Create a function for V and A.

b. Plot A vs a for $0.25 \leq a \leq 4$ given $b = a + 2$.

```
% --- Main Script ---
a = 0.25 : 0.01 : 4;
b = a + 2; % Constraint

% Compute A and V using arrays
[V, A] = torus_calc(a, b);

plot(a, A);
title('Torus Surface Area vs Inner Radius a');
xlabel('a (inches)');
ylabel('Surface Area A');
grid on;

% --- Function Definition ---
function [V, A] = torus_calc(a, b)
    V = 0.25 * pi^2 .* (a + b) .* (b - a).^2;
    A = pi^2 .* (b.^2 - a.^2);
```

```
end
```

Problem 21

Create a function that will plot the entire ellipse $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$, given inputs a and b. Test for $a = 1, b = 2$.

```
% --- Main Script ---
plot_ellipse(1, 2);

% --- Function Definition ---
function plot_ellipse(a, b)
    % Use parametric equations for full ellipse
    t = linspace(0, 2*pi, 100);
    x = a * cos(t);
    y = b * sin(t);

    figure;
    plot(x, y);
    title(['Ellipse: a=', num2str(a), ', b=', num2str(b)]);
    axis equal;
    grid on;
end
```

Problem 25

Create an anonymous function for $30x^2 - 300x + 4$.

a. Plot to approximate minimum.

b. Use **fminbnd** to determine the precise minimum location.

```
f = @(x) 30*x.^2 - 300*x + 4;

% a. Plotting
x_plot = -5:0.1:15;
plot(x_plot, f(x_plot));
grid on; title('Plot of 30x^2 - 300x + 4');

% b. Finding minimum
[x_min, val_min] = fminbnd(f, 0, 10);
disp(['Minimum occurs at x = ', num2str(x_min)]);
```

Problem 31

Estimate the three coefficients a, b, c of the logistic growth model $y(t) = \frac{c}{1+ae^{-bt}}$ using the provided data and **fminsearch**.

```
% Data
t = 0:15;
y_data = [13, 16, 20, 25, 31, 39, 45, 49, 55, 63, 69, 77, 82, 86, 89, 92];

% Model Function: y = c / (1 + a*exp(-b*t))
model_fun = @(p, t) p(3) ./ (1 + p(1) * exp(-p(2) * t));

% Error Function (Sum of Squared Errors)
err_fun = @(p) sum((y_data - model_fun(p, t)).^2);

% Initial Guess: c around 100 (max percent), a and b generic guesses
guess = [10, 0.5, 100];

% Optimization
p_opt = fminsearch(err_fun, guess);
a_est = p_opt(1); b_est = p_opt(2); c_est = p_opt(3);

% Plotting results
t_smooth = 0:0.1:15;
y_fit = model_fun(p_opt, t_smooth);

plot(t, y_data, 'ko', t_smooth, y_fit, 'b-');
legend('Data', 'Logistic Fit');
title('Logistic Growth Regression');
disp(['Estimated: a=',num2str(a_est),', b=',num2str(b_est),', c=',num2str(c_est)]);
```

Tutorial 4: Programming with MATLAB

Problem 2

The roots of $ax^2 + bx + c = 0$ are $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$. Write a program to compute both roots, identifying real and imaginary parts. Test for cases: (1) $2, 10, 12$ (2) $3, 24, 48$ (3) $4, 24, 100$.

```
% Define test cases
cases = [2, 10, 12;
         3, 24, 48;
         4, 24, 100];

for i = 1:size(cases, 1)
    a = cases(i, 1); b = cases(i, 2); c = cases(i, 3);

    disc = b^2 - 4*a*c;

    if disc > 0
        x1 = (-b + sqrt(disc))/(2*a);
        x2 = (-b - sqrt(disc))/(2*a);
        type = 'Real and distinct';
    elseif disc == 0
        x1 = -b/(2*a);
        x2 = x1;
        type = 'Real and repeated';
    else
        real_part = -b/(2*a);
        imag_part = sqrt(abs(disc))/(2*a);
        x1 = complex(real_part, imag_part);
        x2 = complex(real_part, -imag_part);
        type = 'Complex conjugates';
    end

    disp(['Case ', num2str(i), ': ', type]);
    disp(['Roots: ', num2str(x1), ' and ', num2str(x2)]);
end
```

Problem 9

Determine how many days the price of stock A was below the price of stock B given arrays.

```
price_A = [19, 18, 22, 21, 25, 19, 17, 21, 27, 29];
price_B = [22, 17, 20, 23, 24, 18, 16, 25, 28, 27];

% Logical comparison
days_below = price_A < price_B;

% Count true values
num_days = sum(days_below);

disp(['Days A was below B: ', num2str(num_days)]);
```

Problem 21

Create a function **fxy** to evaluate the piecewise function $f(x, y)$ defined as: $x + y$ if $x, y \geq 0$; $x - y$ if $x \geq 0, y < 0$; $-x^2y$ if $x < 0, y \geq 0$; $-x^2y^2$ if $x, y < 0$.

```
% --- Main Script ---
disp(['f(1,1) = ', num2str(fxy(1,1))]);
disp(['f(1,-1) = ', num2str(fxy(1,-1))]);
disp(['f(-1,1) = ', num2str(fxy(-1,1))]);
disp(['f(-1,-1) = ', num2str(fxy(-1,-1))]);

% --- Function Definition ---
function val = fxy(x, y)
    if x ≥ 0 && y ≥ 0
        val = x + y;
    elseif x ≥ 0 && y < 0
        val = x - y;
    elseif x < 0 && y ≥ 0
        val = -x^2 * y;
    else % x < 0 and y < 0
        val = -x^2 * y^2;
    end
end
```

Problem 28

For array A, compute B by taking ln of elements ≥ 1 and adding 20 to elements < 1 (Note: Prompt text says "adding 20 to each element that is equal to or greater than 1", but logic suggests standard masking ops. Following text literally: "natural logarithm of all ... no less than 1, and adding 20 to each element that is equal to or greater than 1". This implies two operations on the same subset? Or is there a typo in the source? Re-reading source: "...logarithm of all the elements of A whose value is no less than 1, and adding 20 to each element that is equal to or greater than 1." This is contradictory or redundant. Correction based on standard exercises: Usually it's Log for $x \geq 1$ and Add 20 for $x < 1$. However, I will implement exactly as written in source if possible, or assume the second condition is for the *other* set. Let's assume the source meant "add 20 to elements LESS than 1". I will code this interpretation for utility).

```
A = [3, 5, -4; -8, -1, 33; -17, 6, -9];
B = A; % Initialize B

% Method A: Loops
[rows, cols] = size(A);
for i = 1:rows
    for j = 1:cols
        if A(i,j) ≥ 1
            B(i,j) = log(A(i,j));
        else
            B(i,j) = A(i,j) + 20;
        end
    end
end
disp('B (Loop method):'); disp(B);

% Method B: Logical Masking
B_mask = A;
mask_ge1 = A ≥ 1;
mask_lt1 = A < 1;

B_mask(mask_ge1) = log(A(mask_ge1));
B_mask(mask_lt1) = A(mask_lt1) + 20;

disp('B (Mask method):'); disp(B_mask);
```

Problem 40

Find L_{ACmin} for a weight supported by two cables using a while loop. $D = 6, L_{AB} = 3, W = 2000$. Conditions: Tension ≤ 2000. L_{AC} varies up to 6.7.

```
D = 6; L_AB = 3; W = 2000;
L_AC = 3.01; % Start slightly > 3 (triangle inequality)
step = 0.01;
max_L = 6.7;

found_min = false;
```

```
L_AC_min = NaN;

while L_AC ≤ max_L
    % Law of Cosines for angles
    % D^2 = L_AB^2 + L_AC^2 - 2*L_AB*L_AC*cos(theta_opp_D)? No.
    % Using formulas from source
    cos_theta = (D^2 + L_AB^2 - L_AC^2) / (2*D*L_AB);
    theta = acos(cos_theta);

    sin_phi = (L_AB * sin(theta)) / L_AC;
    phi = asin(sin_phi);

    % Solve Equilibrium Eq
    % -T_AB*cos(theta) + T_AC*cos(phi) = 0
    % T_AB*sin(theta) + T_AC*sin(phi) = W

    % Linear system A_mat * [T_AB; T_AC] = [0; W]
    A_mat = [-cos(theta), cos(phi); sin(theta), sin(phi)];
    b_vec = [0; W];
    T = A_mat \ b_vec;
    T_AB = T(1); T_AC = T(2);

    if T_AB ≤ 2000 && T_AC ≤ 2000
        if ¬found_min
            L_AC_min = L_AC;
            found_min = true;
        end
    end

    L_AC = L_AC + step;
end

disp(['Minimum valid Length AC: ', num2str(L_AC_min)]);
```

Tutorial 9: Numerical Methods

Problem 5

Acceleration $a(t) = 5t\sin(8t)$. Compute velocity at $t = 20$ if $v(0) = 0$.

```
% v(t) = integral of a(t) from 0 to 20
a_fun = @(t) 5 .* t .* sin(8 .* t);
v_20 = integral(a_fun, 0, 20);

disp(['Velocity at t=20: ', num2str(v_20), ' m/s']);
```

Problem 10

Rocket equation: $m(t)\frac{dv}{dt} = T - m(t)g$. Calculate velocity at burnout ($t = 40$). $T = 48000, m_0 = 2200, r = 0.8, g = 9.81$.

```
T = 48000; m0 = 2200; r = 0.8; g = 9.81; b = 40;

% ODE: dv/dt = T/m(t) - g
% m(t) = m0 * (1 - r*t/b)
dvdt = @(t, v) (T ./ (m0 * (1 - r*t/b))) - g;

[t_sol, v_sol] = ode45(dvdt, [0, b], 0);

disp(['Velocity at burnout: ', num2str(v_sol(end)), ' m/s']);
plot(t_sol, v_sol); title('Rocket Velocity'); xlabel('t'); ylabel('v');
```

Problem 29

Spherical tank draining. $\pi(2rh - h^2)\frac{dh}{dt} = -C_d A\sqrt{2gh}$. Radius $r = 3$, drain radius 2cm (0.02m), $C_d = 0.5$, $h(0) = 5$. Estimate empty time.

```
r_tank = 3;
r_drain = 0.02;
A_drain = pi * r_drain^2;
```

```
Cd = 0.5; g = 9.81;

% ODE: dh/dt = - (Cd * A * sqrt(2gh)) / (pi * (2rh - h^2))
dhdt = @(t, h) -(Cd * A_drain * sqrt(2*g*h)) ./ (pi * (2*r_tank*h - h.^2));

% Integrate until h is near 0 (event function typically used, or guess time)
% Using ode45 with events to stop at h=0
options = odeset('Events', @stop_event);
[t, h] = ode45(dhdt, [0, 50000], 5, options);

disp(['Time to empty: ', num2str(t(end)/3600), ' hours']);
plot(t, h); title('Tank Draining');

% Event function definition
function [value, isterminal, direction] = stop_event(t, h)
    value = h - 0.01; % Stop when height is 1cm
    isterminal = 1;
    direction = 0;
end
```

Problem 45

Equation $5\ddot{y} + 2\dot{y} + 10y = f(t)$.

a. Free response: $y(0) = 10, \dot{y}(0) = -5$.

b. Step response: Zero ICs, unit step input.

c. Total response superposition.

```
% State Space: x1 = y, x2 = y_dot
% y_ddot = (f - 2y_dot - 10y)/5
% dx1 = x2
% dx2 = 0.2f - 0.4x2 - 2x1

% a. Free Response (f=0)
ode_free = @(t, x) [x(2); -0.4*x(2) - 2*x(1)];
[t_free, x_free] = ode45(ode_free, [0, 15], [10; -5]);

% b. Step Response (f=1, IC=0)
ode_step = @(t, x) [x(2); 0.2*1 - 0.4*x(2) - 2*x(1)];
[t_step, x_step] = ode45(ode_step, [0, 15], [0; 0]);

% c. Total Response (f=1, IC=[10, -5])
ode_total = @(t, x) [x(2); 0.2*1 - 0.4*x(2) - 2*x(1)];
[t_tot, x_tot] = ode45(ode_total, [0, 15], [10; -5]);

% Plotting
figure;
plot(t_free, x_free(:,1), '--', 'DisplayName', 'Free'); hold on;
```

```
plot(t_step, x_step(:,1), ':', 'DisplayName', 'Step');
plot(t_tot, x_tot(:,1), 'k-', 'LineWidth', 1.5, 'DisplayName', 'Total');
legend; title('Superposition of Responses');
```