Figures & Discussions

Problem A

Problem A.1:

Characteristic Roots:

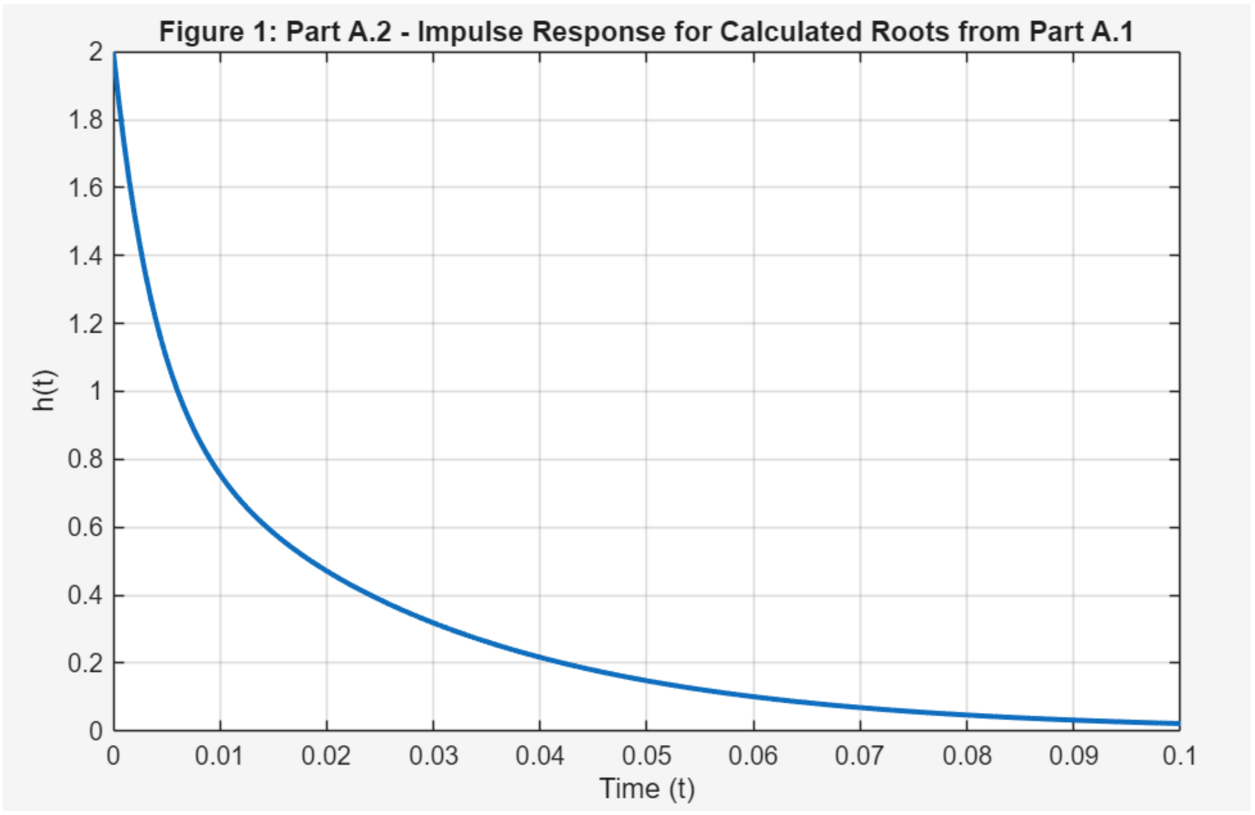
```
-261.8034
 -38.1966
```

Characteristic polynomial coefficients:

```
  1.0e+04 *

   0.0001    0.0300    1.0000
```

Problem A.2:

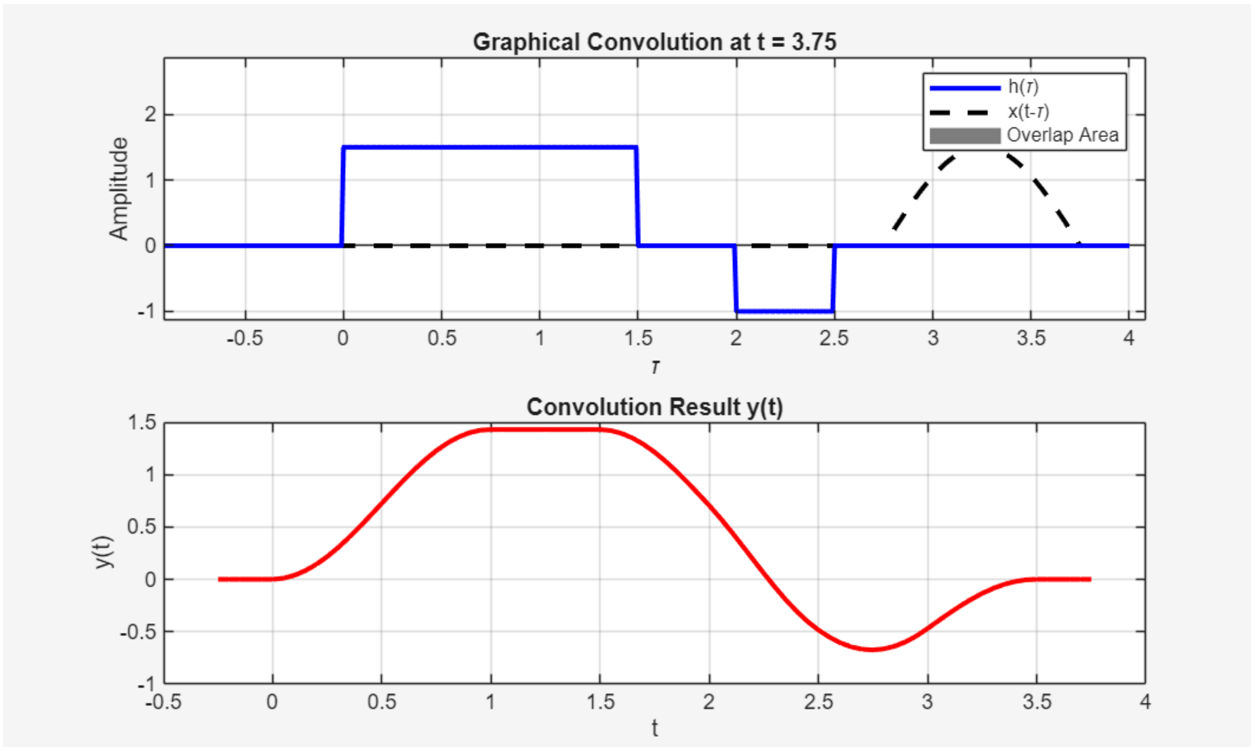


Problem A.3:

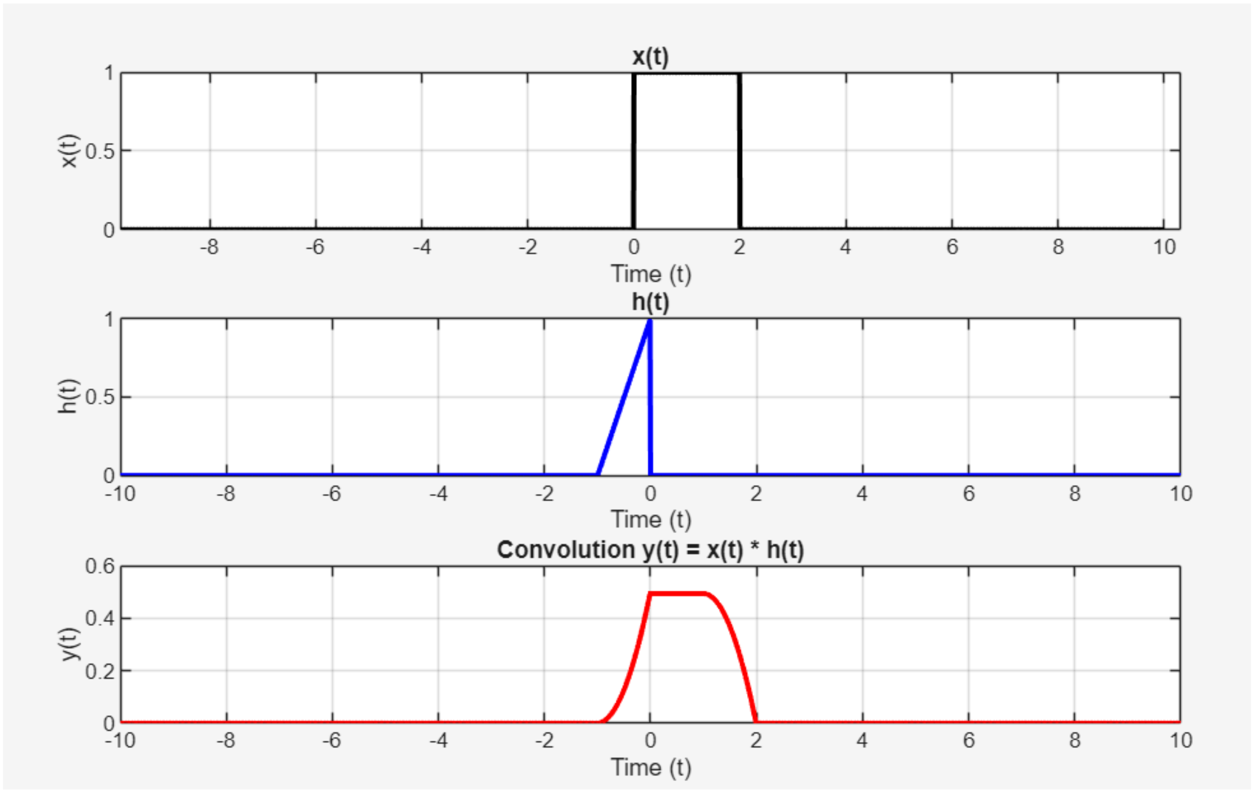
```
lambda =

-261.8034
 -38.1966
```

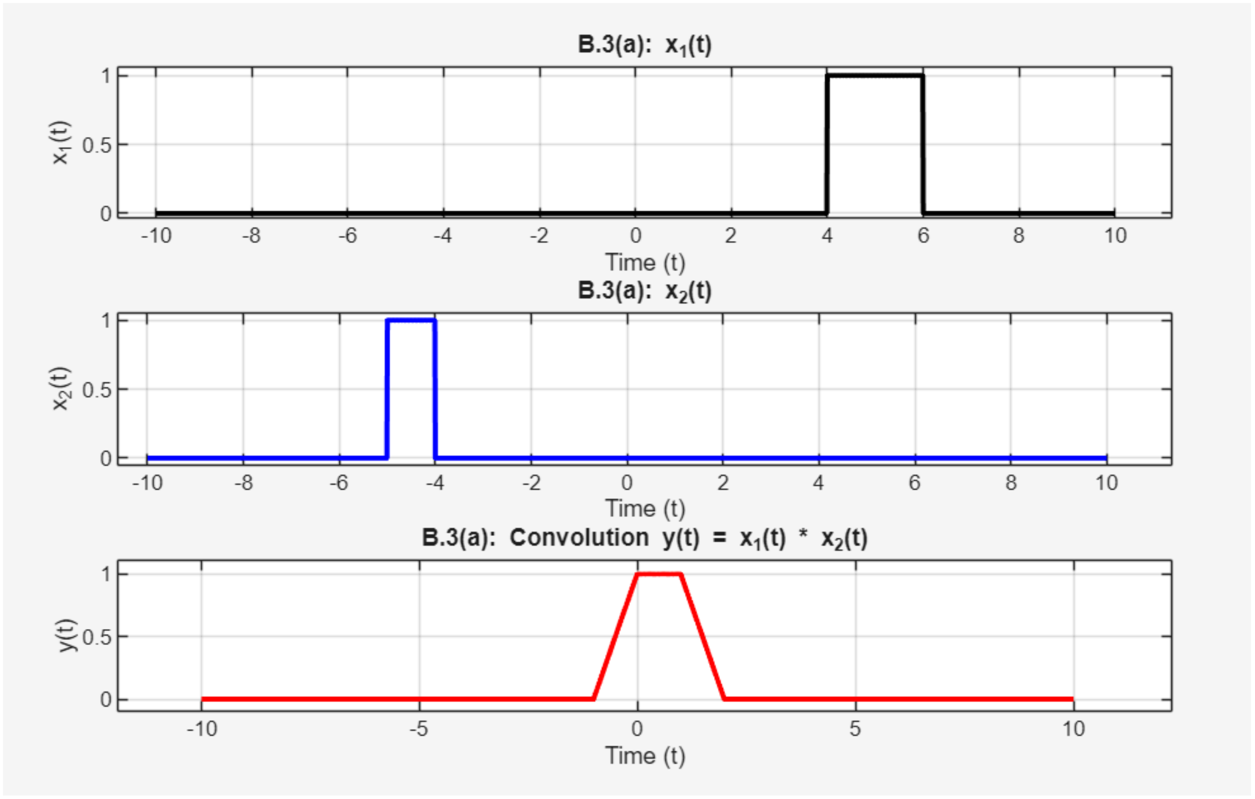
Problem B.1:



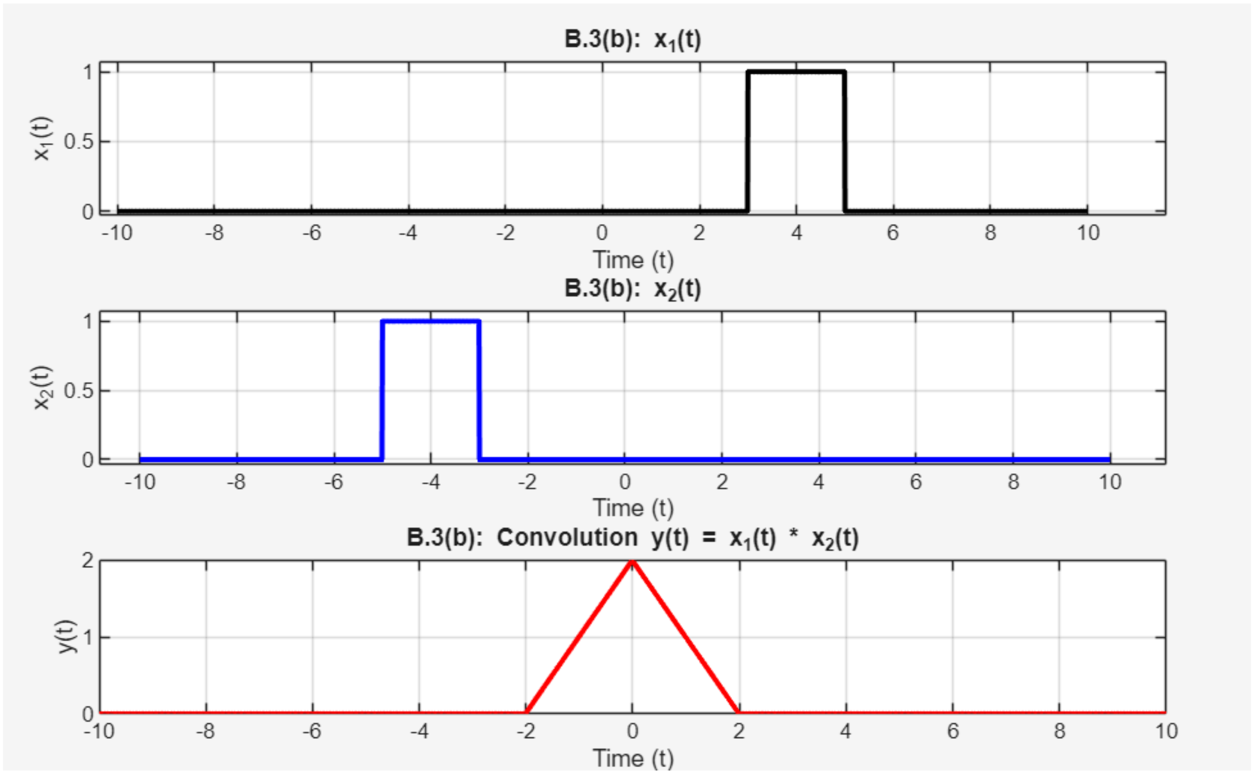
Problem B.2:


x(t)
h(t)
Convolution y(t) = x(t) * h(t)
Time (t)
x(t)
h(t)
y(t)

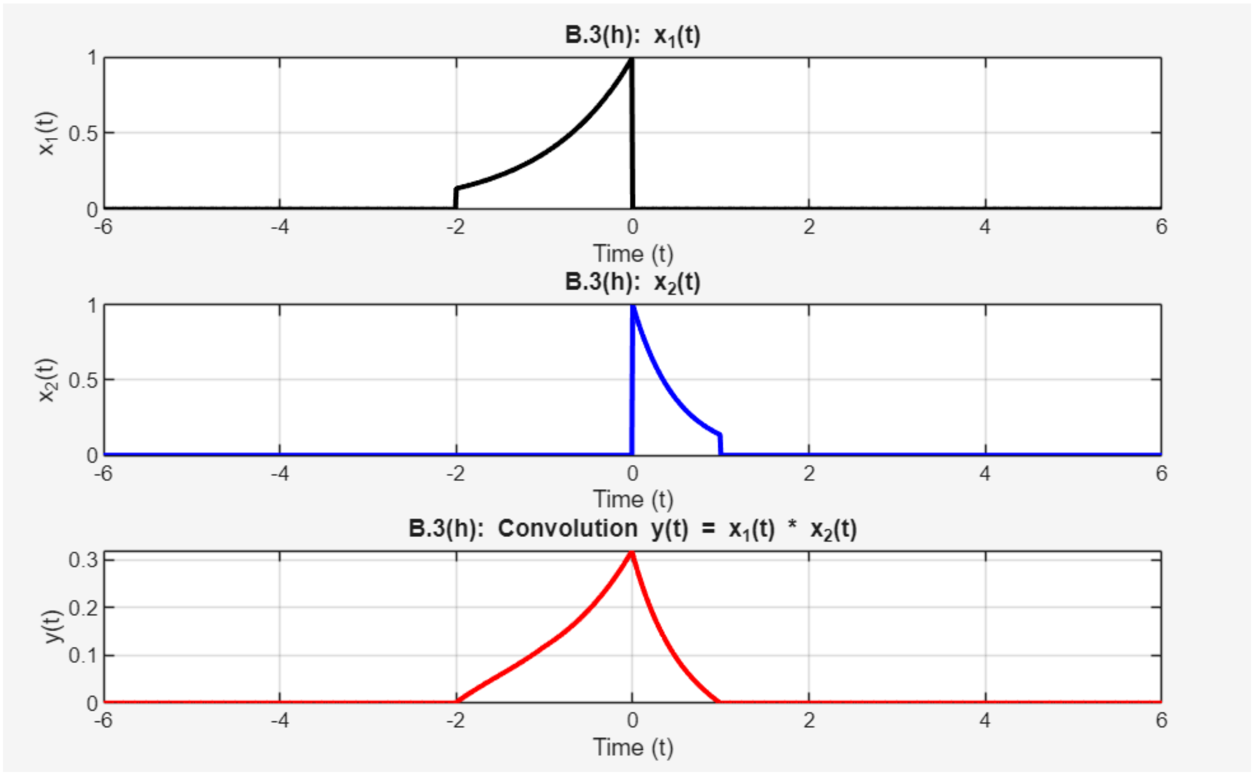
Problem B.3.1:


B.3(a): x1(t)
B.3(a): x2(t)
B.3(a): Convolution y(t) = x1(t) * x2(t)
Time (t)
x1(t)
x2(t)
y(t)

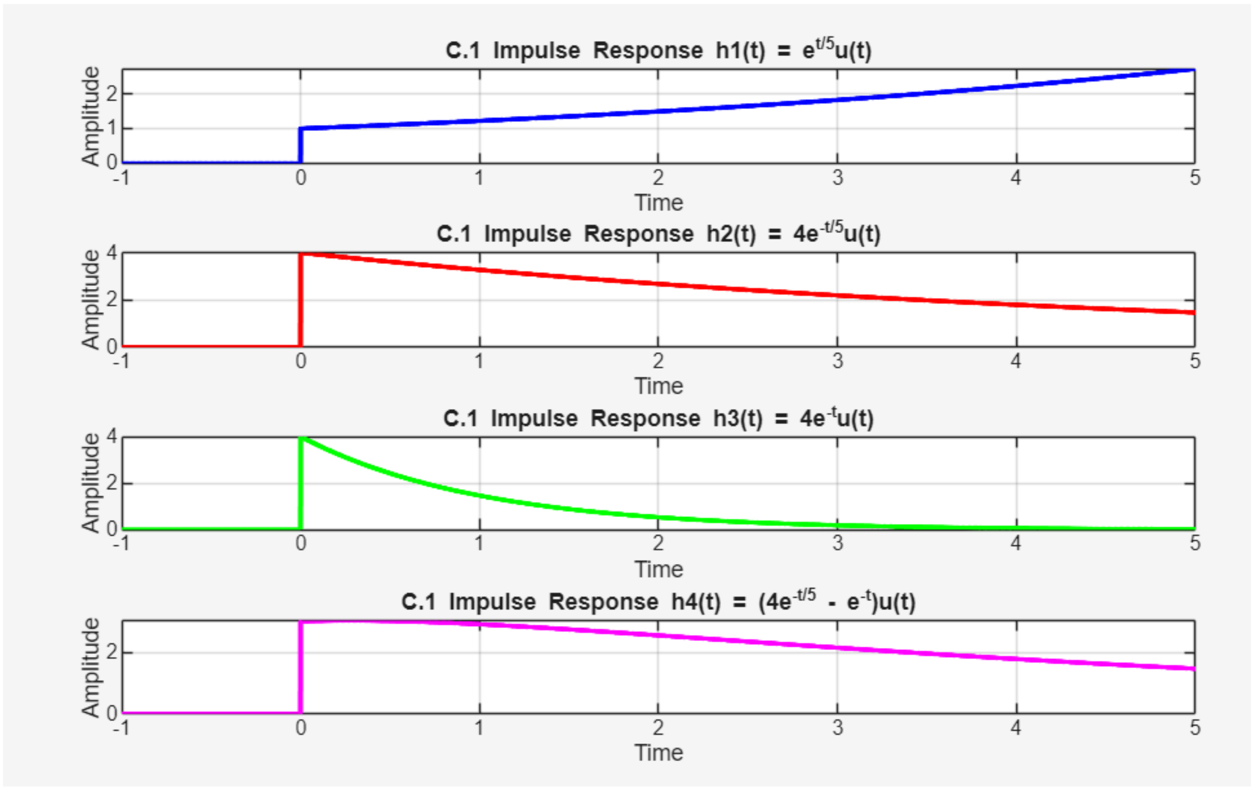
Problem B.3.2:



Problem B.3.3:



Problem C.1:



Problem C.2:

C.2 Eigenvalues (Poles) for each system:

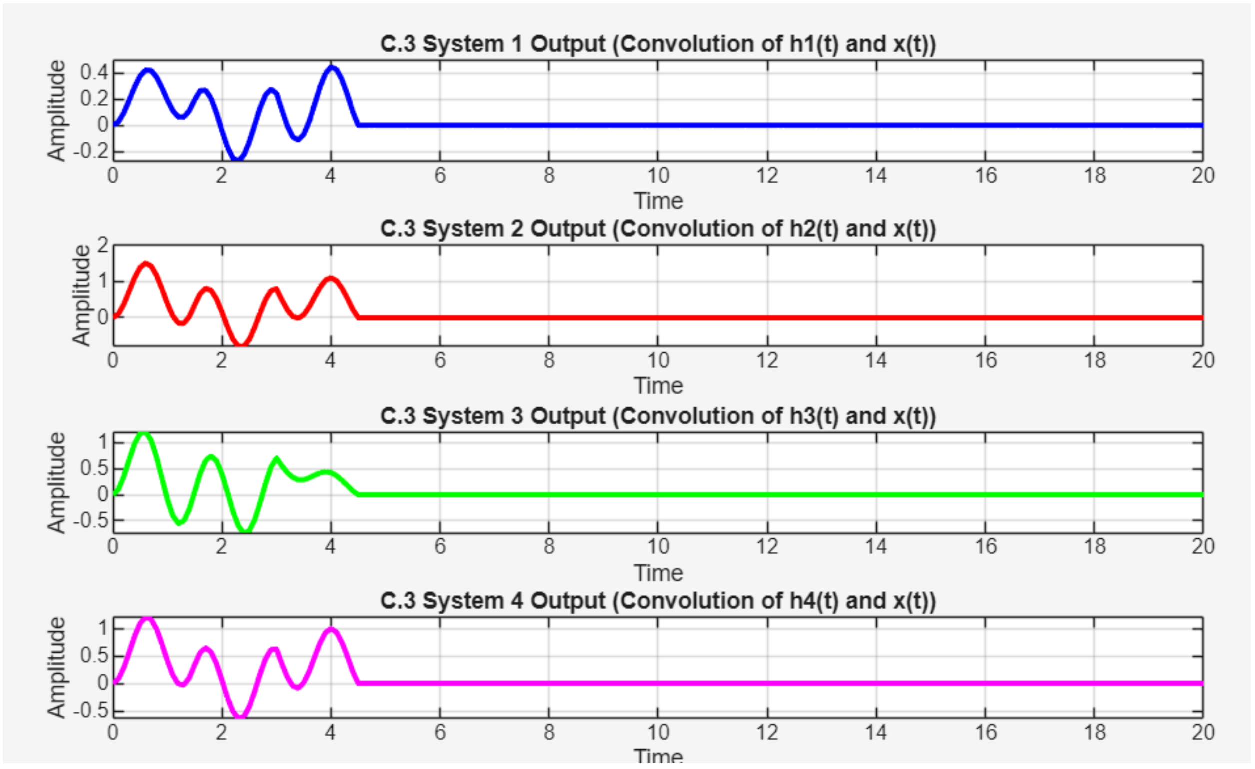
Eigenvalues for h1 (e^{t/5}): 0.2

Eigenvalues for h2 (4e^{-t/5}): -0.2

Eigenvalues for h3 (4e^{-t}): -1

Eigenvalues for h4 (4e^{-t/5} - e^{-t}): -1   -0.2

Problem C.3:

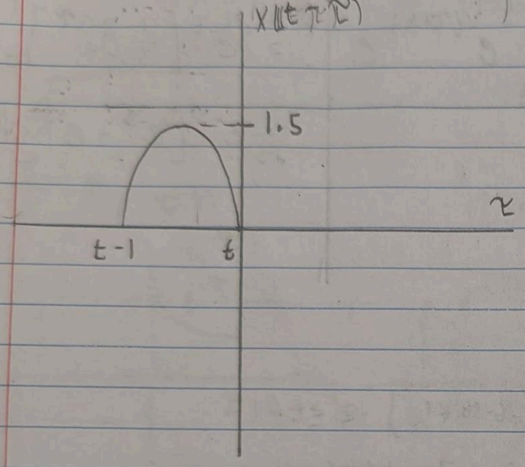


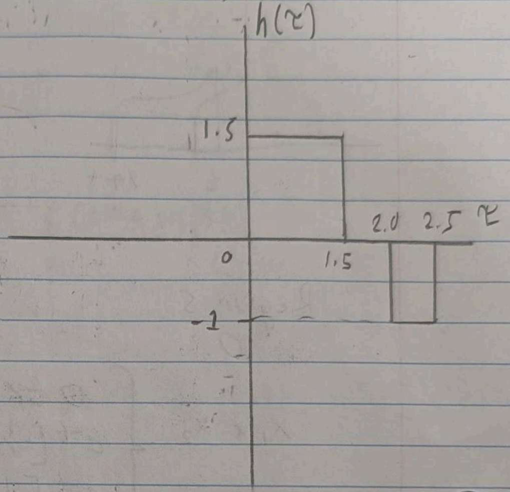
C.3 System 1 Output (Convolution of h1(t) and x(t))
Amplitude
Time
C.3 System 2 Output (Convolution of h2(t) and x(t))
Amplitude
Time
C.3 System 3 Output (Convolution of h3(t) and x(t))
Amplitude
Time
C.3 System 4 Output (Convolution of h4(t) and x(t))
Amplitude
Time

Problem D.1

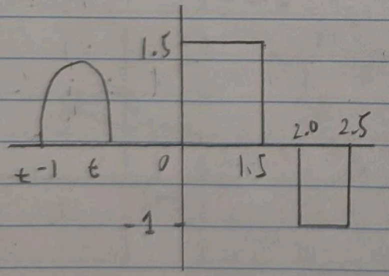
B.1) $x(t) = 1.5\sin(\pi t)(u(t) - u(t-1))$

$h(t) = 1.5(u(t) - u(t-1.5)) - u(t-2) + u(t-2.5)$



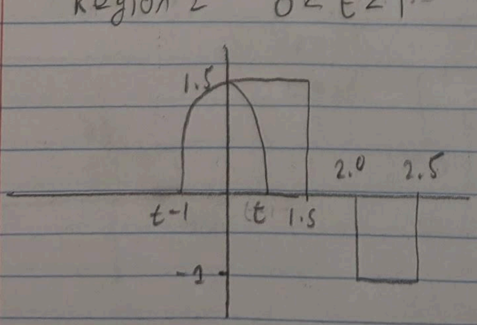


Region 1 $t<0$



$x(t) * h(t) = 0$

Region 2 $0 < t < 1$



$$x(t) * h(t) = \int_0^t x(\tau)\cdot h(\tau)\,d\tau$$
$$= \int_0^t 1.5\sin(\pi\tau)\cdot 1.5\,d\tau$$
$$= 2.25\int_0^t \sin(\pi\tau)\,d\tau$$
$$= 2.25\left(\frac{-\cos(\pi\tau)}{\pi}\right)\Big|_0^t$$
$$= \frac{2.25}{\pi}\left(-\cos(\pi t) + 1\right)$$

Region 3 $1 < t < 1.5$

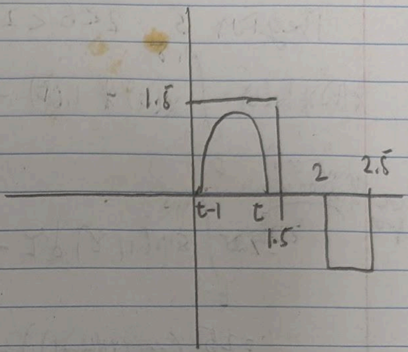
$$x(t) * h(t) = \int_{0}^{t} x(\tau)\cdot h(\tau)\, d\tau$$

$$= \int_{0}^{t} 1.5 \sin(\pi\tau) \cdot 1.5\, d\tau$$

$$= \frac{2.25}{\pi}\left(-\cos(\pi\tau)\right)\Big|_{0}^{t}$$

$$= \frac{2.25}{\pi}\left(-\overset{-(-1)}{\cos}(\pi(1)) + \overset{1}{\cos}(\pi(0))\right)$$

$$= \frac{2.25}{\pi}(2) = 1.43$$



Region 4 $1.5 < t < 2$

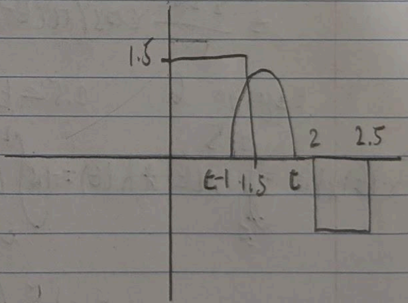
$$x(t) * h(t) = \int_{t-1}^{1.5} x(\tau)\cdot h(\tau)\, d\tau$$

$$= \int_{t-1}^{1.5} 1.5 \sin(\pi\tau) \cdot 1.5\, d\tau$$

$$= \frac{2.25}{\pi}\left(-\cos(\pi\tau)\right)\Big|_{t-1}^{1.5}$$

$$= \frac{2.25}{\pi}\left(-\overset{0}{\cos}(\pi(1.5)) + \cos(\pi(t-1))\right)$$

$$= \frac{2.25}{\pi}\left(\cos(\pi(t-1))\right)$$



Region 5 $2 < t < 2.5$

$$x(t)*h(t) = \int_{t-1}^{1.5} x(t)*h(t) - \int_{2}^{t} x(t)*h(t)$$

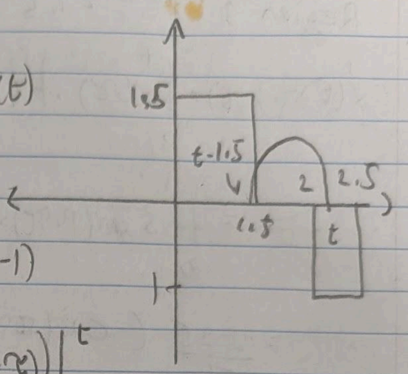
we know ans from region 4

$$= 2.25\int_{t-1}^{1.5} \sin(\pi\tau)\,d\tau - \int_{2}^{t} 1.5\sin(\pi\tau)(-1)$$

$$= \frac{2.25}{\pi}(\cos(\pi(t-1)) + \frac{1.5}{\pi}(-\cos(\pi\tau))\Big|_{2}^{t}$$

$$= \frac{2.25}{\pi}\cos(\pi(t-1)) + \frac{1.5}{\pi}(-\cos(\pi t) + \cos(\overset{1}{\pi(2)}))$$

$$= \frac{2.25}{\pi}\cos(\pi(t-1)) - \frac{1.5}{\pi}(-\cos(\pi t)+1)$$



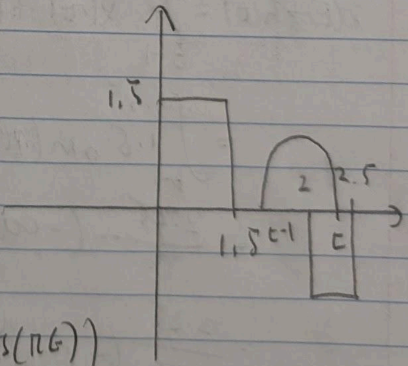
Region 6 $2.5 < t < 3$

$$x(t)*h(t) = \int_{t}^{2.5} x(t)*h(t) = -1.5\int_{t}^{2.5} (\sin(\pi\tau))\,d\tau$$

$$= \frac{-1.5}{\pi}(-\cos(\pi\tau)\Big|_{t}^{2.5}$$

$$= -\frac{1.5}{\pi}(-\cos\overset{0}{(\pi(2.5)} + \cos(\pi t))$$

$$= \frac{-1.5}{\pi}(\cos \pi t)$$



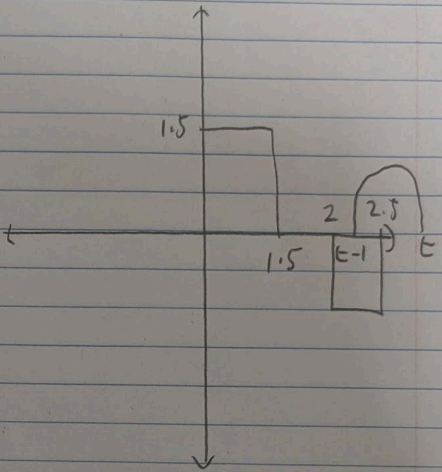
Region 7 $3 < t < 3.5$

$$x(t) * h(t) = \int_{t-1}^{2.5} x(\tau) \cdot h(\tau)\, d\tau$$

$$= -1.5 \int_{t-1}^{2.5} \sin(\pi \tau)\, d\tau$$

$$= \frac{-1.5}{\pi} \left(-\cos(\pi t)\right) \Big|_{t-1}^{2.5}$$

$$= \frac{-1.5}{\pi} (\cos(\pi(t-1))$$



Region 8 $t > 3.5$

$$x(t) * h(t) = 0$$

from the convolution we can say the graphs on matlab are correct

Problem D.2
From doing these convolutions we can say that the width/duration of each convolution can be found by adding the duration of both functions for example, the convolution for **B.1** lasted 3.5 seconds which is the duration of x(t) (1 second) and h(t) (2.5 seconds) added together

Program

```
% --- PART A: Circuit Analysis ---
% A.1 - Calculate characteristic equation coefficients and roots
R = 10e3; % Resistor values (10 kOhm). Using scalar R here for the simple example.
C1 = 1e-6; % Capacitor C1 (1 µF)
C2 = 1e-6; % Capacitor C2 (1 µF)
A = [1, (3/(R*C2)), (1/(R^2*C1*C2))]; % Coefficients vector [a2 a1 a0]
lambda = roots(A); % lambda is a column vector of the two roots
% Display the characteristic roots (eigenvalues) to the command window
disp('Characteristic Roots:');
disp(lambda);
p = poly(lambda); % poly returns the polynomial coefficients given roots
disp('Characteristic polynomial coefficients:');
disp(p)
% A.2 - Impulse response plot using the calculated roots
t = 0:0.0005:0.1; % Time vector from 0 to 0.1 seconds in steps of 0.0005 s
% response as sum of exponentials with exponents equal to the real parts of roots.
impulse_response = exp(real(lambda(1))*t) + exp(real(lambda(2))*t);
% Plot the impulse response
figure(1);
plot(t, impulse_response, 'LineWidth', 2); % LineWidth makes the plot easier to see
title('Figure 1: Part A.2 - Impulse Response for Calculated Roots from Part A.1');
xlabel('Time (t)');
ylabel('h(t)');
grid on;
% --- PART B: Convolution Operations ---
% B.1 - Graphical Convolution Visualization
% Define unit step function using anonymous function syntax
u = @(t) t >= 0; % Returns logical 1 for t >= 0 and 0 otherwise
% Define x(t) and h(t) as anonymous functions that use the unit step to window signals
x_t = @(t) 1.5 * sin(pi*t) .* (u(t) - u(t-1)); % 1.5*sin(pi t) between t=0 and t=1
h_t = @(t) 1.5 * (u(t)-u(t-1.5)) - u(t-2) + u(t-2.5);
% h(t) composed of a 1.5 amplitude window on [0,1.5) and step changes at 2 and 2.5
tau = -1:0.01:4; % tau spans -1 to 4 with 0.01 resolution
t_vec = -0.25:0.05:3.75; % times at which we compute the convolution result
% Preallocate result vector for efficiency
y = zeros(size(t_vec));
% Loop over each desired output time and numerically integrate (graphical illustration)
for i = 1:length(t_vec)
  ti = t_vec(i); % current time instant to evaluate y(t)

  % Evaluate h(tau) and shifted x(t - tau) over the tau vector
```

```
    h_tau   = h_t(tau);            % h(τ)
    x_shift = x_t(ti - tau);       % x(t - τ)

    % Compute integrand h(τ) * x(t - τ)
    integrand = h_tau .* x_shift;

    % Numerical integration using trapezoidal rule to approximate convolution integral
    y(i) = trapz(tau, integrand); % trapz approximates integral over tau
    % Plot h(τ), x(t-τ), and the overlapping (integrand) area for visualization
    figure(2); clf; % Clear figure 2 so animation frames update
    % Subplot 1: show the two functions and shaded overlap area
    subplot(2,1,1);
    area(tau, integrand, 'FaceColor', [0.5 0.5 0.5], 'EdgeColor', 'none');
    hold on;
    plot(tau, x_shift, 'k--', 'LineWidth', 2);  % plot x(t-τ)
    plot(tau, h_tau,   'b-',  'LineWidth', 2);  % plot h(τ)
    title(['Graphical Convolution at t = ', num2str(t_vec(i))]);
    xlabel('\tau'); ylabel('Amplitude');
    legend('Overlap Area', 'x(t-\tau)', 'h(\tau)');
    axis([-1 4 -2 2]); % set axis limits
    grid on;
    % Subplot 2: show accumulated convolution
    subplot(2,1,2);
    plot(t_vec(1:i), y(1:i), 'r-', 'LineWidth', 2);
    title('Convolution Result y(t)');
    xlabel('t'); ylabel('y(t)');
    grid on;
    pause(0.1); % Pause briefly to visualize
end
% B.2 - Numerical Convolution (using conv)
% Redefine x(t) and h(t)
x_t = @(t) (u(t) - u(t-2));         % rectangular pulse from 0 to 2
h_t = @(t) (t+1) .* (u(t+1) - u(t));% (t+1) on the interval [-1,0)
% Time vector for sampling the two functions
t = -10:0.01:10; % large enough range to contain signals
x_values = x_t(t);   % sample x(t) over t
h_values = h_t(t);   % sample h(t) over t
% Perform discrete convolution and scale by time step
y_t = conv(x_values, h_values, 'same') * (t(2)-t(1)); % 'same' gives same length as t
% Plot x(t), h(t), and the convolution result y(t)
figure(3);
subplot(3,1,1);
plot(t, x_values, 'k', 'LineWidth', 2);
title('x(t)');
```

```
xlabel('Time (t)'); ylabel('x(t)');
grid on;
subplot(3,1,2);
plot(t, h_values, 'b', 'LineWidth', 2);
title('h(t)');
xlabel('Time (t)'); ylabel('h(t)');
grid on;
subplot(3,1,3);
plot(t, y_t, 'r', 'LineWidth', 2);
title('Convolution y(t) = x(t) * h(t)');
xlabel('Time (t)'); ylabel('y(t)');
grid on;
% B.3(a) - Convolution of two rectangular pulses (shifted)
x1_t = @(t) (u(t-4) - u(t-6)); % pulse active on [4,6)
x2_t = @(t) (u(t+5) - u(t+4)); % pulse active on [-5,-4)
% Reuse time vector
t = -10:0.01:10;
x1_values = x1_t(t);
x2_values = x2_t(t);
% Convolve and scale by time step
y_t = conv(x1_values, x2_values, 'same') * (t(2)-t(1));
% Plot results for B.3(a)
figure(4);
subplot(3,1,1);
plot(t, x1_values, 'k', 'LineWidth', 2);
title('B.3(a): x_1(t)');
xlabel('Time (t)'); ylabel('x_1(t)');
grid on;
subplot(3,1,2);
plot(t, x2_values, 'b', 'LineWidth', 2);
title('B.3(a): x_2(t)');
xlabel('Time (t)'); ylabel('x_2(t)');
grid on;
subplot(3,1,3);
plot(t, y_t, 'r', 'LineWidth', 2);
title('B.3(a): Convolution y(t) = x_1(t) * x_2(t)');
xlabel('Time (t)'); ylabel('y(t)');
grid on;
% B.3(b) - Another pair of rectangular pulses
x1_t = @(t) (u(t-3) - u(t-5)); % pulse on [3,5)
x2_t = @(t) (u(t+5) - u(t+3)); % pulse on [-5,-3)
t = -10:0.01:10;
x1_values = x1_t(t);
x2_values = x2_t(t);
```

```
y_t = conv(x1_values, x2_values, 'same') * (t(2)-t(1));
% Plot results for B.3(b)
figure(5);
subplot(3,1,1);
plot(t, x1_values, 'k', 'LineWidth', 2);
title('B.3(b): x_1(t)');
xlabel('Time (t)'); ylabel('x_1(t)');
grid on;
subplot(3,1,2);
plot(t, x2_values, 'b', 'LineWidth', 2);
title('B.3(b): x_2(t)');
xlabel('Time (t)'); ylabel('x_2(t)');
grid on;
subplot(3,1,3);
plot(t, y_t, 'r', 'LineWidth', 2);
title('B.3(b): Convolution y(t) = x_1(t) * x_2(t)');
xlabel('Time (t)'); ylabel('y(t)');
grid on;
% B.3(h) - Exponential signals convolved
x1_t = @(t) exp(t)   .* (u(t+2) - u(t));   % exp(t) on [-2,0)
x2_t = @(t) exp(-2*t).* (u(t)   - u(t-1)); % exp(-2t) on [0,1)
% Time vector chosen to capture the significant energy of exponentials
t = -6:0.01:6;
x1_values = x1_t(t);
x2_values = x2_t(t);
% Convolution and scaling
y_t = conv(x1_values, x2_values, 'same') * (t(2)-t(1));
% Plot B.3(h)
figure(6);
subplot(3,1,1);
plot(t, x1_values, 'k', 'LineWidth', 2);
title('B.3(h): x_1(t)');
xlabel('Time (t)'); ylabel('x_1(t)');
grid on;
subplot(3,1,2);
plot(t, x2_values, 'b', 'LineWidth', 2);
title('B.3(h): x_2(t)');
xlabel('Time (t)'); ylabel('x_2(t)');
grid on;
subplot(3,1,3);
plot(t, y_t, 'r', 'LineWidth', 2);
title('B.3(h): Convolution y(t) = x_1(t) * x_2(t)');
xlabel('Time (t)'); ylabel('y(t)');
grid on;
```

```
% --- PART C: System Analysis ---
% C.1 - Plot Impulse Responses for four systems
t = -1:0.001:5; % time vector for plotting impulse responses
u = @(t) (t >= 0); % unit step function reused
% Define each system impulse response as an anonymous function
h1 = @(t) exp(t/5) .* u(t);
h2 = @(t) 4 * exp(-t/5) .* u(t);
h3 = @(t) 4 * exp(-t) .* u(t);
h4 = @(t) (4 * exp(-t/5) - exp(-t)) .* u(t);
% Plot the four impulse responses in a 4-row subplot figure
figure(7);
subplot(4,1,1);
plot(t, h1(t), 'b', 'LineWidth', 2);
title('C.1 Impulse Response h1(t) = e^{t/5}u(t)');
xlabel('Time');
ylabel('Amplitude');
grid on;
subplot(4,1,2);
plot(t, h2(t), 'r', 'LineWidth', 2);
title('C.1 Impulse Response h2(t) = 4e^{-t/5}u(t)');
xlabel('Time');
ylabel('Amplitude');
grid on;
subplot(4,1,3);
plot(t, h3(t), 'g', 'LineWidth', 2);
title('C.1 Impulse Response h3(t) = 4e^{-t}u(t)');
xlabel('Time');
ylabel('Amplitude');
grid on;
subplot(4,1,4);
plot(t, h4(t), 'm', 'LineWidth', 2);
title('C.1 Impulse Response h4(t) = (4e^{-t/5} - e^{-t})u(t)');
xlabel('Time');
ylabel('Amplitude');
grid on;
% C.2 - Find Eigenvalues (Poles) via Laplace transforms
syms s t
% Compute Laplace transforms (H(s)) for each impulse response
H1 = laplace(exp(t/5) * heaviside(t), t, s);   % symbolic H1(s)
H2 = laplace(4 * exp(-t/5) * heaviside(t), t, s); % H2(s)
H3 = laplace(4 * exp(-t) * heaviside(t), t, s);   % H3(s)
H4 = laplace((4 * exp(-t/5) - exp(-t)) * heaviside(t), t, s); % H4(s)
eigenvalues_1 = solve(s - (1/5), s);
eigenvalues_2 = solve(s + (1/5), s);
```

```
eigenvalues_3 = solve(s + 1, s);
eigenvalues_4 = solve((s + 1/5)*(s + 1), s);
% Convert symbolic solutions to numeric doubles for display
eigenvalues_1 = double(eigenvalues_1);
eigenvalues_2 = double(eigenvalues_2);
eigenvalues_3 = double(eigenvalues_3);
eigenvalues_4 = double(eigenvalues_4);
% Display the computed poles for each system
disp(' ');
disp('C.2 Eigenvalues (Poles) for each system:');
disp(['Eigenvalues for h1 (e^{t/5}): ' num2str(eigenvalues_1')]);
disp(['Eigenvalues for h2 (4e^{-t/5}): ' num2str(eigenvalues_2')]);
disp(['Eigenvalues for h3 (4e^{-t}): ' num2str(eigenvalues_3')]);
disp(['Eigenvalues for h4 (4e^{-t/5} - e^{-t}): ' num2str(eigenvalues_4')]);
% C.3 - Numerical convolution of each system with an input x(t)
% Define time vector for output and the input signal x(t)
t = 0:0.1:20; % time vector for plotting results
u = @(t) 1.0*(t>=0); % unit step used in input definition
x = @(t) (u(t) - u(t - 3)) .* sin(5 * t); % input: sin(5t) windowed on [0,3)
% shorten impulse responses to [0, 1.5)
h1_conv = @(t) exp(t/5) .* (u(t) - u(t-1.5));
h2_conv = @(t) 4 * exp(-t/5) .* (u(t) - u(t-1.5));
h3_conv = @(t) 4 * exp(-t) .* (u(t) - u(t-1.5));
h4_conv = @(t) (4 * exp(-t/5) - exp(-t)) .* (u(t) - u(t-1.5));
dtau = 0.005;
tau = -1:dtau:20;
tvec = 0:0.1:20; % output time points for which y(t) is computed
y1 = zeros(1,length(tvec)); % assigned output vectors
y2 = zeros(1,length(tvec));
y3 = zeros(1,length(tvec));
y4 = zeros(1,length(tvec));
% Loop to compute convolution for each system numerically
for ti = 1:length(tvec)
  current_t = tvec(ti);

  % Evaluate integrand x(t - tau) .* h(tau) for each system
  xh1 = x(current_t - tau).*h1_conv(tau);
  xh2 = x(current_t - tau).*h2_conv(tau);
  xh3 = x(current_t - tau).*h3_conv(tau);
  xh4 = x(current_t - tau).*h4_conv(tau);

  % Integrate over tau using trapezoidal rule
  y1(ti) = trapz(tau, xh1);
  y2(ti) = trapz(tau, xh2);
  y3(ti) = trapz(tau, xh3);
  y4(ti) = trapz(tau, xh4);
end
% Plot the outputs y(t) for each system in a 4-row subplot
figure(8);
subplot(4,1,1);
plot(tvec, y1, 'b', 'LineWidth', 2);
title('C.3 System 1 Output (Convolution of h1(t) and x(t))');
xlabel('Time'); ylabel('Amplitude'); grid on;
subplot(4,1,2);
plot(tvec, y2, 'r', 'LineWidth', 2);
title('C.3 System 2 Output (Convolution of h2(t) and x(t))');
xlabel('Time'); ylabel('Amplitude'); grid on;
subplot(4,1,3);
plot(tvec, y3, 'g', 'LineWidth', 2);
title('C.3 System 3 Output (Convolution of h3(t) and x(t))');
xlabel('Time'); ylabel('Amplitude'); grid on;
subplot(4,1,4);
plot(tvec, y4, 'm', 'LineWidth', 2);
title('C.3 System 4 Output (Convolution of h4(t) and x(t))');
xlabel('Time'); ylabel('Amplitude'); grid on;
% A.3 - function to compute roots for a generalized R array
function y = my_roots(R, C1, C2)
% Compute the coefficients of the characteristic equation:
A = [1, (1/R(1) + 1/R(2) + 1/R(3))/C2, (1/(R(1)*R(2)*C1*C2))];
% Compute the roots of the quadratic polynomial and return them
y = roots(A);
end
```