ELE532 Lab 2 Nini Yang

A.1:

**Code:**

% CH2MP1.m : chapter 2, p1

% 1

% script m file determines roots of op amp

% set component values

R = [1e4,1e4,1e4];

C = [1e-6, 1e-6];

% det coeffs for characteristic eqn

A = [1, (1/R(1)+1/R(2)+1/R(3))/C(2), 1/(R(1)\*R(2)\*C(1)\*C(2))];

% det roots:

lamba = roots(A);

**A screenshot of a computer code

Description automatically generatedOutput:**

A.2:

**Code:**

Code:

% CH2MP1.m : chapter 2, p1

% 1

% script m file determines roots of op amp

% set component values

t =[0:0.0005:0.1];

h = @(t) (-0.0045\*exp(-261.8034\*t)+0.0045\*exp(-38.1966\*t)).\*(t>0);

plot(t, h(t));

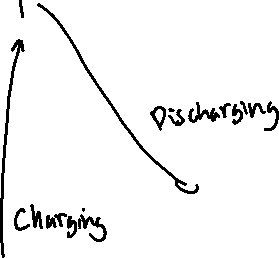
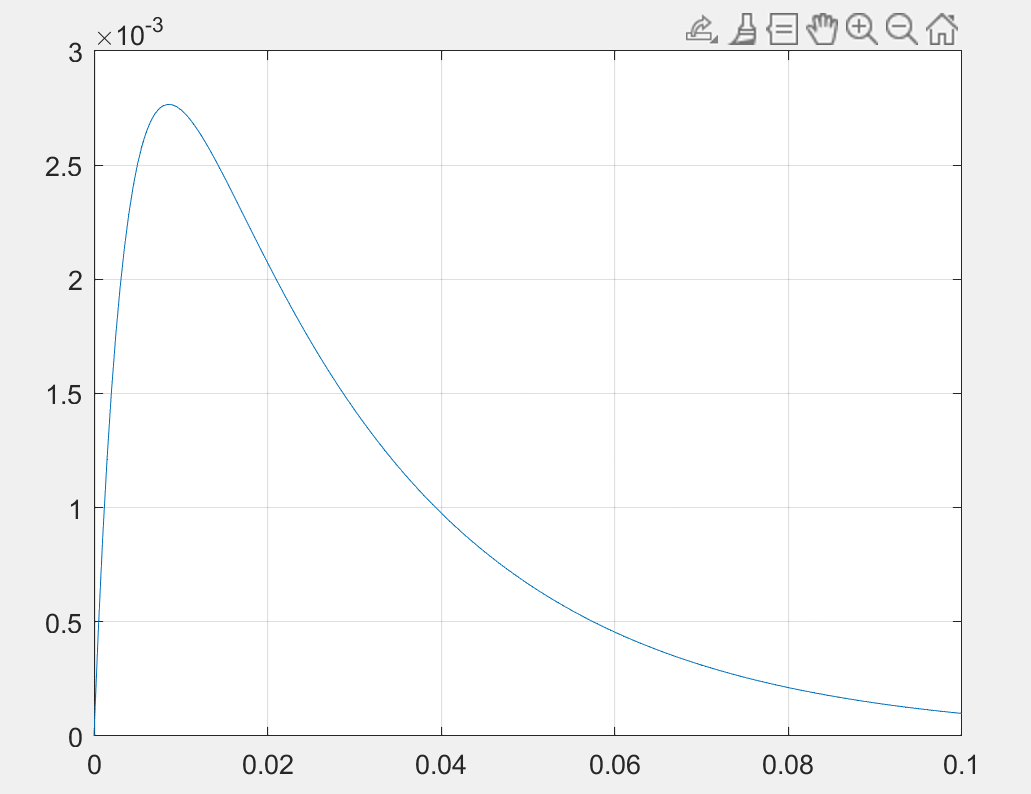
grid();

**Report:**

The graph represents the voltage output of C1 (see diagram below). When an impulse is delivered to the system, the capacitor charges up. While there is no impulse, the capacitor slowly discharges which is represented by the decreasing exponential graph.

A diagram of a circuit

Description automatically generated



A.3:

**Code:**

% CH2MP1.m : chapter 2, p1

function [lambda] = CH2MP2(R,C);

% CH2MP2.m : chpt 2, p 2

% Function find char roots of op amp circuit

% INPUTS: R = length-3 vector of resistances

% C = length-2 vector of capacitances

% OUTPUTS: lambda = characteristic roots

% Determine coefficients for characteristic equation:

A = [1, (1/R(1)+1/R(2)+1/R(3))/C(2), 1/(R(1)\*R(2)\*C(1)\*C(2))];

% Determine characteristic roots:

lambda = roots(A);

**A white screen with black text

Description automatically generatedOutput:**



B.1:

**Code:**

% lab 2 B.1

% CH2MP4.m : chpt 2, p 4

% Script M-file graphically demonstrates the convolution process.

figure(1) % Create figure window and make visible on screen

u = @(t) 1.0\*(t>=0);

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);

dtau = 0.005; tau = -1:dtau:4;

ti = 0; tvec = -.25:.1:3.75;

y = NaN\*zeros(1,length(tvec)); % Pre-allocate memory

for t = tvec

ti = ti+1; % Time index

xh = x(t-tau).\*h(tau); lxh = length(xh);

y(ti) = sum(xh.\*dtau); % Trapezoidal approximation of convolution integral

subplot(2,1,1),plot(tau,h(tau),'k-',tau,x(t-tau),'k--',t,0,'ok');

axis([tau(1) tau(end) -2.0 2.5]);

patch([tau(1:end-1);tau(1:end-1);tau(2:end);tau(2:end)],...

[zeros(1,lxh-1);xh(1:end-1);xh(2:end);zeros(1,lxh-1)],...

[.8 .8 .8],'edgecolor','none');

xlabel('\tau'); title('h(\tau) [solid], x(t-\tau) [dashed], h(\tau)x(t-\tau) [gray]');

c = get(gca,'children'); set(gca,'children',[c(2);c(3);c(4);c(1)]);

subplot(2,1,2),plot(tvec,y,'k',tvec(ti),y(ti),'ok');

xlabel('t'); ylabel('y(t) = \int h(\tau)x(t-\tau) d\tau');

axis([tau(1) tau(end) -1.0 2.0]); grid;

drawnow; pause (0.01);

end

A graph of a function

Description automatically generated**Output:**

B.2:

**Code:**

% 2

% Create figure window and make visible on screen

u = @(t) 1.0\*(t>=0);

x = @(t) u(t)-u(t-2);

h = @(t) (t+1).\*(u(t+1)-u(t));

dtau = 0.005; tau = -2:dtau:3;

ti = 0;

tvec = -1.5:.1:3;

y = NaN\*zeros(1,length(tvec)); % Pre-allocate memory

for t = tvec

ti = ti+1; % Time index

xh = x(t-tau).\*h(tau); lxh = length(xh);

y(ti) = sum(xh.\*dtau); % Trapezoidal approximation of convolution integral

subplot(2,1,1),plot(tau,h(tau),'k-',tau,x(t-tau),'k--',t,0,'ok');

axis([tau(1) tau(end) -2.0 2.5]);

patch([tau(1:end-1);tau(1:end-1);tau(2:end);tau(2:end)],...

[zeros(1,lxh-1);xh(1:end-1);xh(2:end);zeros(1,lxh-1)],...

[.8 .8 .8],'edgecolor','none');

xlabel('\tau'); title('h(\tau) [solid], x(t-\tau) [dashed], h(\tau)x(t-\tau) [gray]');

c = get(gca,'children'); set(gca,'children',[c(2);c(3);c(4);c(1)]);

subplot(2,1,2),plot(tvec,y,'k',tvec(ti),y(ti),'ok');

xlabel('t');

ylabel('y(t) = \int h(\tau)x(t-\tau) d\tau');

axis([tau(1) tau(end) -1.0 2.0]); grid;

drawnow;

end

A graph of a function

Description automatically generated

B.3:

**Code:**

% 3 ===========================================================

% Create figure window and make visible on screen

u = @(t) 1.0\*(t>=0);

x = @(t) 0.5\*(u(t-4)-u(t-6));

h = @(t) 1.0\*(u(t+5)-u(t+4));

dtau = 0.005; tau = -6:dtau:2.5;

ti = 0;

tvec = -5:.1:5;

y = NaN\*zeros(1,length(tvec)); % Pre-allocate memory

for t = tvec

ti = ti+1; % Time index

xh = x(t-tau).\*h(tau); lxh = length(xh);

y(ti) = sum(xh.\*dtau); % Trapezoidal approximation of convolution integral

subplot(2,1,1),plot(tau,h(tau),'k-',tau,x(t-tau),'k--',t,0,'ok');

axis([tau(1) tau(end) -2.0 2.5]);

patch([tau(1:end-1);tau(1:end-1);tau(2:end);tau(2:end)],...

[zeros(1,lxh-1);xh(1:end-1);xh(2:end);zeros(1,lxh-1)],...

[.8 .8 .8],'edgecolor','none');

xlabel('\tau'); title('h(\tau) [solid], x(t-\tau) [dashed], h(\tau)x(t-\tau) [gray]');

c = get(gca,'children'); set(gca,'children',[c(2);c(3);c(4);c(1)]);

subplot(2,1,2),plot(tvec,y,'k',tvec(ti),y(ti),'ok');

xlabel('t');

ylabel('y(t) = \int h(\tau)x(t-\tau) d\tau');

axis([tau(1) tau(end) -1.0 2.0]); grid;

drawnow;

end

%}

% 4 ====================================================================

%{

% Create figure window and make visible on screen

u = @(t) 1.0\*(t>=0);

x = @(t) 0.5\*(u(t-3)-u(t-5));

h = @(t) 1.0\*(u(t+5)-u(t+3));

dtau = 0.005; tau = -6:dtau:2.5;

ti = 0;

tvec = -5:.1:5;

y = NaN\*zeros(1,length(tvec)); % Pre-allocate memory

for t = tvec

ti = ti+1; % Time index

xh = x(t-tau).\*h(tau); lxh = length(xh);

y(ti) = sum(xh.\*dtau); % Trapezoidal approximation of convolution integral

subplot(2,1,1),plot(tau,h(tau),'k-',tau,x(t-tau),'k--',t,0,'ok');

axis([tau(1) tau(end) -2.0 2.5]);

patch([tau(1:end-1);tau(1:end-1);tau(2:end);tau(2:end)],...

[zeros(1,lxh-1);xh(1:end-1);xh(2:end);zeros(1,lxh-1)],...

[.8 .8 .8],'edgecolor','none');

xlabel('\tau'); title('h(\tau) [solid], x(t-\tau) [dashed], h(\tau)x(t-\tau) [gray]');

c = get(gca,'children'); set(gca,'children',[c(2);c(3);c(4);c(1)]);

subplot(2,1,2),plot(tvec,y,'k',tvec(ti),y(ti),'ok');

xlabel('t');

ylabel('y(t) = \int h(\tau)x(t-\tau) d\tau');

axis([tau(1) tau(end) -1.0 2.0]); grid;

drawnow;

end

%}

% 5 =====================================================================

%{

% Create figure window and make visible on screen

u = @(t) 1.0\*(t>=0);

x = @(t) exp(t).\*(u(t+2)-u(t));

h = @(t) exp(-2\*t).\*(u(t)-u(t-1));

dtau = 0.005; tau = -3:dtau:7.5;

ti = 0;

tvec = -5:.1:5;

y = NaN\*zeros(1,length(tvec)); % Pre-allocate memory

for t = tvec

ti = ti+1; % Time index

xh = x(t-tau).\*h(tau); lxh = length(xh);

y(ti) = sum(xh.\*dtau); % Trapezoidal approximation of convolution integral

subplot(2,1,1),plot(tau,h(tau),'k-',tau,x(t-tau),'k--',t,0,'ok');

axis([tau(1) tau(end) -2.0 2.5]);

patch([tau(1:end-1);tau(1:end-1);tau(2:end);tau(2:end)],...

[zeros(1,lxh-1);xh(1:end-1);xh(2:end);zeros(1,lxh-1)],...

[.8 .8 .8],'edgecolor','none');

xlabel('\tau'); title('h(\tau) [solid], x(t-\tau) [dashed], h(\tau)x(t-\tau) [gray]');

c = get(gca,'children'); set(gca,'children',[c(2);c(3);c(4);c(1)]);

subplot(2,1,2),plot(tvec,y,'k',tvec(ti),y(ti),'ok');

xlabel('t');

ylabel('y(t) = \int h(\tau)x(t-\tau) d\tau');

axis([tau(1) tau(end) -1.0 2.0]); grid;

drawnow;

end

%}

A graph of a graph with a line

Description automatically generatedA screenshot of a graph

Description automatically generatedA graph of a graph

Description automatically generated**Output:**

C.1:

**Code:**

t = [-1:0.001:5];

%4 functions

u = @(t) 1.0.\*(t>=0);

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);

% plotting

plot(t,h1(t));

xlabel ("t");

ylabel ("h(t)");

hold on;

plot(t,h2(t));

plot(t,h3(t));

plot(t,h4(t));

**Output:**A graph of colored lines

Description automatically generated

C.2:

**Report:**

Eigenvalues:

h1(t) = 1/5

h2(t) = -1/5

h3(t) = -1

h4(t) = -1/5 and -1

C.3:

**Code:**

%5

% Create figure window and make visible on screen

u = @(t) 1.0\*(t>=0);

x = @(t) sin(5\*t).\*(u(t)-u(t-3));

h = @(t) 4\*(exp(-t/5)-exp(-t)).\* u(t);

dtau = 0.005; tau = 0:dtau:20;

ti = 0;

tvec = 0:.1:20;

y = NaN\*zeros(1,length(tvec)); % Pre-allocate memory

for t = tvec

ti = ti+1; % Time index

xh = x(t-tau).\*h(tau); lxh = length(xh);

y(ti) = sum(xh.\*dtau); % Trapezoidal approximation of convolution integral

subplot(2,1,1),plot(tau,h(tau),'k-',tau,x(t-tau),'k--',t,0,'ok');

axis([tau(1) tau(end) -2.0 2.5]);

patch([tau(1:end-1);tau(1:end-1);tau(2:end);tau(2:end)],...

[zeros(1,lxh-1);xh(1:end-1);xh(2:end);zeros(1,lxh-1)],...

[.8 .8 .8],'edgecolor','none');

xlabel('\tau'); title('h(\tau) [solid], x(t-\tau) [dashed], h(\tau)x(t-\tau) [gray]');

c = get(gca,'children'); set(gca,'children',[c(2);c(3);c(4);c(1)]);

subplot(2,1,2),plot(tvec,y,'k',tvec(ti),y(ti),'ok');

xlabel('t');

ylabel('y(t) = \int h(\tau)x(t-\tau) d\tau');

axis([tau(1) tau(end) -1.0 2.0]); grid;

drawnow;

end

**Report:**

Relationship between outputs:

Fourth equation is a combination of equation 2 and 3: h4(t) = h3(t) - h2(t).

A graph of a function

Description automatically generated

A screenshot of a notebook

Description automatically generatedD.1:

A paper with math equations

Description automatically generated

A screenshot of a notebook

Description automatically generatedb.3

D.2:

The width of the resulting signal is a combination of the widths of the signals being convoluted, since the left and right end points of the resulting signal equal the sum of the left and right end points of the functions being combined.