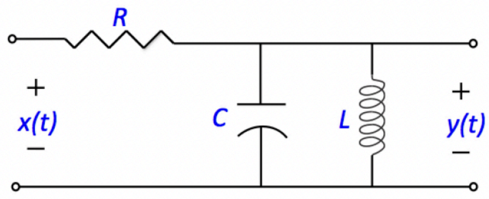
1. Following *RLC* circuit is described by the differential equation (1). Use Matlab built-in differential equation solver `dsolve()` to derive the impulse response *h(t)* for this circuit when $R=2\,\Omega$, $C=1F$, $L=0.5\ H$. Plot the impulse response $h(t)$ from a range $-10 \le t \le 30$.

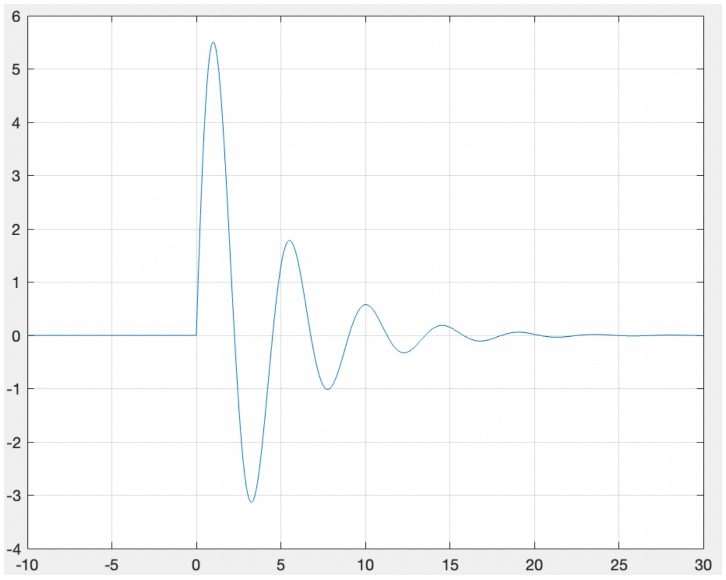


$$RC\frac{d^2y(t)}{dt^2} + \frac{dy(t)}{dt} + \frac{R}{L}y(t) = \frac{dx(t)}{dt} \qquad \ldots (1)$$

Summary:

We define the differential and call dsolve(). We plot the resulting function across the given time domain. I chose initial values of y(0)=0 and Dy(0)=10.

Results:



Code:

```
clf; % clear current figure
syms y(t) % Define the symbolic function
R = 2; % intialize variables
C = 1;
L = 0.5;
ode = R*C*diff(y, t, 2) + diff(y, t) + (R/L)*y; % define differential
Dy = diff(y, t); % define Dy
t = -10:0.001:30; % define time domain
sol = eval(dsolve(ode, [y(0) == 0, Dy(0) == 10])).*(t>=0); % eval values
plot(t, sol); % plot results
grid on; % grid on
```

2. Consider the following input signal

$$x_1(t) = \begin{cases} 5, & 0 \le t < 10 \\ 0, & \text{elsewhere} \end{cases}$$

$$x_2(t) = 2x_1(t - 10)$$

$$x_{linear\_comb}(t) = x_1(t) + 2x_1(t - 10)$$

Using the example Matlab file `simplified_convolution_runtime.m`, plot the output signals in three separate figure windows:

(a) $y_1(t) = x_1(t) * h(t)$

(b) $y_2(t) = x_2(t) * h(t)$

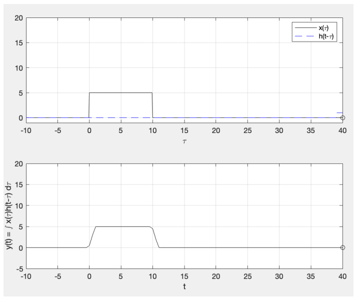
(c) $y_{linear\_comb} = x_{linear\_comb}(t) * h(t)$.

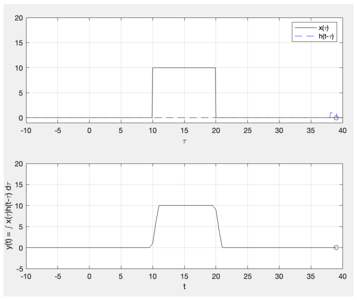
Use the ranges of 'τ' and 't' as $-10 \le \tau \le 40$ and $-10 \le t \le 40$. Also plot $y_3(t)=y_1(t)+y_2(t)$ and comment on similarity of $y_3(t)$ and $y_{linear\_comb}(t)$.

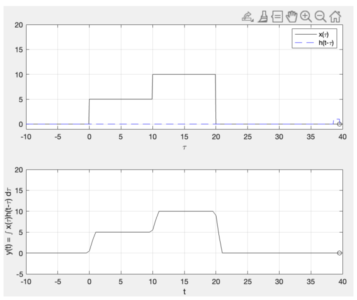
Summary:

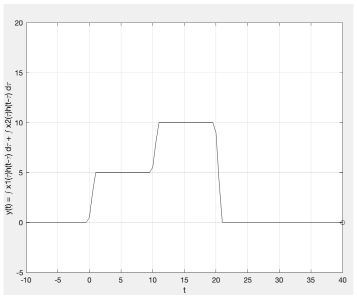
We define the x(t)'s and y(t)'s and plot across the given time domain. We use the given method to do so. Additional logic is needed to control how everything is plotted.

Results (y1, y2, ylincomb, y3):









Code:

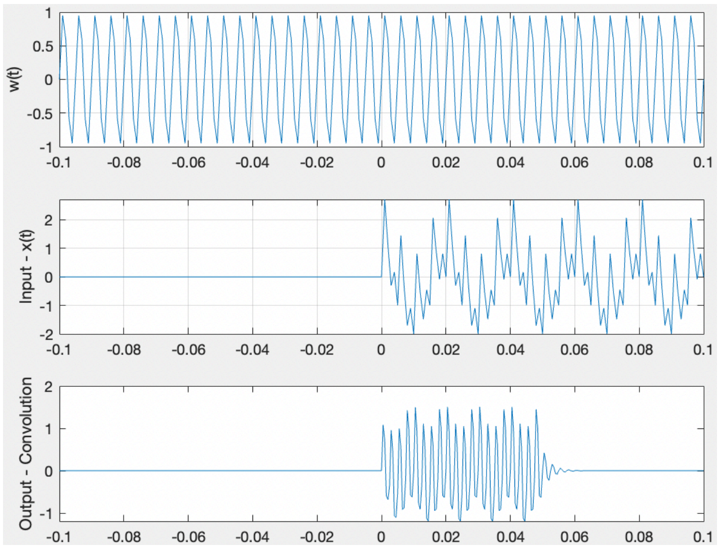
```
for j = 1:1:4 % iterate over each plot
   clearvars -except j % clear everything except j
   clf % clear figure
   x1 = @(t) 5.*(t>=0 & t<10); % define x1
   x2 = @(t) 2*x1(t-10); % define x2
   xlinearcomb = @(t) x1(t)+x2(t); % define xlincomb
   funcs = {x1, x2, xlinearcomb, x1}; % use to call proper function to x
   h = @(time) (time>=0 & time<1); % define h
   dtau = 0.1; % define dtau (tau step size)
   tau = -10:dtau:40; % define tau range
   dT=0.5; % define dT (t step size)
   t = -10:dT:40;  % time variable for the time delay and output signal
   y = NaN(1, length (t)); % allocate memory
   x = funcs{j}; % set x to proper function
   for ii=1:length(t)
       if j == 4 % if we are doing y3
           xh = h(t(ii)-tau).*x(tau) + h(t(ii)-tau).*x2(tau); % set xh to sum of convolutions
       else
           xh = h(t(ii)-tau).*x(tau); % calc convolution
       end
       y(ii)=trapz(tau,xh);   % evaluating the overlap integral of x(tau) and h(t-tau)
       if j == 4
           plot (t, y, 'k', t (ii), y(ii), 'ok');
           xlabel ('t'); ylabel ('y(t) = \int x1(\tau)h(t-\tau) d\tau + \int x2(\tau)h(t-\tau) d\tau ');
           axis ([tau(1) tau(end) -5 20]); grid;
           drawnow;
       else
           subplot (211)
           plot(tau, x(tau), 'k-', tau, h(t(ii)-tau), 'b--', t(ii), 0, 'ok');
           xlabel('\tau'); axis ([tau(1) tau(end) -1 20]); grid;
           legend('x(\tau)','h(t-\tau)');
           subplot (212)
           plot (t, y, 'k', t (ii), y(ii), 'ok');
           xlabel ('t'); ylabel ('y(t) = \int x(\tau)h(t-\tau) d\tau');
           axis ([tau(1) tau(end) -5 20]); grid;
           drawnow;
       end
   end
end
```

3. A single-tone signal $w(t) = \sin(400\pi t)$ is transmitted to an audio amplifier and speaker to produce a high-temperature warning for a silicon crystal-growing factory. A filter having impulse response $h(t) = 400e^{-200t}\cos(400\pi t)u(t)$ has been designed to reduce additive interference in the received signal. Using Matlab in-built convolution function: `conv()`, find the filter output signal $y(t)$, when the received signal is $x(t) = [\cos(100\pi t) + \sin(400\pi t) - \cos(800\pi t)]u(t)$ (signal $w(t)$ was corrupted by interference and resulted in an input signal $x(t)$). Plot the output signal, the input signal, and $w(t)$ for the range of $-0.1 \le t \le 0.1$. Comment on the effect of the filter on the signal. While solving this problem, pay attention to the time resolution/step (`dT`) you need to use.

Summary:

We define the above functions across the given time domain, plot, and call conv() to plot the convolution. The filter changes the jumpy behavior of the input to more closely resemble w(t).

Results:



Code:

```
t = -0.1:0.001:0.1;
w = sin(400*pi*t);
h = 400*exp(-200*t).*cos(400*pi*t).*(t>=0);
x = (cos(100*pi*t)+sin(400*pi*t)-cos(800*pi*t)).*(t>=0);
figure(1)
subplot(311)
plot(t, w)
ylabel('w(t)')
grid on
subplot(312)
plot(t, x)
ylabel('Input - x(t)')
grid on
subplot(313)
Conv = 0.001*conv(x, h);
t = -0.1:0.0005:0.1;
disp(length(t))
disp(length(Conv))
plot(t, Conv)
ylabel('Output - Convolution')
```

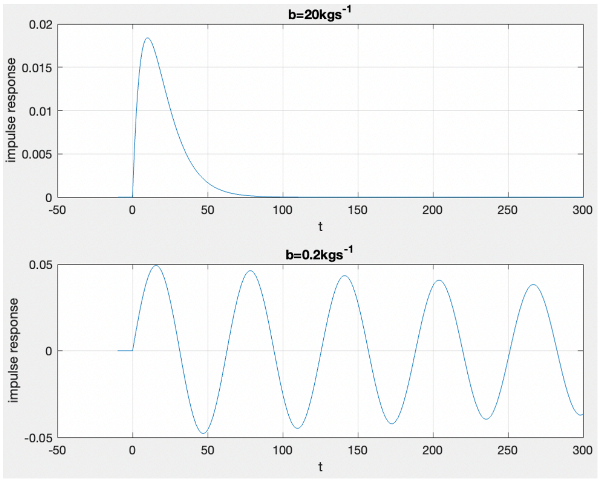
4. System response for an Industrial Shock Absorber (figure below) can be modeled with the following differential equation:

$$M\frac{d^2y(t)}{dt^2} + b\frac{dy(t)}{dt} + ky(t) = r(t) \qquad \ldots (2)$$

Let's assume the mass of the damper M is 100 kg, the spring constant k is 1 kgs^{-2}, and the friction coefficient b is 20 kgs^{-1}. Using Matlab built-in differential equation solver `dsolve()` to derive the impulse response $h_1(t)$ for this Industrial Shock Absorber and the impulse response $h_1(t)$ from a range $-10s \le t \le 300s$.
Overtime the oil inside the shock absorber degrades and the friction coefficient b becomes 0.2 kgs^{-1}. Derive the new impulse response $h_2(t)$ for this Industrial Shock Absorber and plot $h_2(t)$ from a range $-10s \le t \le 300s$.

Summary:
We define the differential with both values of b and call dsolve(), plotting and printing the result. We use y(0)=0 and Dy(0)=0 and set the differential == dirac(t).

Results:



```
>> lab3problem4
(t*exp(-t/10)*sign(t))/200

(5*1111^(1/2)*exp(-t/1000)*sin((3*1111^(1/2)*t)/1000)*sign(t))/3333
```

Code:

```
M=100;
k=1;
b1=20;
b2=0.2;
clf; % clear current figure
syms y(t) % Define the symbolic function
ode1 = M*diff(y, t, 2) + b1*diff(y, t) + k*y == dirac(t);
ode2 = M*diff(y, t, 2) + b2*diff(y, t) + k*y == dirac(t);
Dy = diff(y, t);
t = -10:0.01:300;
sol1 = eval(dsolve(ode1, [y(0) == 0, Dy(0) == 0])).*(t>=0);
sol2 = eval(dsolve(ode2, [y(0) == 0, Dy(0) == 0])).*(t>=0);
subplot(2, 1, 1)
disp(dsolve(ode1, [y(0) == 0, Dy(0) == 0]));
plot(t, sol1);
xlabel('t')
ylabel('impulse response');
title('b=20kgs^{-1}')
grid on; % grid on
subplot(2, 1, 2)
disp(dsolve(ode2, [y(0) == 0, Dy(0) == 0]));
plot(t, sol2);
xlabel('t')
ylabel('impulse response');
title('b=0.2kgs^{-1}')
grid on; % grid on
```