Table of Contents

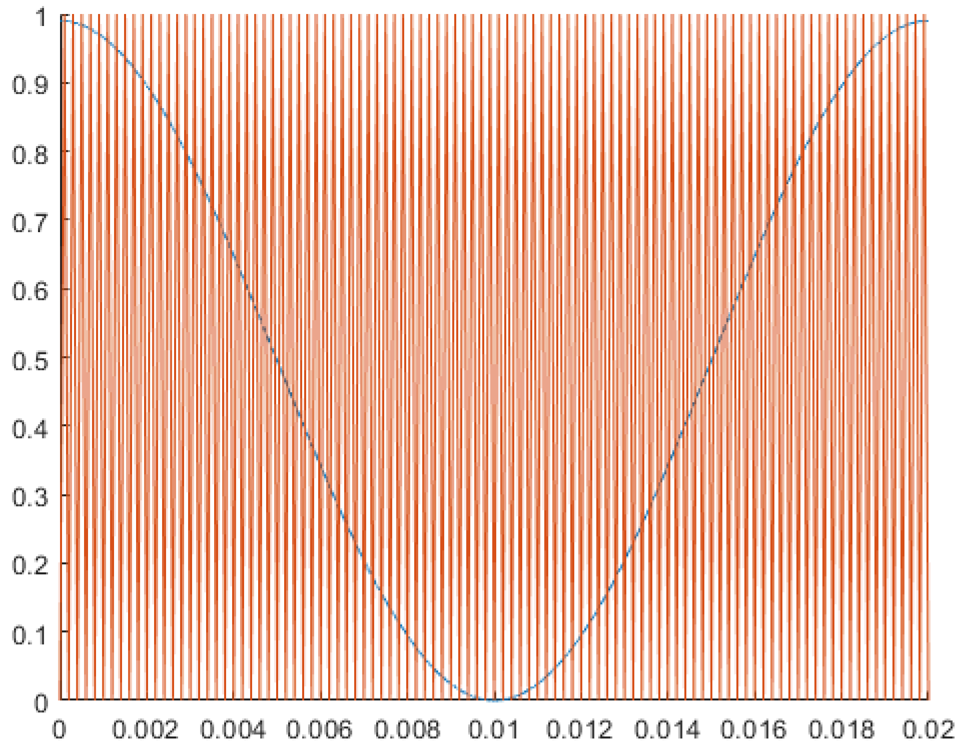
.... 1
Part 1 1
Part 2 3
Part 3 4
Part 4 6
Functions 8

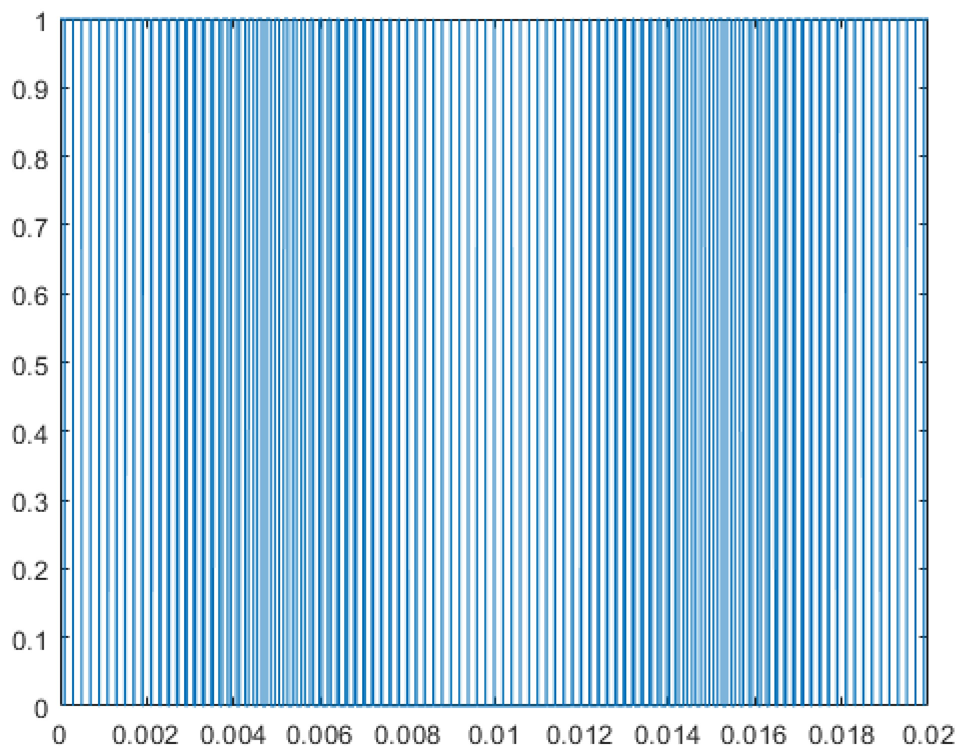

```
%**************************************************************************
% File name: lab4.m
%
% Programmer: Yaqub Mukhtar
%
% Course: EGRE 335, Fall 2023       Instructor: Dr. Ashok Iyer
%
% Description: This is the implementation file for the Lab 4 assignment,
% where the task is to simulate the response of an AC synchronous motor
% to an SPWM input signal.
%**************************************************************************

close all;
clc;
clear;
```

Part 1

Defining simulation parameters

```
T0 = 0.02;
f_c = 5000;
f_m = 50;
dt = 1e-6;
t = 0:dt:T0;

% Defining carrier signal using sawtooth function
carrier_signal = 1/2 * sawtooth(2*pi*f_c*t, 1/2) + 1/2;
% Defining message signal
message_signal = 0.99/2 * cos(2*pi*f_m*t) + 0.99/2;
% SPWM is 1 when messsage is larger or equal to the carrier, and 0 else
spwm_signal = (message_signal >= carrier_signal);

figure(1)
hold on
    plot(t, message_signal);
    plot(t, carrier_signal);
hold off

figure(2)
plot(t, spwm_signal);
```

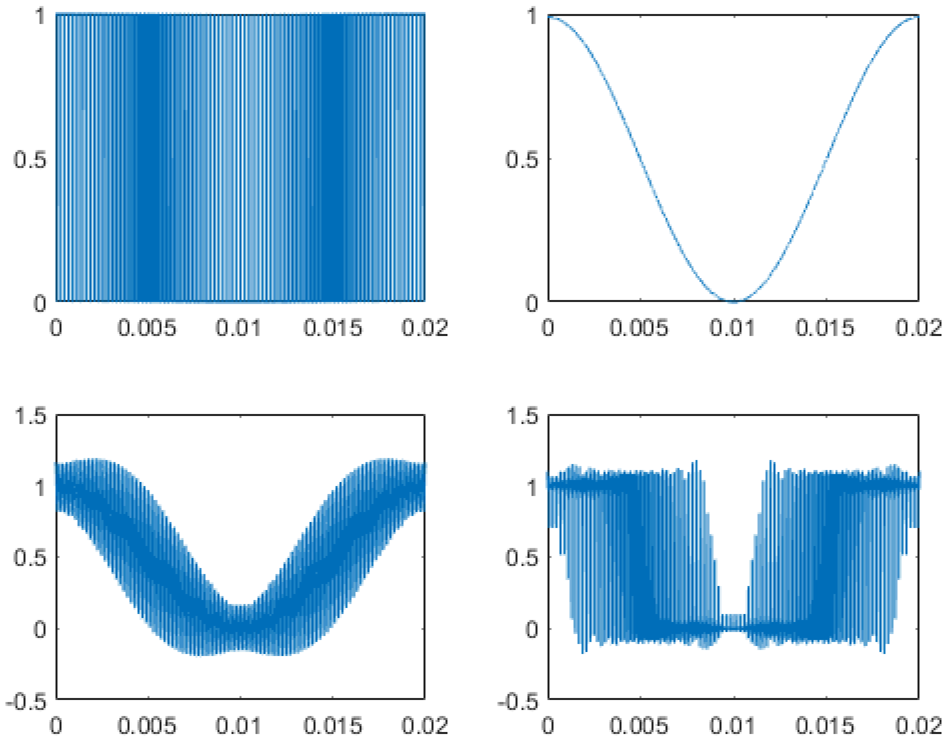

1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 0.002 0.004 0.006 0.008 0.01 0.012 0.014 0.016 0.018 0.02


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 0.002 0.004 0.006 0.008 0.01 0.012 0.014 0.016 0.018 0.02

Part 2

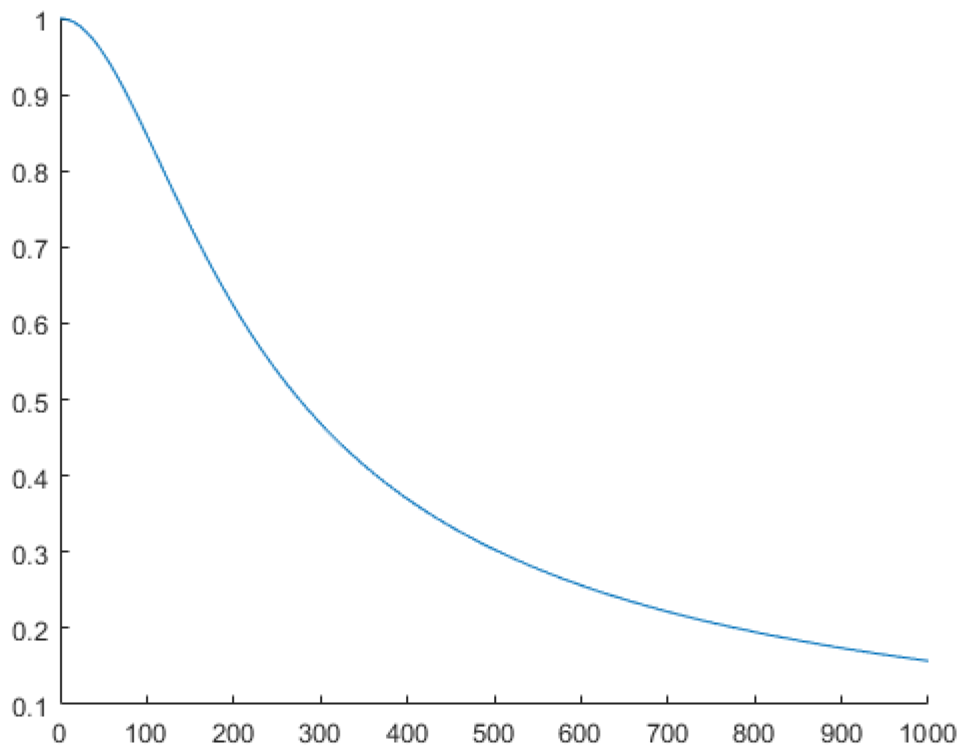
Calculating a_0

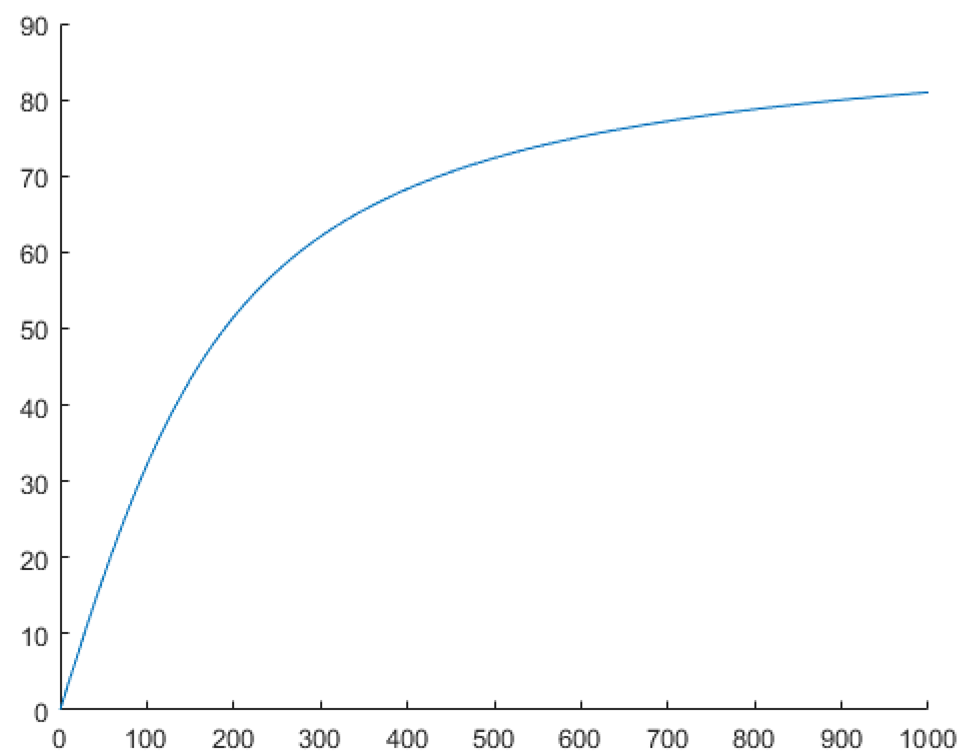
```
dc_component = 1/T0 * integral(spwm_signal, dt);

% Calculating a_1 + a_2 ... + a_1000
for n = 1:1000
    coefficient(n) = 2/T0 * integral(spwm_signal .* cos(2*pi*f_m*n*t), dt);
end

harmonic_100 = dc_component;       % Initialize a value for 0th harmonic
% Calculate first 500 harmonics of SPWM signal
for n = 1:100
    harmonic_100 = harmonic_100 + coefficient(n)*cos(2*pi*n*f_m*t);
end

harmonic_1000 = dc_component;       % Initialize a value for 0th harmonic
% Calculate first 1000 harmonics of SPWM signal
for n = 1:1000
    harmonic_1000 = harmonic_1000 + coefficient(n)*cos(2*pi*n*f_m*t);
end

figure(3)
hold on
    subplot(2,2,1);
    plot(t, spwm_signal)
    subplot(2,2,2);
    plot(t, coefficient(1)*cos(2*pi*f_m*t) + dc_component);
    subplot(2,2,3);
    plot(t, harmonic_100);
    subplot(2,2,4);
    plot(t, harmonic_1000);
hold off
```



Part 3

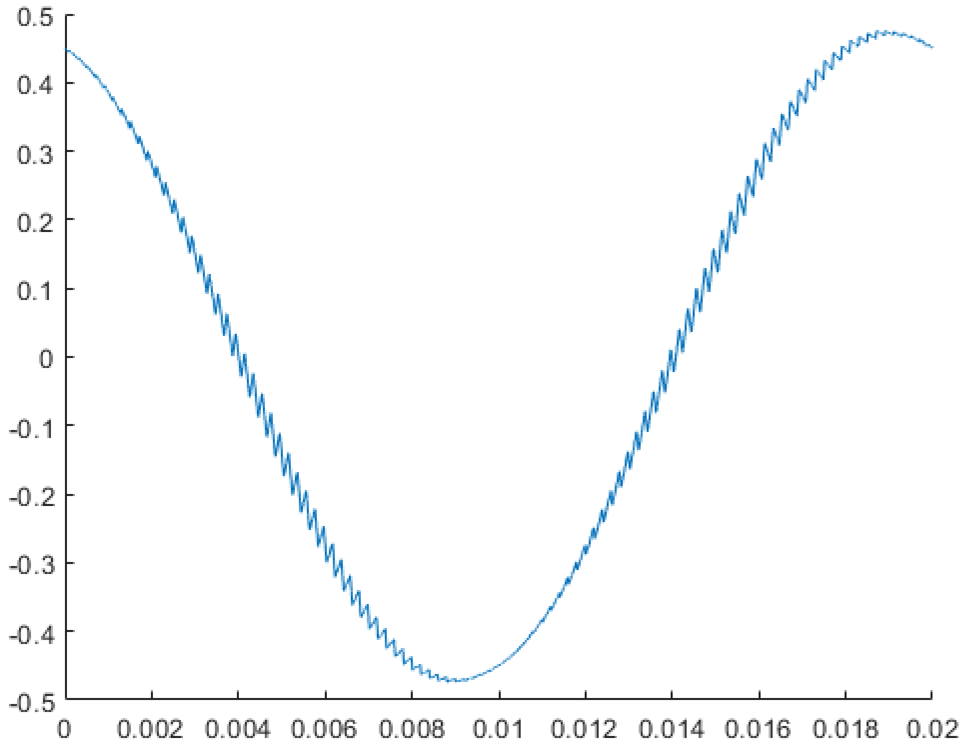
```
R = 0.5;
L = 0.0005;

f = 0:1000;
figure(4)
hold on
    plot(f, magnitude(f, R, L));
hold off

figure(5)
hold on
    plot(f, rad2deg(phase(f, R, L)))
hold off
```

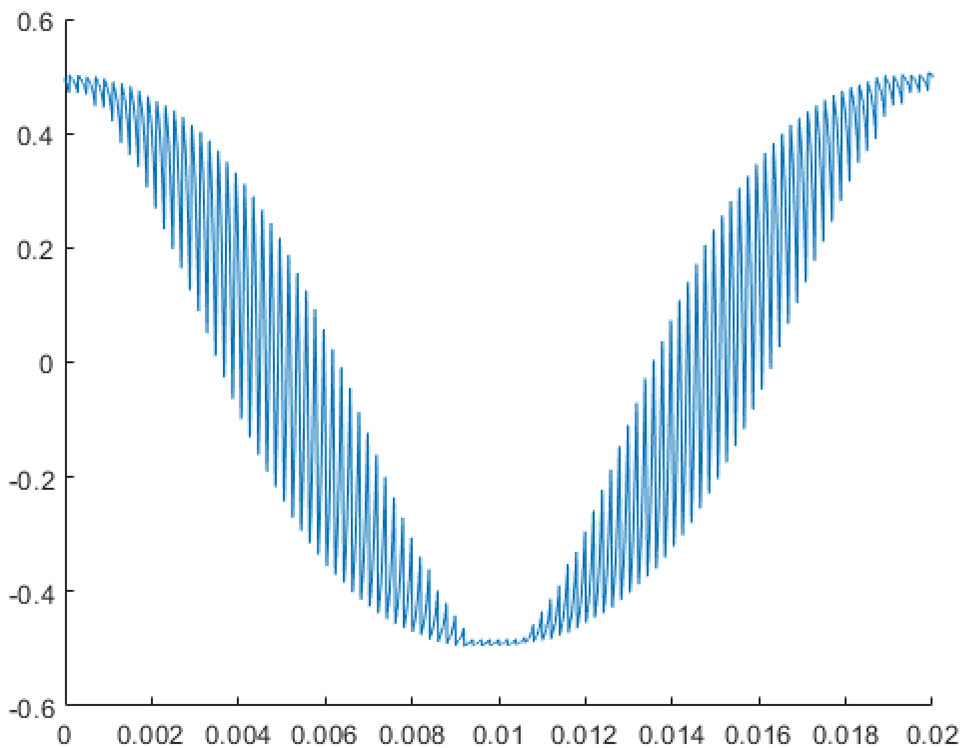

1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
100
200
300
400
500
600
700
800
900
1000


90
80
70
60
50
40
30
20
10
0
0
100
200
300
400
500
600
700
800
900
1000

Part 4

```
output_signal_1 = 0;        % Initial conditions of output signal are 0
% Calculating output signal using superposition of harmonics
for n = 1:1000
    output_signal_1 = output_signal_1 + magnitude(n*f_m, R,
 L)*coefficient(n)*cos(2*pi*n*f_m*t + phase(n*f_m, R, L));
end

figure(6)
hold on
    plot(t, output_signal_1)
hold off

% Changing simulation parameters to observe change
L = 0.00005;

output_signal_2 = 0;
for n = 1:1000
    output_signal_2 = output_signal_2 + magnitude(n*f_m, R,
 L)*coefficient(n)*cos(2*pi*n*f_m*t + phase(n*f_m, R, L));
end

figure(7)
hold on
    plot(t, output_signal_2)
 hold off
```


0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
0.002
0.004
0.006
0.008
0.01
0.012
0.014
0.016
0.018
0.02


0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
0.002
0.004
0.006
0.008
0.01
0.012
0.014
0.016
0.018
0.02

Functions

```
function I = integral(x, dt)
% Function: integral
% Input parameters: Value vector (x), time increment (dt)
% Output: Integral of x (I)
% Description: Calculates the integral of an arbitrary input signal for
% the period of time the signal was defined for. Returns a single value.
    I = 0;      % Initialize integral value
    for i = 1:length(x)
        I = I + x(i)*dt;        % Riemann summation
    end
end

function M = magnitude(f, R, L)
% Function: magnitude
% Input parameters: Frequency Vector (f), Resistance (R), Inductance (L)
% Output: RL circuit magnitude (M)
% Description: Generates the amplitdue spectra of an RL lowpass filter for
% given parameters and frequency range.
    M = R ./ (sqrt(R^2 + (2*pi*f).^2 * L^2));
end

function phi = phase(f, R, L)
% Function: phase
% Input parameters: Frequency Vector (f), Resistance (R), Inductance (L)
% Output: RL circuit phase (phi)
% Description: Generates the phase spectra of an RL lowpass filter for
% given parameters and frequency range.
    phi = atan(2*pi*f*L/R);
end
```

Published with MATLAB® R2022b