Moe Ye Htet ECE 43300 htet@purdue.edu

Project 3 Report

Part 1: Analysis of 6-diode rectifier

The source voltages of the generator were given as follows.

$$e_{as} = 100\cos(100t)$$

$$e_{bs} = 100\cos(100t - \frac{2\pi}{3})$$

$$e_{cs} = 100\cos(100t + \frac{2\pi}{3})$$

The inductance values was given as 1 mH. The MATLAB code used to do the calculations and obtain the plots was attached.

a. Average Output Voltage vs Current Plot over Modes 1 and 2

For (a) and (d), average output voltage and current formulas of Mode 1 and 2 of the 6-diode rectifier were used to generate the plot and determine the load that yields the maximum power delivered. The formulas used were as follows.

For Mode 1,

$$\langle V_{dc}' \rangle = \frac{3\sqrt{3}V_{ac}}{\pi} - \frac{3}{\pi}\omega_{ac} L_{ac} i_{dc}$$

For Mode 2,

$$\langle V_{dc}' \rangle = \frac{9V}{2\pi}\cos(\alpha + \frac{\pi}{6})$$

$$i_{dc} = \frac{\sqrt{3}V}{2\omega_{ac}L_{ac}}\sin(\alpha + \frac{\pi}{6})$$

Using the MATLAB code, the obtained plot for average output voltage vs current is as follows.

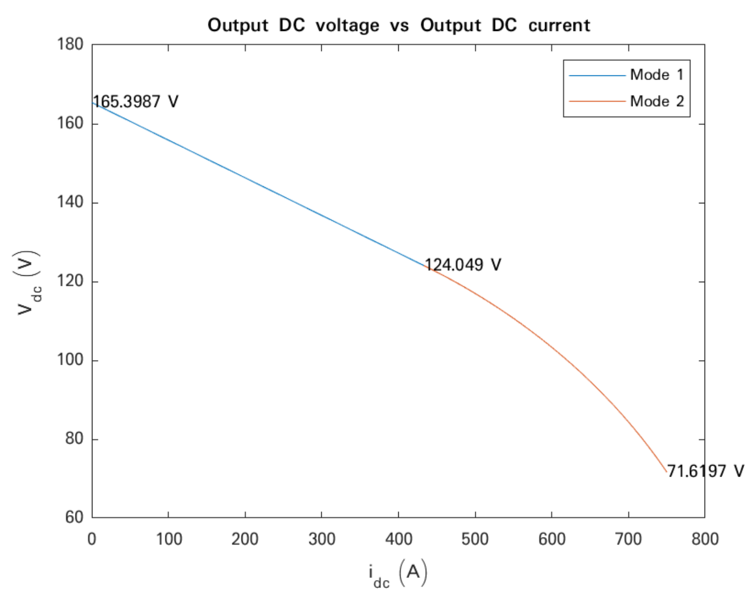


Figure 1: Average Output Voltage versus Output Current over Mode 1 and 2

b. Phase a current for commutation angle of 45°

To find the phase a current, the average dc voltage was first calculated using the following formula.

$$V_{dc}=\frac{3\sqrt{3}V}{2\pi}(\cos(\gamma)+1)$$

Then, the dc output current was calculated using average dc voltage using the following formula.

$$V_{dc}=\frac{3\sqrt{3}V}{\pi}-\frac{3}{pi}\omega_{ac}L_{ac}i_{dc}$$

Then, following the diode conduction logic, the phase a current plot was generated using the MATLAB code.

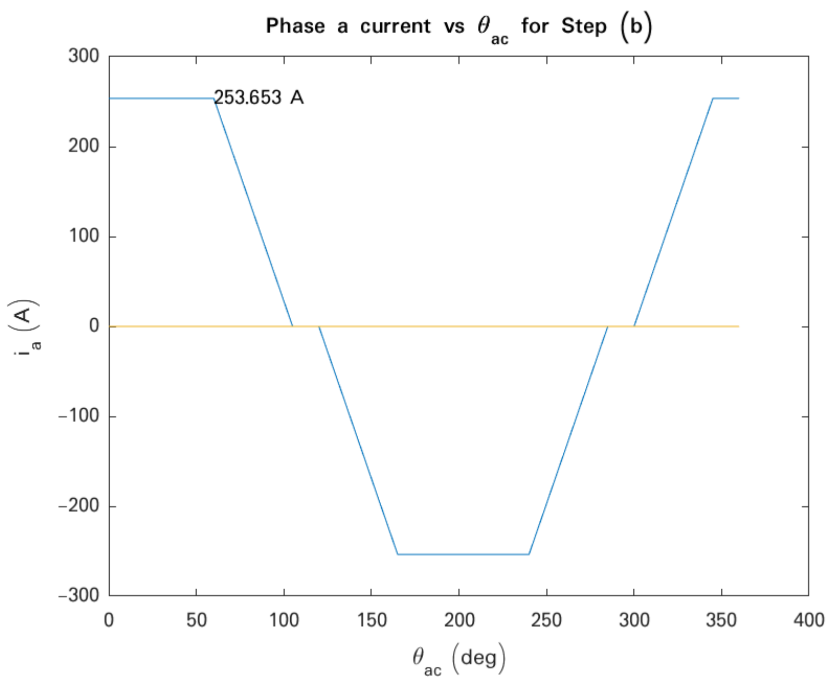


Figure 2: Phase a current with commutation angle of 45°

c. Phase a current for delay angle of 10°

To find phase a current with a delay angle of 10°, output dc current was calculated using the following formula.

$$i_{dc}=\frac{\sqrt{3}V}{2\,\omega_{ac}L_{ac}}\sin\left(\alpha+\frac{\pi}{6}\right)$$

Then, following the diode conduction logic, the phase a current plot was generated using the MATLAB code.

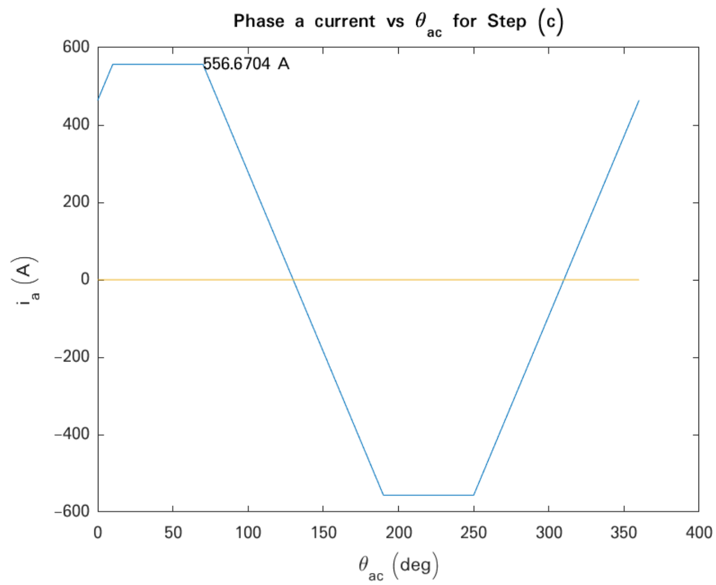


Figure 3: Phase a current with delay angle of 10°

d. Load that yields the maximum power delivered

The load resistance value that yields the maximum power delivered was also calculated using the above formulas for Mode 1 and 2 and the MATLAB code. The average output voltage and current were calculated first and the load resistance and power delivered were from voltage and current values. Then, the power delivered vs load resistance plot was generated from the plot, the point where the power delivered is maximum was determined and the load resistance for that point was recorded. The value of load that yields the maximum power delivered was found to be 0.1654 Ω.

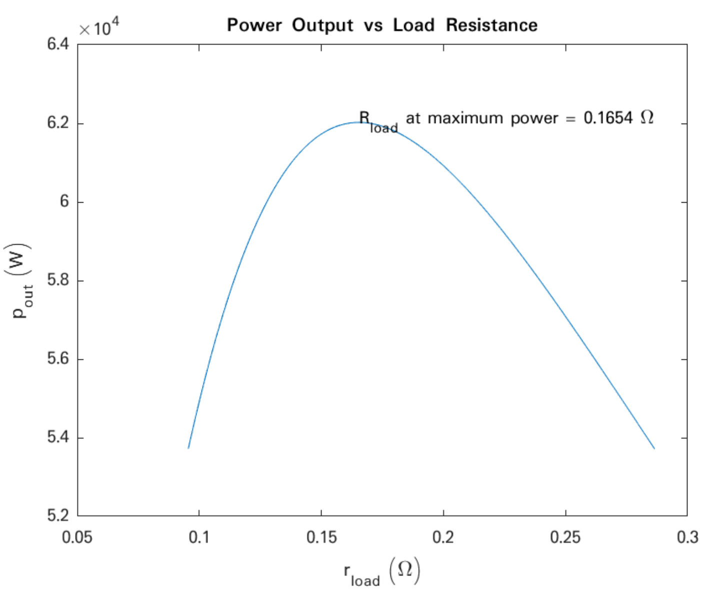


Figure 4: Power Delivered vs load resistance plot

Part 2: Simulation of 6-diode Rectifier

In this part, a 6-diode rectifier with a dc inductor of 5 mH was simulated using a backward Euler integration algorithm. The simulation code was written in MATLAB. The simulation was done by using a while loop and equations related to the circuit. The MATLAB code was attached.

The plots of phase a current, output dc voltage and output dc current were plotted for the load resistor values of part 1(b) and part 1(c) which were determined to be 0.5566 Ω and 0.1971 Ω respectively. The plots obtained were as follows.

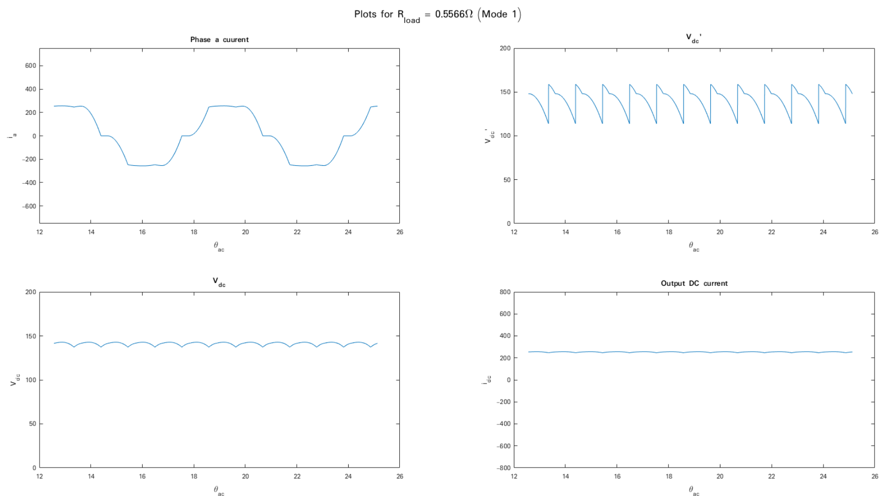


Figure 5: Phase a current, Output current and DC voltage for 1(b)

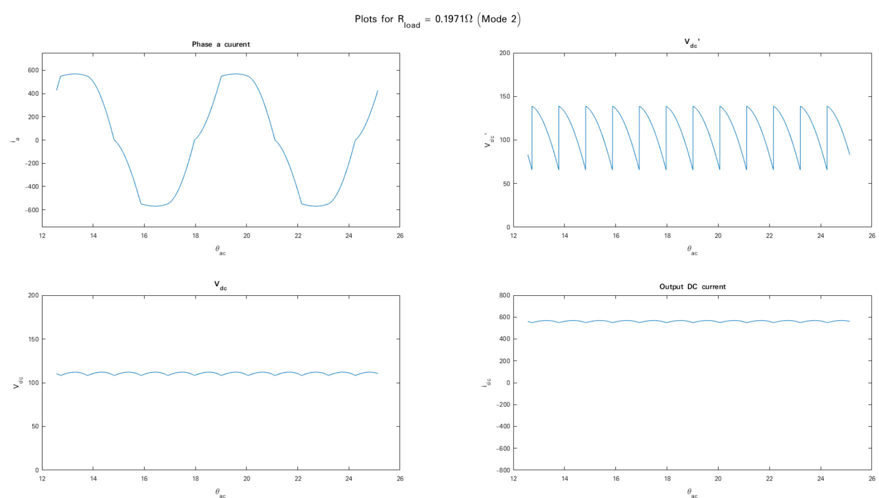
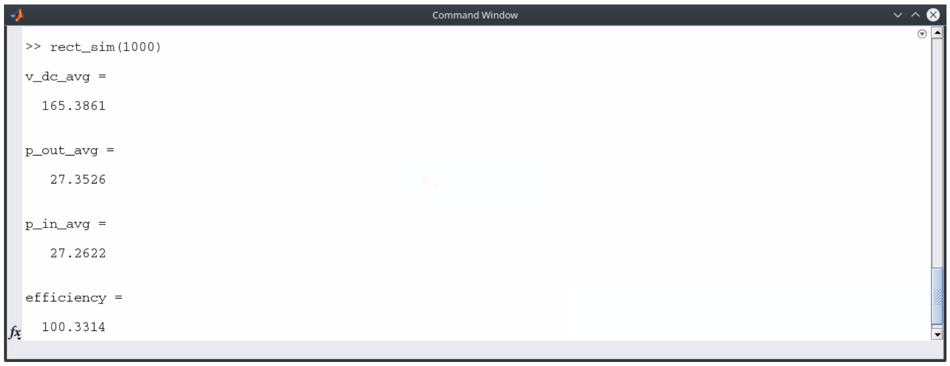
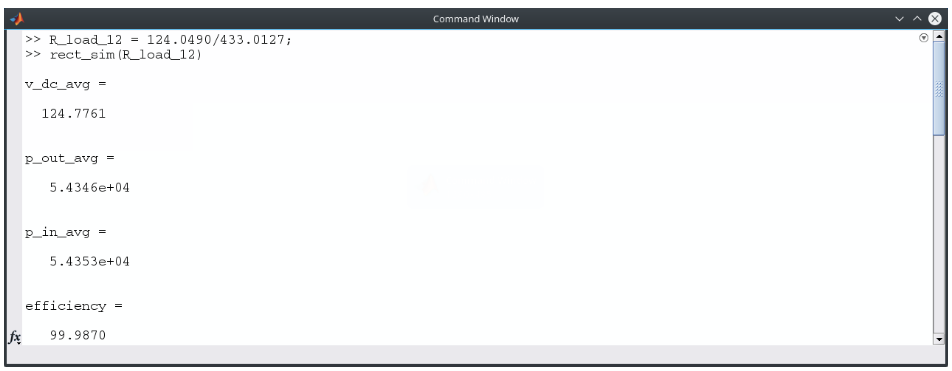


Figure 5: Phase a current, Output current and DC voltage for 1(c)

From the above graphs, it can be seen that the current values from the plots match the analytical results obtained in the previous part.

Then, the simulation is validated by comparing the average dc voltages obtained in 1(a). From 1(a), three voltages at the borders of the modes are 165.3987 V at the start of Mode 1, 124.049 V at the border of Mode 1 and 2 and 71.6197 V at the border of Mode 2 and 3. From the simulations, the average dc voltages are 165.3861 at the start of Mode 1 (simulated with load resistance as 1 kΩ), 124.7761 V at the border of Mode 1 and 2 and 72.8881 V at the border of Mode 2 and 3 as seen in Command Window outputs below. Therefore, it can be concluded that the simulation model is working correctly.





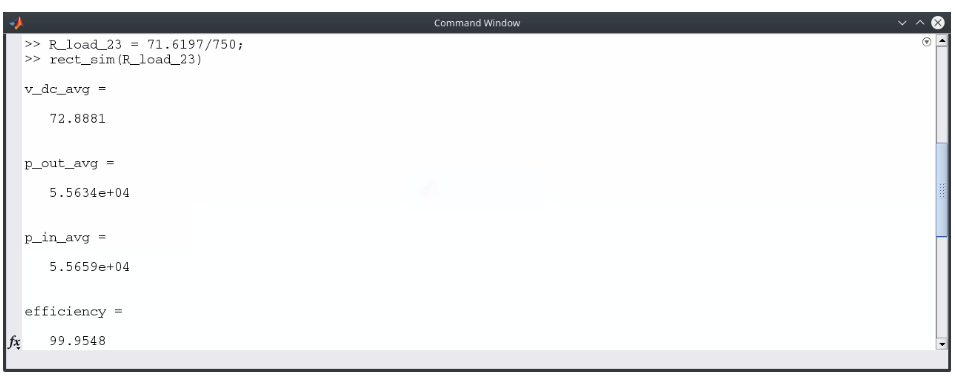


Figure 6: Command Window outputs of simulation for 1(a)

Then, the average input and output powers for the load values of 1(b) and 1(c) and efficiency were calculated. The average input and output powers for 1(b) were 35789 W and 35784 W respectively and the efficiency was 99.9860%. The average input and output powers for 1(c) were 62280 W and 62272 W respectively and the efficiency was 99.9868%. Since the efficiency was close to 100%, it can also be validated that the simulation model was working correctly.

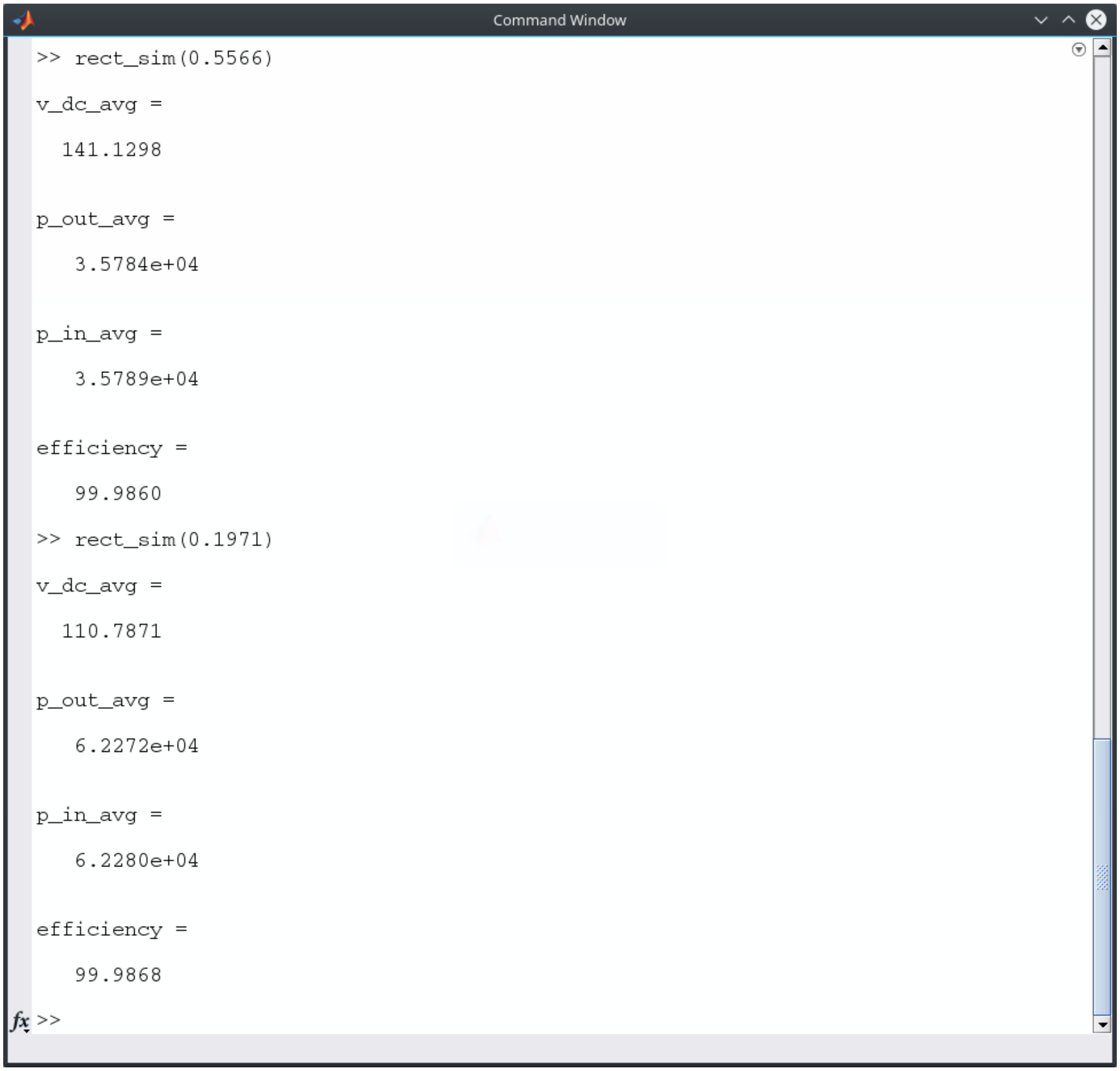


Figure 7: Command Window outputs of simulation for 1(b) and 1(c)

Codes

1. analytical.m

```
clear all
clc

V_ac = 100;
w_ac = 100;
L_ac = 1e - 3;

V_dc_12 = 9 * sqrt(3) * V_ac / (4 * pi);
V_dc_23 = 9 * V_ac / (4 * pi);

V_dc_mode2 = V_dc_23:0.00005:V_dc_12;
V_dc_mode1 = V_dc_12:0.00005:3 * sqrt(3) * V_ac / pi;

i_dc_mode1 = (pi / (3 * w_ac * L_ac)) .* ((3 * sqrt(3) * V_ac / pi) - V_dc_mode1);
alpha = acos((2 * pi .* V_dc_mode2) / (9 * V_ac)) - pi / 6;
i_dc_mode2 = ((sqrt(3) * V_ac) / (2 * w_ac * L_ac)) .* sin(alpha + pi / 6);

i_dc = [i_dc_mode1, i_dc_mode2];
V_dc = [V_dc_mode1, V_dc_mode2];

p_out = V_dc_mode2 .* i_dc_mode2;
r_load = V_dc_mode2 ./ i_dc_mode2;

[p_max, p_max_index] = max(p_out)
r_p_max = r_load(p_max_index)

theta_ac = 0:5:360;
gamma = 45;
V_dc_b = ((3 * sqrt(3) * V_ac) / (2 * pi)) * (cosd(gamma) + 1);
i_dc_b = (pi / (3 * w_ac * L_ac)) * (((3 * sqrt(3) * V_ac) / pi) - V_dc_b);
i_ac_b = zeros(length(theta_ac));

pos = find(theta_ac <= 60 | theta_ac >= 300 + gamma);
neg = find(theta_ac >= 120 + gamma & theta_ac <= 240);
neg_slope_1 = find(theta_ac > 60 & theta_ac < 60 + gamma);
neg_slope_2 = find(theta_ac > 120 & theta_ac < 120 + gamma);
pos_slope_1 = find(theta_ac > 240 & theta_ac < 240 + gamma);
pos_slope_2 = find(theta_ac > 300 & theta_ac < 300 + gamma);
zero = find((theta_ac >= 60 + gamma & theta_ac <= 120) | (theta_ac >= 240 + gamma &
theta_ac <= 300));

i_ac_b(pos) = i_dc_b;
i_ac_b(neg) = - i_dc_b;
i_ac_b(neg_slope_1) = - (i_dc_b / 45) .* (theta_ac(neg_slope_1) - 60) + i_dc_b;
i_ac_b(neg_slope_2) = - (i_dc_b / 45) .* (theta_ac(neg_slope_2) - 120);
i_ac_b(pos_slope_1) = (i_dc_b / 45) .* (theta_ac(pos_slope_1) - 240) - i_dc_b;
i_ac_b(pos_slope_2) = (i_dc_b / 45) .* (theta_ac(pos_slope_2) - 300);
i_ac_b(zero) = 0;
```

```
alpha = 10;
V_dc_c = ((9 * V_ac) / (2 * pi)) * cosd(alpha + 30);
i_dc_c = ((sqrt(3) * V_ac) / (2 * w_ac * L_ac)) * sind(alpha + 30);
i_ac_c = zeros(length(theta_ac));

pos_c = find(theta_ac >= alpha & theta_ac <= 60 + alpha);
neg_c = find(theta_ac >= 180 + alpha & theta_ac <= 240 + alpha);
neg_slope_c = find(theta_ac > 60 + alpha & theta_ac < 180 + alpha);
pos_slope_c_1 = find(theta_ac < alpha);
pos_slope_c_2 = find(theta_ac > 240 + alpha);

i_ac_c(pos_c) = i_dc_c;
i_ac_c(neg_c) = - i_dc_c;
i_ac_c(neg_slope_c) = - (i_dc_c / 60) * (theta_ac(neg_slope_c) - 60 - alpha) +
i_dc_c;
i_ac_c(pos_slope_c_1) = (i_dc_c / 60) * (theta_ac(pos_slope_c_1) + 60 - alpha);
i_ac_c(pos_slope_c_2) = (i_dc_c / 60) * (theta_ac(pos_slope_c_2) - 240 - alpha) -
i_dc_c;

n = find(V_dc == V_dc_12);
m = find(V_dc == V_dc_23);
[V_dc_max, o] = max(V_dc);
p = find(theta_ac == 60);
q = find(theta_ac == 60 + alpha);

figure;
plot(i_dc_mode1, V_dc_mode1)
hold on
plot(i_dc_mode2, V_dc_mode2)
legend('Mode 1', 'Mode 2')
xlabel('i_d_c (A)')
ylabel('V_d_c (V)')
title('Output DC voltage vs Output DC current')
text(i_dc(o), V_dc(o), [num2str(V_dc_max), ' V'])
text(i_dc(n), V_dc(n), [num2str(V_dc_12), ' V'])
text(i_dc(m), V_dc(m), [num2str(V_dc_23), ' V'])
saveas(gcf, 'v-vs-i', 'tiffn')

figure;
plot(r_load, p_out)
xlabel('r_l_o_a_d (\Omega)')
ylabel('p_o_u_t (W)')
title('Power Output vs Load Resistance')
text(r_p_max, p_max, ['R_l_o_a_d at maximum power = ', num2str(r_p_max), ' \
Omega'])
saveas(gcf, 'p-vs-r', 'tiffn')

figure;
plot(theta_ac, i_ac_b)
xlabel('\theta_a_c (deg)')
ylabel('i_a (A)')
title('Phase a current vs \theta_a_c for Step (b)')
text(theta_ac(p), i_ac_b(p), [num2str(i_ac_b(p)), ' A'])
saveas(gcf, 'ia-vs-theta-b', 'tiffn')
```

```
figure;
plot(theta_ac, i_ac_c)
xlabel('\theta_a_c (deg)')
ylabel('i_a (A)')
title('Phase a current vs \theta_a_c for Step (c)')
text(theta_ac(q), i_ac_c(p), [num2str(i_ac_c(q)), ' A'])
saveas(gcf, 'ia-vs-theta-c', 'tiffn')
```

2. rect_sim.m

```
function rect_sim(R_load)
% given
L_ac = 1e-3;
L_dc = 5e-3;
w_ac = 100;
T_ac = (2*pi)/w_ac;
esp = 0.01;
tau = 1e-5;
%tau = 1e-2;

R_load_12 = 124.0490/433.0127; % borderline mode 1&2
R_load_23 = 71.6197/750; % borderline mode 2&3

if R_load > R_load_12
    mode = 1;
elseif (R_load < R_load_12) & (R_load > R_load_23)
    mode = 2;
elseif R_load == R_load_12
    mode = 12;
elseif R_load == R_load_23
    mode = 23;
end

% init
i_d1(1) = 0;
i_d3(1) = 0;
i_d5(1) = 0;
i_a(1) = 0;
i_b(1) = 0;
i_c(1) = 0;
i_dc(1) = 0;
v_rect(1) = 0;
v_dc(1) = 0;

% time
del_t = tau/100;
t_end = 5*T_ac;
t(1) = 0;
k = 1;

while t(k) < t_end

    e_a(k) = 100*cos(100*t(k));
```

```
e_b(k) = 100*cos(100*t(k) - 2*pi/3);
e_c(k) = 100*cos(100*t(k) + 2*pi/3);

if i_a(k) > esp
    v_ag(k) = v_rect(k);
elseif i_a(k) < -esp
    v_ag(k) = 0;
else
    v_ag(k) = (v_rect(k)/(2*esp))*i_a(k) + (0.5*v_rect(k));
end

if i_b(k) > esp
    v_bg(k) = v_rect(k);
elseif i_b(k) < -esp
    v_bg(k) = 0;
else
    v_bg(k) = (v_rect(k)/(2*esp))*i_b(k) + (0.5*v_rect(k));
end

if i_c(k) > esp
    v_cg(k) = v_rect(k);
elseif i_c(k) < -esp
    v_cg(k) = 0;
else
    v_cg(k) = (v_rect(k)/(2*esp))*i_c(k) + (0.5*v_rect(k));
end

v_a(k) = (2/3)*v_ag(k) - (1/3)*v_bg(k) - (1/3)*v_cg(k);
v_b(k) = (2/3)*v_bg(k) - (1/3)*v_ag(k) - (1/3)*v_cg(k);
v_c(k) = (2/3)*v_cg(k) - (1/3)*v_ag(k) - (1/3)*v_bg(k);

i_a(k+1) = i_a(k) + (del_t/L_ac)*(e_a(k) - v_a(k)); % need init
i_b(k+1) = i_b(k) + (del_t/L_ac)*(e_b(k) - v_b(k)); % need init
i_c(k+1) = i_c(k) + (del_t/L_ac)*(e_c(k) - v_c(k)); % need init

if i_a(k+1) > 0
    i_d1(k+1) = i_a(k+1); % need int
else
    i_d1(k+1) = 0;
end

if i_b(k+1) > 0
    i_d3(k+1) = i_b(k+1); % need init
else
    i_d3(k+1) = 0;
end

if i_c(k+1) > 0
    i_d5(k+1) = i_c(k+1); % need init
else
    i_d5(k+1) = 0;
end

i_dc(k+1) = i_d1(k+1) + i_d3(k+1) + i_d5(k+1); % need init
```

```
    v_rect(k+1) = (1/(1+(del_t/tau))) * (v_rect(k) + ((L_dc*(i_dc(k+1) - i_dc(k)))/
tau) + ((del_t*i_dc(k+1)*R_load)/tau)); % need init
    v_dc(k+1) = R_load*i_dc(k+1); % need init

    t(k+1) = t(k) + del_t;
    k = k+1;
end

theta_ac = w_ac.*t;

n = find(t >= 3*T_ac & t < 4*T_ac);
m = find(t >= 2*T_ac & t < 4*T_ac);

v_dc_avg = average(v_dc(n),T_ac,del_t)
i_dc_avg = average(i_dc(n),T_ac,del_t);

p_out_avg = v_dc_avg*i_dc_avg

p_in = v_a(n).*i_a(n) + v_b(n).*i_b(n) + v_c(n).*i_c(n);
p_in_avg = average(p_in,T_ac,del_t)

efficiency = (p_out_avg/p_in_avg)*100

subplot(2,2,1)
plot(theta_ac(m),i_a(m))
ylim([-750 750])
xlabel('\theta_a_c')
ylabel('i_a')
title('Phase a cuurent')

subplot(2,2,2)
plot(theta_ac(m),v_rect(m))
ylim([0 200])
xlabel('\theta_a_c')
ylabel('V_d_c''')
title('V_d_c''')

subplot(2,2,3)
plot(theta_ac(m),v_dc(m))
ylim([0 200])
xlabel('\theta_a_c')
ylabel('V_d_c')
title('V_d_c')

subplot(2,2,4)
plot(theta_ac(m),i_dc(m))
ylim([-800 800])
xlabel('\theta_a_c')
ylabel('i_d_c')
title('Output DC current')

if mode == 12
```

```
    sgtitle(['Plots for R_l_o_a_d = ', num2str(R_load_12), '\Omega (Borderline Mode 1 & 2)'])
elseif mode == 23
    sgtitle(['Plots for R_l_o_a_d = ', num2str(R_load_23), '\Omega (Borderline Mode 2 & 3)'])
else
    sgtitle(['Plots for R_l_o_a_d = ', num2str(R_load), '\Omega (Mode ', num2str(mode),')'])
end
```

3. average.m

```
function avg = average(x,T,dt)
n_period = T/dt;
avg = 0;

for m = 1:n_period
    avg = avg + x(m)*dt/T;
end
```