Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science
6.685 Electric Machines

Problem Set 9 Solutions November 12, 2011

Problem 1: Large Generator

For reference we leave the original table of parameters here.

Machine Rationg		600	MVA
Rater Terminal Voltage		22	kV
Synchronous, d- axis reactance	x_d	2.0	
Synchronous, q- axis reactance	x_q	1.8	
Transient, d- axis reactance	x'_d	0.40	
Subtransient, d- axis reactance	x_d"	0.20	
Subtransient, q- axis reactance	x_q"	0.20	
Transient, open-circuit time constant	T_do'	5.0	s
Subtransient, open-circuit time constant	T_{do}"	0.2	s
Subtransient, open-circuit time constant	T_{qo}"	0.5	s
Inertial Constant	H	3.0	s
Armature Time Constant	T_a	0.1	s

1. The *equal mutuals* parameters are computed by inverting these expressions:

$$\begin{aligned} x_d &= x_{a\ell} + x_{ad} \\ x'_d &= x_{a\ell} + x_{ad}||x_{f\ell} \\ x_d" &= x_{a\ell} + x_{ad}||x_{f\ell}||x_{kd\ell} \\ x_q &= x_{a\ell} + x_{aq} \\ x_q" &= x_{a\ell} + x_{aq}||x_{kq\ell} \end{aligned}$$

Assuming a given value of $x_{a\ell}$ and with a little manipulation, we find the following:

$$\begin{aligned} x_{ad} &= x_d - x_{a\ell} \\ x_{f\ell} &= x_{ad}\left(\frac{x'_d - x_{a\ell}}{x_d - x'_d}\right) \\ x_{kd\ell} &= \frac{1}{\frac{1}{x_d" - x_{a\ell}} - \frac{1}{x_{ad}} - \frac{1}{x_{f\ell}}} \\ x_{aq} &= x_q - x_{a\ell} \\ x_{kq\ell} &= x_{aq}\frac{x_q" - x_{a\ell}}{x_q - x_q"} \end{aligned}$$

Now, the *transient* time constant is:

$$T'_{do} = \frac{x_{f\ell} + x_{ad}}{\omega_0 r_f}$$

and the *subtransient* time constant is:

$$T_{do}" = \frac{x_{kd\ell} + x_{ad} || x_{f\ell}}{\omega_0 r_{kd}}$$

the q-axis is similar, so the resistances are:

$$\begin{aligned} r_f &= \frac{x_{f\ell} + x_{ad}}{\omega_0 T'_{do}} \\ r_{kd} &= \frac{x_{kd\ell} + x_{ad} || x_{f\ell}}{\omega_0 T_{do}"} \\ r_{dq} &= \frac{x_{kq\ell} + x_{aq}}{\omega_0 T_{qo}"} \end{aligned}$$

These expressions are implemented in two scripts, `mi.m` and `miq.m` which are appended and are exercised by another script called `p9_2.m` which is also appended. The results are, in both per-unit and ohms (referred to the stator: see below):

```
>> p9_2
6.685 Problem Set 9, Problem 3, Part 2: Basics
Parameter            Per Unit      Ohms
  xad                   1.9           1.53
  xaq                   1.7           1.37
  xfl                 0.356          0.287
  xkdl                 0.15          0.121
  xkql                0.106         0.0857
  rf                 0.0012       0.000966
  rkd               0.00597        0.00481
  rkq               0.00958        0.00773
  ra                0.00531        0.00428
```

2. The *base* impedance is:

$$Z_B = \frac{V_B^2}{P_B} = \frac{22^2}{600} \approx .806\Omega$$

The parameters in ordinary units (ohms) are simply the per-unit parameters multiplied by the base.

3. The 'classical' result for this is:

$$i_a = \frac{1}{x_d"} e^{-\frac{t}{T_a}} - \left\{ \frac{1}{x_d} + \left(\frac{1}{x'_d} - \frac{1}{x_d} \right) e^{-\frac{t}{T'_d}} + \left(\frac{1}{x_d"} - \frac{1}{x'_d} \right) e^{-\frac{t}{T_d"}} \right\} \cos \omega t$$

This is plotted on top of the simulation results from the next part.

4. The simulation model is fairly straightforward: Assuming that speed does not change much during the simulation, the state equations are:

$$
\begin{aligned}
\frac{d\psi_d}{dt} &= \omega\psi_q - \omega_0 r_a i_d \\
\frac{d\psi_q}{dt} &= \omega\psi_d - \omega_0 r_a i_q \\
\frac{d\psi_{kd}}{dt} &= -\omega_0 r_{kd} i_{kd} \\
\frac{d\psi_{dq}}{dt} &= -\omega_0 r_k q i_{kq} \\
\frac{d\psi_f}{dt} &= v_f - \omega_0 r_f i_f
\end{aligned}
$$

During the simulation we need currents which are simple combinations of fluxes:

$$
\begin{aligned}
i_d &= \frac{x_f x_{kd} - x_{ad}^2}{D}\psi_d + \frac{x_{ad}\left(x_{ad} - x_f\right)}{D}\psi_{kd} + \frac{x_{ad}\left(x_{ad} - x_{kd}\right)}{D}\psi_f \\
i_{kd} &= \frac{x_{ad}\left(x_{ad} - x_f\right)}{D}\psi_d + \frac{x_d x_f - x_{ad}^2}{D}\psi_{kd} + \frac{x_{ad}\left(x_{ad} - x_d\right)}{D}\psi_f \\
i_f &= \frac{x_{ad}\left(x_{ad} - x_{kd}\right)}{D}\psi_d + \frac{x_{ad}\left(x_{ad} - x_d\right)}{D}\psi_{kd} + \frac{x_d x_{kd} - x_{ad}^2}{D}\psi_f
\end{aligned}
$$

and of course the determinant, which is the denominator of these expressions is:

$$
D = x_d x_{kd} x_f + 2x_{ad}^3 - x_{ad}^2\left(x_d + x_{kd} + x_f\right)
$$

Finally, we need to have initial conditions. If the machine is initially unloaded and has terminal voltage $e_{af} = 1$, the direct- and quadrature- axis fluxes are at the beginning of the fault:

$$
\begin{aligned}
\psi_d &= e_{af} = 1.0 \\
\psi_q &= 0 \\
\psi_{kd} &= \psi_d = 1.0 \\
\psi_{kq} &= \psi_q = 0 \\
\psi_f &= \psi_d + \frac{x_{f\ell}}{x_{ad}}
\end{aligned}
$$

This fault is simulated using a script appended. the resulting current, and the current from the classical method, are both included in Figure 1.

The difference between the 'simulated' and 'classical' calculations are small enough that it is difficult to see in Figure 1. We can take the difference and plot it and do that in Figure 2.

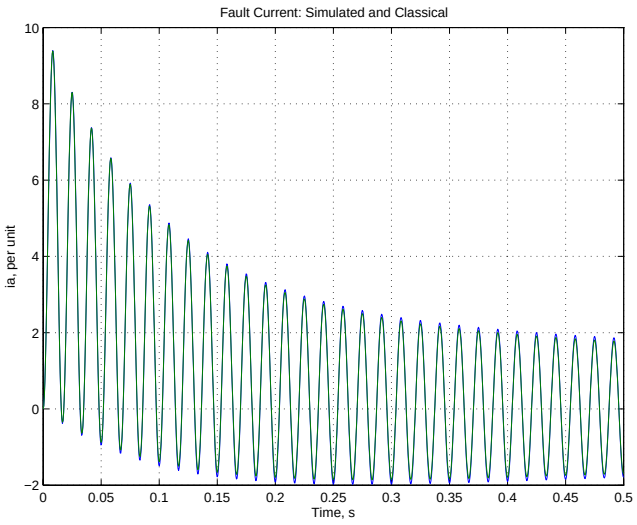

Figure 1: Phase A Fault Current: Simulated and Classical

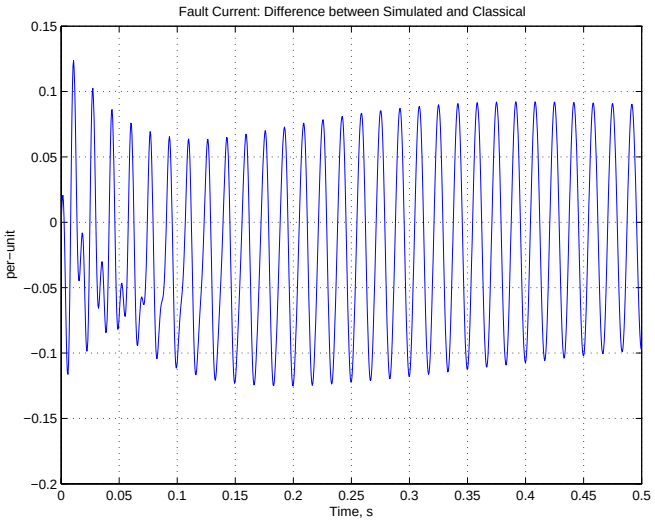

Figure 2: Phase A Fault Current: Difference between Simulated and Classical

5. The open circuit transient would (if completed) have the final flux equal to

$$\psi_{do} = e_{af} = \psi_d + x_d i_d = 2.234$$

and the transient and subtransient flux quantities are:

$$\begin{aligned} \psi'_{do} = e'_q &= \psi_d + x'_d i_d \\ \psi_{do}" = e_{qo}" &= \psi_d + x_d" i_d \\ \psi_{qo}" = -e_{do}" &= \psi_q + x_q" i_q \end{aligned}$$

then:

Values have been computed in the appended script:

```
Problem Set 9, Problem 1, part 5
Torque Angle     =      1.064  Power Factor Angle =          0
D axis flux psid =     0.4856  Q axis flux psiq   =    -0.8742
D axis current Id=     0.8742  Q axis current Iq  =     0.4856
V behind xdp, eqp=     0.8353  V behind sync x eaf=      2.234
V beh xdpp, eqpp =     0.6605  V behind xqpp edpp =    -0.9713
```

$$\begin{aligned} \psi_d &= (\psi_{do}" - \psi'_{do})\, e^{-\frac{t}{T_{do}"}} + (\psi'_{do} - \psi_{do})\, e^{-\frac{t}{T'_{do}}} + \psi_{do} \\ \psi_q &= \psi_{qo}" e^{-\frac{t}{T_{qo}"}} \end{aligned}$$

These results are shown in Figure 3

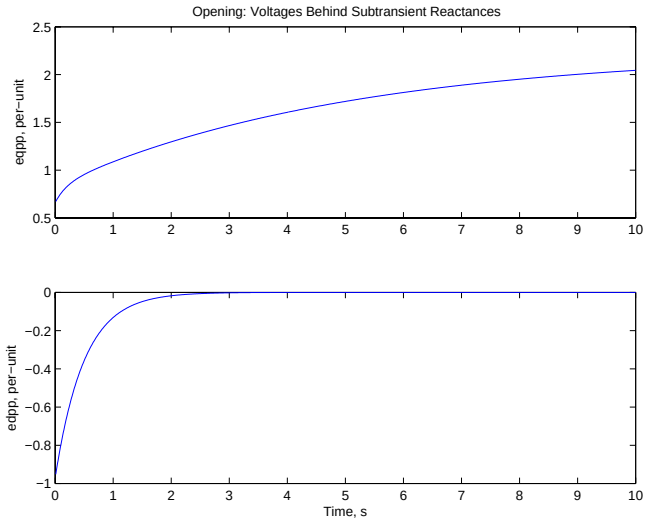

Figure 3: Voltage Behind Subtransient Reactance: Open

6. The synchronous, *transient* and 'fictitious transient' torque angle curves are:

$$T_{\text{synchronous}} = -\frac{ve_{af}}{x_d}\sin\delta - \frac{v^2}{2}\left(\frac{1}{x_q} - \frac{1}{x_d}\right)\sin 2\delta$$

$$T_{\text{transient}} = -\frac{ve'_q}{x'_d}\sin\delta - \frac{v^2}{2}\left(\frac{1}{x_q} - \frac{1}{x'_d}\right)\sin 2\delta$$

$$T_{\text{fictitious}} = -\frac{ve'_q}{x'_d}\sin\delta$$

Values for the important parameters were calculated in conjunction with Problem 6. The results are plotted in Figure 4.

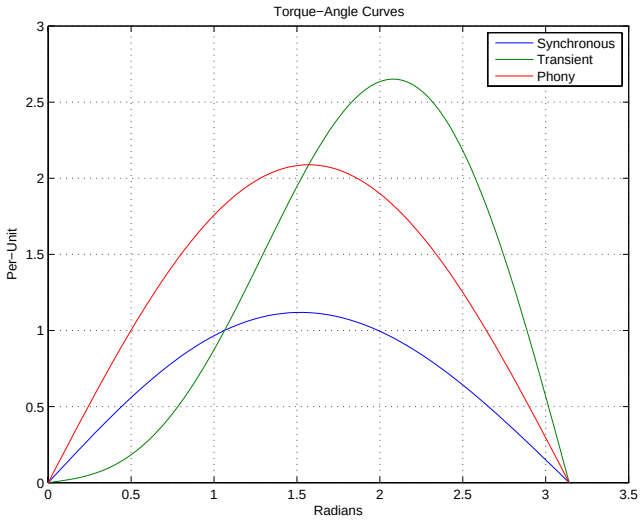

Figure 4: Torque-Angle Curves

7. Steady state operation is found with reference to Figure 5. This should look familiar, with the addition of a little piece that represents voltage produced by armature resistance. Direct and quadrature axis fluxes are related to the voltage 'inside' this resistance: $V_i = V + r_a i_a$. Then the normal rules apply. There is a voltage on the q-axis: $e_1 = V_i + jx_q i_a$ and that sets the torque angle: $\delta = \text{angle}(e_1)$. Then, as normal, $i_d = i_a \sin(\delta + \psi)$ and $i_q = i_a \cos(\delta + \psi)$. Once we have δ, we can also find the fluxes. Noting the angle of the internal voltage: $\alpha = \text{angle}(V_i)$, and with reference to the figure, $\psi_d = |V_i| \cos(\delta + \alpha)$ and $-\psi_q = |V_i| \sin(\delta + \alpha)$.

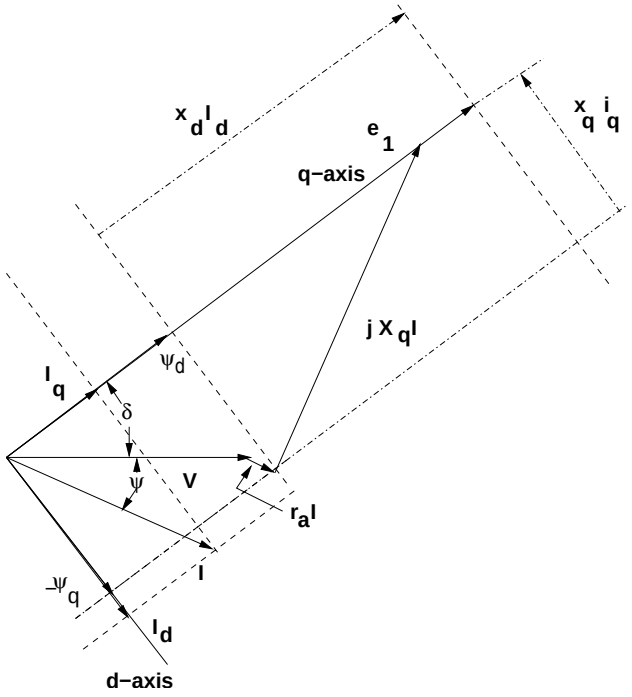

Figure 5: Steady State Operation

Now: the voltage behind synchronous reactance is $e_{af} = \psi_d + x_d i_d = x_{ad} i_{f0}$ and this fixes the value of field current and voltage: $v_f = r_f i_{f0}$. In turn, field winding flux must be $\psi_f = (x_{ad} + x_{f\ell}) i_{f0} = x_f i_{f0}$
Finally, we need to find the fluxes linked by the 'damper' windings, and these are simply $\psi_{kd} = \psi_d + x_{a\ell} i_d$ and $\psi_{kq} = \frac{x_{aq}}{x_q} \psi_q$.
And in steady state, we have found δ and $\omega = \omega_0$.

```
Problem Set 9, Steady State Operation
psid = 0.789533
psiq = -0.621035
psikd = 0.883392
psikq = -0.586533
psif = 1.3834
delta = 0.663726
Torque = 0.855305
```

```
V Field = 0.00168
```

These are the five state variables as initial conditions. To prove that we have it right, we can simulate the machine. The script for this is appended. Note that the simulation does seem to show some odd oscillation. It is not too difficult to determine that this odd oscillation, which is indeed quite small, is affected by the integration scheme (ode23 or ode45) and by the error tolerance limits that can be set to govern operation of the integration, so we can conclude that our even seeing this is an artifact of the error limits set by MATLAB. See figure 6.

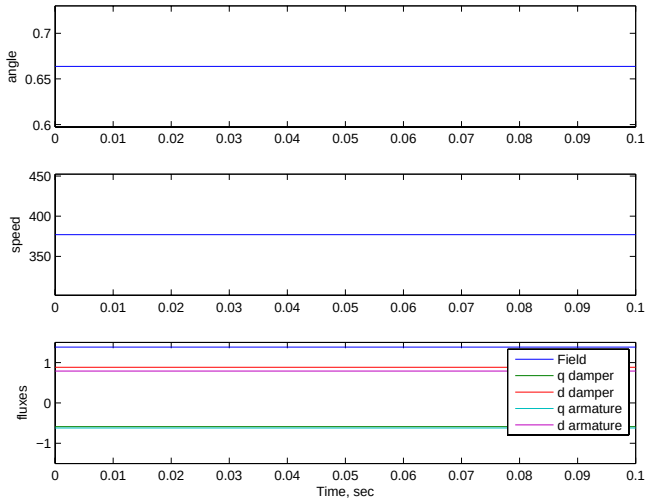

Figure 6: Steady State Operation

8. The last part of this is synchronization out of phase by 20 degrees. This requires the full, seven state simulation, and we can use the same integration routine we used in the previous part. The initial conditions are the same as for the fault simulation, with the added valuse of $\delta_0 = \pi/9$ and ω_0. Of course, mechanical torque must be set to zero. The results are shown in Figures 7 through 9.

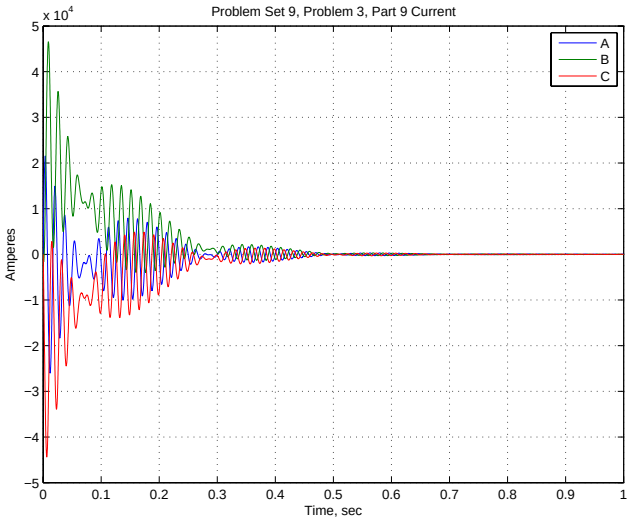

Figure 7: Synchronizing out of Phase: Phase Currents

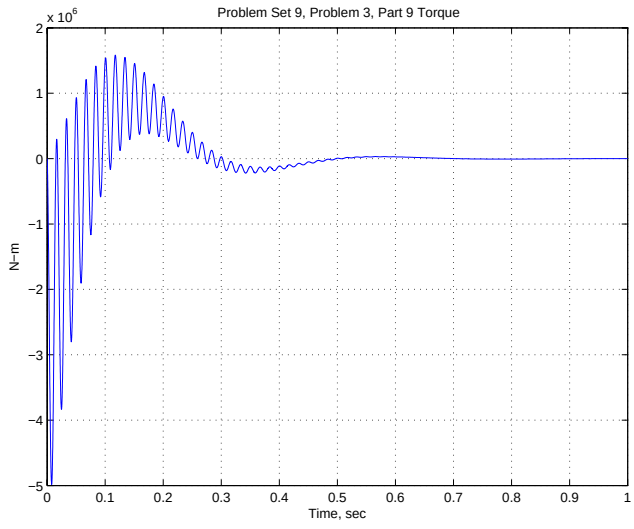

Figure 8: Synchronizing out of Phase: Torque

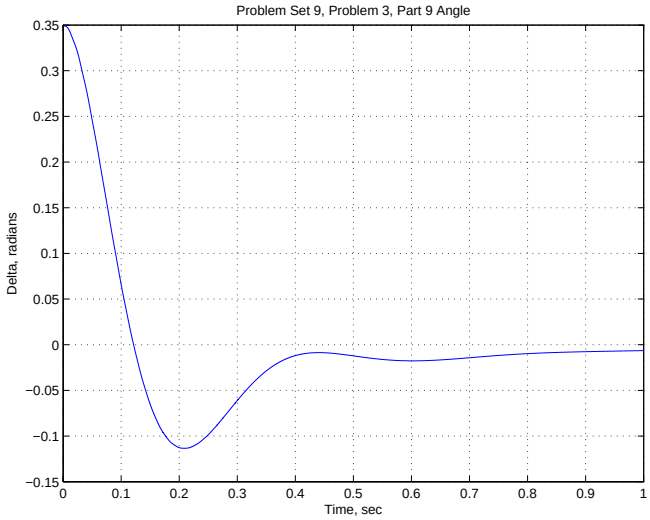

Figure 9: Synchronizing out of Phase: Angle

```
% 6.685 Problem Set 9, Problem 1, basic calculator
ps9params           % this gets the parameters
                    % the following two calls generate the internal stuff

[xad xkdl xfl rkd rf] = mi(xd, xdp, xdpp, tdop, tdopp, xl, omz);
[xaq xkql rkq] = miq(xq, xqpp, tqopp, xl, omz);
Zb = Vb^2/Pb;
ra = .5*(xdpp+xqpp)/(omz*ta);
fprintf('6.685 Problem Set 9, Problem 3, Part 2: Basics\n');
fprintf('Parameter            Per Unit      Ohms\n');
fprintf('  xad            %10.3g      %10.3g\n',xad,xad*Zb);
fprintf('  xaq            %10.3g      %10.3g\n',xaq,xaq*Zb);
fprintf('  xfl            %10.3g      %10.3g\n',xfl,xfl*Zb);
fprintf('  xkdl           %10.3g      %10.3g\n',xkdl,xkdl*Zb);
fprintf('  xkql           %10.3g      %10.3g\n',xkql,xkql*Zb);
fprintf('  rf             %10.3g      %10.3g\n',rf,rf*Zb);
fprintf('  rkd            %10.3g      %10.3g\n',rkd,rkd*Zb);
fprintf('  rkq            %10.3g      %10.3g\n',rkq,rkq*Zb);
fprintf('  ra             %10.3g      %10.3g\n',ra, ra*Zb);

------------------------
% 6.685 Fall 2013
% Parameters for Problem Set 9, Problem 3:
% This is ps9params.m

xd=2.0;           % synchronous d- axis reactance
xq=1.8;           % synchronous q- axis reactance
xdp = .4;        % transient (d-axis) reactance
xdpp = .2;       % subtransient d- axis reactance
xqpp = .2;       % subtransient q- axis reactance
tdop = 5;         % transient (open circuit) time constant
tdopp = .2;       % subtransient d- axis time constant
tqopp = .5;       % subtransient q- axis time constant
xl = .1;          % armature leakage reactance
omz = 60*2*pi;    % base frequency
H = 3;            % rotor inertia constant
ta = .1;          % armature time constant
psi=0;            % power factor angle
Pb = 600e6;       % base power (rating)
Vb = 22e3;        % base voltage (rating)
p=1;              % number of pole pairs

------------------------
```

```
% 6.685 Fall 2011
% Problem Set 9, Problem 3, parts 3 and 4

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf
ps9params
t0 = 0;
tf = .5;
[xad xkdl xfl rkd rf] = mi(xd, xdp, xdpp, tdop, tdopp, xl, omz);
xkd = xad + xkdl;
xf = xad + xfl;
xmd = [xd xad xad; xad xkd xad; xad xad xf];
ymd = inv(xmd);
[xaq xkql rkq] = miq(xq, xqpp, tqopp, xl, omz);
xkq = xaq + xkql;
xmq = [xq xaq; xaq xkq];
ymq = inv(xmq);
ydd = ymd(1,1);
ydk = ymd(1,2);
ykd = ymd(2,2);
ydf = ymd(1,3);
yff = ymd(3,3);
ykf = ymd(2,3);
yqq = ymq(1,1);
yqk = ymq(1,2);
ykq = ymq(2,2);
ra = .5*(xdpp+xqpp)/(omz*ta);
vf = rf/xad;
% flux vector is [psid psiq psikd psikq psif]
psi0 = [1 0 1 0 1+xfl/xad];
dt = (tf-t0)/1024;
time = t0:dt:tf;
[t,psi] = ode23('sf',time, psi0);

id = ydd .* psi(:,1) + ydk .* psi(:,3) + ydf .* psi(:,5);
iq = yqq .* psi(:,2) + yqk .* psi(:,4);

a = 2*pi/3;
ia = id .* cos (omz .* t) - iq .* sin (omz .* t);
ib = id .* cos (omz .* t - a) - iq .* sin (omz .* t - a);
ic = id .* cos (omz .* t + a) - iq .* sin (omz .* t + a);
iff = ydf .* psi(:,1) + ykf .* psi(:,3) + yff .* psi(:,5);
ikd = ydk .* psi(:,1) + ykd .* psi(:,2) + ykf .* psi(:,3);

% here is the fault current done by 'classical' calculation
tdp = tdop*xdp/xd;
```

```
tdpp = tdopp*xdpp/xdp;
iac = (1/xdpp).* exp(-t ./ ta) - (1/xd + (1/xdp-1/xd) .* exp(-t ./ tdp) ...
   + (1/xdpp - 1/xdp) .* exp(-t ./ tdpp)) .* cos(omz .* t);

figure(1)
clf
plot(t, ia, t, iac)
title('Fault Current: Simulated and Classical');
ylabel('ia, per unit');
xlabel('Time, s');
grid on

figure(2)
clf
plot(t, ia-iac)
title('Fault Current: Difference between Simulated and Classical')
ylabel('per-unit')
xlabel('Time, s')
grid on

-----------------------------
% model elements from terminal parameters
function [xad, xkd, xf, rkd, rf] = mi(xd, xdp, xdpp, tdop, tdopp, xl, omz)
xad = xd - xl;
xf = xad * (xdp - xl) / (xad - xdp + xl);
xkd = 1/(1/(xdpp-xl) - 1/xad - 1/xf);
rf = (xf+xad)/(omz * tdop);
rkd = (xkd + xad*xf/(xad+xf))/(omz*tdopp);

-----------------------------
% model elements from terminal parameters

function [xaq, xkq, rkq] = miq(xq, xqpp, tqopp, xl, omz)
xaq = xq - xl;
xkq = xaq*(xqpp - xl)/(xaq-xqpp+xl);
rkq = (xaq+xkq)/(omz*tqopp);

-----------------------------
% 6.685 Fall 2011
% Problem Set 9, Problem 3, parts 3 and 4

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf
ps9params
t0 = 0;
tf = .5;
```

```
[xad xkdl xfl rkd rf] = mi(xd, xdp, xdpp, tdop, tdopp, xl, omz);
xkd = xad + xkdl;
xf = xad + xfl;
xmd = [xd xad xad; xad xkd xad; xad xad xf];
ymd = inv(xmd);
[xaq xkql rkq] = miq(xq, xqpp, tqopp, xl, omz);
xkq = xaq + xkql;
xmq = [xq xaq; xaq xkq];
ymq = inv(xmq);
ydd = ymd(1,1);
ydk = ymd(1,2);
ykd = ymd(2,2);
ydf = ymd(1,3);
yff = ymd(3,3);
ykf = ymd(2,3);
yqq = ymq(1,1);
yqk = ymq(1,2);
ykq = ymq(2,2);
ra = .5*(xdpp+xqpp)/(omz*ta);
vf = rf/xad;
% flux vector is [psid psiq psikd psikq psif]
psi0 = [1 0 1 0 1+xfl/xad];
dt = (tf-t0)/1024;
time = t0:dt:tf;
[t,psi] = ode23('sf',time, psi0);

id = ydd .* psi(:,1) + ydk .* psi(:,3) + ydf .* psi(:,5);
iq = yqq .* psi(:,2) + yqk .* psi(:,4);

a = 2*pi/3;
ia = id .* cos (omz .* t) - iq .* sin (omz .* t);
ib = id .* cos (omz .* t - a) - iq .* sin (omz .* t - a);
ic = id .* cos (omz .* t + a) - iq .* sin (omz .* t + a);
iff = ydf .* psi(:,1) + ykf .* psi(:,3) + yff .* psi(:,5);
ikd = ydk .* psi(:,1) + ykd .* psi(:,2) + ykf .* psi(:,3);

% here is the fault current done by 'classical' calculation
tdp = tdop*xdp/xd;
tdpp = tdopp*xdpp/xdp;
iac = (1/xdpp).* exp(-t ./ ta) - (1/xd + (1/xdp-1/xd) .* exp(-t ./ tdp) ...
   + (1/xdpp - 1/xdp) .* exp(-t ./ tdpp)) .* cos(omz .* t);

figure(1)
clf
plot(t, ia, t, iac)
```

```
title('Fault Current: Simulated and Classical');
ylabel('ia, per unit');
xlabel('Time, s');
grid on

figure(2)
clf
plot(t, ia-iac)
title('Fault Current: Difference between Simulated and Classical')
ylabel('per-unit')
xlabel('Time, s')
grid on

----------------------------
function dpsi = sf(t, psi);

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf

psid = psi(1);
psiq = psi(2);
psikd = psi(3);
psikq = psi(4);
psif = psi(5);

id = ydd * psid + ydk * psikd + ydf * psif;
ikd = ydk * psid + ykd * psikd + ykf * psif;
iff = ydf * psid + ykf * psikd + yff * psif;

iq = yqq * psiq + yqk * psikq;
ikq = yqk * psiq + ykq * psikq;

psidot1 = omz*psiq - omz*ra*id;
psidot2 = -omz*psid - omz*ra*iq;
psidot3 = -omz*rkd*ikd;
psidot4 = -omz*rkq*ikq;
psidot5 = omz*vf - omz*rf*iff;

dpsi = [psidot1 psidot2 psidot3 psidot4 psidot5]';

----------------------------
% 6.685 Electric Machines
% Problem Set 9 Problem 3, part 5: Open Circuit voltage

ps9params
```

```
% Establish operating point

delt = atan((xq*cos(psi))/(1+sin(psi)));
psid = cos(delt);
psiq = - sin(delt);
id = sin(delt+psi);
iq = cos(delt+psi);

eqopp = psid + xdpp * id;
eqop = psid + xdp * id;
edopp = psiq - xqpp * iq;
eaf = psid + xd * id;

t = .01:.01:5;
eqpp = (eqopp-eqop) .* exp(-t ./ tdopp) ...
      + (eqop - eaf) .* exp(-t ./ tdop) + eaf;
edpp = edopp .* exp(-t ./ tqopp);

figure(3)
clf
subplot 211
plot(t,eqpp);
title('Opening: Voltages Behind Subtransient Reactances');
ylabel('eqpp, per-unit');
grid on
subplot 212
plot(t, edpp);
xlabel('Time, s');
ylabel('edpp, per-unit');
grid on

fprintf('Problem 9.3, Part 7\n')
fprintf('Torque Angle     = %10.4g  Power Factor Angle = %10.4g\n', delt, psi)
fprintf('D axis flux psid = %10.4g  Q axis flux psiq   = %10.4g\n', psid, psiq)
fprintf('D axis current Id= %10.4g  Q axis current Iq  = %10.4g\n', id, iq)
fprintf('V behind xdp, eqp= %10.4g  V behind sync x eaf= %10.4g\n', eqop, eaf)
fprintf('V beh xdpp, eqpp = %10.4g  V behind xqpp edpp = %10.4g\n', eqopp, edopp)

--------------------------
% 6.685 Fall 2011
% Problem Set 9, Problem 3 part 6: torque-angle curves
ps9params

% Establish operating point (rated load, unity pf)
% We are assuming a generator and the associated odd
```

```
% sign conventions

delt = atan((xq*cos(psi))/(1+sin(psi)));  % torque angle
psid = cos(delt);        % fluxes on d- and q- axes
psiq = - sin(delt);
id = sin(delt+psi);      % these are generator convention currents!
iq = cos(delt+psi);

eqop = psid + xdp * id;  % voltage behind transient reactance
eaf = psid + xd * id;    % voltage behind synchronous reactance

d = 0:pi/100:pi;         % plot over this range of angle

tesync = (eaf/xd) .* sin(d) + .5*(1/xq - 1/xd) .* sin(2 .* d);
tetrans = (eqop/xdp) .* sin(d) + .5*(1/xq - 1/xdp) .* sin(2 .* d);
tefict = (eqop/xdp) .* sin(d);

figure(4)
clf
plot(d, tesync, d, tetrans, d, tefict)
title('Torque-Angle Curves');
xlabel('Radians');
ylabel('Per-Unit');
grid on
legend('Synchronous', 'Transient', 'Phony')

----------------------------
% 6.685 homework set 9, Problem 3, part 8
% initial conditions for steady operation

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf H Tm yado ydo yfo xkq

ps9params                                % get the starting point parameters
p9_2                                      % and this does the early part
                                         % note some things will get reset
                                         % (see immediately below for
                                         % example)
pf = .85;                                % power factor
vt = 1;                                  % terminal voltage
ia = 1;                                  % terminal current magnitude
psi = acos(pf);                          % power factor angle
i_a = ia*(pf - j*sin(psi));              % as a complex vector
v_i = vt + ra*i_a;                       % this is the internal voltage vector
vi = abs(v_i);                           % this is the absolute value of internal voltage
ai = angle(v_i);                         % and this is the angle of the internal voltage
```

```
e_1 = v_i + j*i_a*xq;                    % the voltage that defines the q axis
e1 = abs(e_1);
delta = angle(e_1);                      % and this is the phase angle
id = ia*sin(delta + psi);                % this is direct axis current
iq = ia*cos(delta + psi);                % and this is quadrature axis current
vq = vi*cos(delta - ai);                 % q axis component of that internal voltage
vd = vi*sin(delta - ai);                 % d axis component
eaf = vq + id*xd;                        % voltage behind synchronous reactance
i_f = eaf/xad;                           % steady field current
vf = i_f*rf;                             % exciter voltage
psid0 = vq;                              % fluxes are -90 degrees rotated from internal voltag
psiq0 = -vd;
psikd0 = psid0 + id*xl;                   % same as flux on the magnetizing inductance
psikq0 = psiq0 + iq*xl;                   % this is the same too
psif0 = psikd0 + xfl*i_f;                 % flux initial condition

Tm = psid0*iq-psiq0*id;                   % generating torque

fprintf('Problem Set 11, Steady State Operation\n')
fprintf('psid = %g\n', psid0)
fprintf('psiq = %g\n', psiq0)
fprintf('psikd = %g\n', psikd0)
fprintf('psikq = %g\n', psikq0)
fprintf('psif = %g\n', psif0)
fprintf('delta = %g\n', delta)
fprintf('Torque = %g\n', Tm)
fprintf('V Field = %g\n', vf)

% now precompute some stuff used by the simulation
% direct axis
xkd = xad + xkdl;
xf = xad + xfl;
xmd = [xd xad xad; xad xkd xad; xad xad xf];
ymd = inv(xmd);
% quadrature axis
xkq = xaq + xkql;
xmq = [xq xaq; xaq xkq];
ymq = inv(xmq);
ydd = ymd(1,1);
ydk = ymd(1,2);
ykd = ymd(2,2);
ydf = ymd(1,3);
yff = ymd(3,3);
ykf = ymd(2,3);
yqq = ymq(1,1);
```

```
yqk = ymq(1,2);
ykq = ymq(2,2);
xdo = [xkd xad; xad xf];
ydop = inv(xdo);
ydo = ydop(1,1);
yado = ydop(1,2);
yfo = ydop(2,2);
% Establish operating point to start

% initial state vector is [psikd psikq psif om delt]
x0 = [psid0 psiq0 psikd0 psikq0 psif0 omz delta];

timeo = 0:.00001:.1;

%[tb,xb] = ode45('steadydelt', timeo, x0, newopts);  If higher fidelity is
%needed
[tb,xb] = ode45('steadydelt', timeo, x0);

figure(1)
subplot 311
plot(tb, xb(:,7));
ylabel('angle')
ma = max(xb(:,7));
na = min(xb(:,7));
axis([0 .1 .9*na 1.1*ma]);
subplot 312
plot(tb, xb(:,6));
ylabel('speed')
ma = max(xb(:,6));
na = min(xb(:,6));
axis([0 .1 .8*na 1.2*ma]);
subplot 313
plot(tb, xb(:,5), tb, xb(:,4), tb, xb(:,3), tb, xb(:,2), tb, xb(:,1))
ylabel('fluxes')
axis([0 .1 -1.5 1.5]);
legend('Field', 'q damper', 'd damper', 'q armature', 'd armature')
xlabel('Time, sec')

--------------------------
function dpsi = steadydelt(t, psi);

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf H Tm yado ydo yfo xkq

psid = psi(1);
psiq = psi(2);
```

```
psikd = psi(3);
psikq = psi(4);
psif = psi(5);
om = psi(6);
delta = psi(7);

id = ydd * psid + ydk * psikd + ydf * psif;
ikd = ydk * psid + ykd * psikd + ykf * psif;
iff = ydf * psid + ykf * psikd + yff * psif;

iq = yqq * psiq + yqk * psikq;
ikq = yqk * psiq + ykq * psikq;

te = psid*iq - psiq*id;

psidot1 = om*psiq - omz*ra*id + omz*sin(delta);
psidot2 = -om*psid - omz*ra*iq + omz*cos(delta);
psidot3 = -omz*rkd*ikd;
psidot4 = -omz*rkq*ikq;
psidot5 = omz*vf - omz*rf*iff;
omdot = (omz/(2*H)) * (Tm+te);
ddot = om-omz;

dpsi = [psidot1 psidot2 psidot3 psidot4 psidot5 omdot ddot]';

------------------------
% 6.685 Fall 2011
% synchronizing out of phase by 20 degrees = pi/9;

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf H Tm yado ydo yfo xkq

ps9params                  % to get stuff into the environment
p9_2                       % some more stuff
t0 = 0;
tf = 1.0;
Npoints = 10000;
[xad xkdl xfl rkd rf] = mi(xd, xdp, xdpp, tdop, tdopp, xl, omz);
xkd = xad + xkdl;
xf = xad + xfl;
xmd = [xd xad xad; xad xkd xad; xad xad xf];
ymd = inv(xmd);
[xaq xkql rkq] = miq(xq, xqpp, tqopp, xl, omz);
xkq = xaq + xkql;
xmq = [xq xaq; xaq xkq];
ymq = inv(xmq);
```

```
ydd = ymd(1,1);
ydk = ymd(1,2);
ykd = ymd(2,2);
ydf = ymd(1,3);
yff = ymd(3,3);
ykf = ymd(2,3);
yqq = ymq(1,1);
yqk = ymq(1,2);
ykq = ymq(2,2);
ra = .5*(xdpp+xqpp)/(omz*ta);
vf = rf/xad;
% flux vector is [psid psiq psikd psikq psif omz deltz]
deltaz=pi/9;                            % 20 degrees!
psi0 = [1 0 1 0 1+xfl/xad omz deltaz];
Tm = 0;
dt = (tf-t0)/Npoints;
time = t0:dt:tf;
[t,psi] = ode23('steadydelt',time, psi0);

psid = psi(:,1);
psiq = psi(:,2);
psikd = psi(:,3);
psikq = psi(:,4);
psif = psi(:,5);
om = psi(:,6);
delt = psi(:,7);

id = ydd .* psid + ydk .* psikd + ydf .* psif;
iq = yqq .* psiq + yqk .* psikq;

a = 2*pi/3;
ia = id .* cos (omz .* t + delt) - iq .* sin (omz .* t + delt);
ib = id .* cos (omz .* t - a+ delt) - iq .* sin (omz .* t - a + delt);
ic = id .* cos (omz .* t + a+ delt) - iq .* sin (omz .* t + a + delt);

iff = ydf .* psid + ykf .* psikd + yff .* psif;
ikd = ydk .* psid + ykd .* psikd + ykf .* psif;

te = psid .* iq - psiq .* id;

% now convert to ordinary variables

IB = Pb/(sqrt(3)*Vb);
Tb = p*Pb/omz;
```

```
Ia = IB .* ia;
Ib = IB .* ib;
Ic = IB .* ic;
Torque = te .* Tb;

figure(1)
plot(t, Ia, t, Ib, t, Ic);
title('Problem Set 9, Problem 3, Part 9 Current')
ylabel('Amperes')
xlabel('Time, sec')
legend('A', 'B', 'C')
grid on

figure(2)
plot(t, Torque)
title('Problem Set 9, Problem 3, Part 9 Torque')
ylabel('N-m')
xlabel('Time, sec')
grid on

figure(3)
clf
plot(t, delt)
title('Problem Set 9, Problem 3, Part 9 Angle')
ylabel('Delta, radians')
xlabel('Time, sec')
grid on

figure(4)
clf
plot(t, psi(:,6)-omz)
title('Problem Set 9, Problem 3, Part 9 Differential Speed')
ylabel('Slip, radians/second')
xlabel('time, sec')
grid on
```

MIT OpenCourseWare
http://ocw.mit.edu

6.685 Electric Machines
Fall 2013

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.