Massachusetts Institute of Technology

Department of Electrical Engineering and Computer Science

6.691 Seminar in Advanced Electric Power Systems

Problem Set 5 Solutions April 30, 2006

Much of the story here is told by the attached scripts. The basic model to be used is the seventh-order description of the synchronous machine:

$$\begin{aligned}
\frac{d\psi_d}{dt} &= \omega_0 v_d + \omega\psi_q - \omega_0 r_a i_d \\
\frac{d\psi_q}{dt} &= \omega_0 v_q - \omega\psi_d - \omega_0 r_a i_q \\
\frac{d\psi_{kd}}{dt} &= -\omega_0 r_{kd} i_{kd} \\
\frac{d\psi_{kq}}{dt} &= -\omega_0 r_{kq} i_{kq} \\
\frac{d\psi_f}{dt} &= \omega_0 v_f - \omega_0 r_f i_f \\
\frac{d\omega}{dt} &= \frac{\omega_0}{2H}(T_e + T_m) \\
\frac{d\delta}{dt} &= \omega - \omega_0
\end{aligned}$$

and, of course,

$$T_e = \psi_d i_q - \psi_q i_d$$

Simulation of this model requires finding the currents in terms of the fluxes (which are the state variables). For the d-axis this is:

$$\begin{bmatrix} i_d \\ i_{kd} \\ i_f \end{bmatrix} = \begin{bmatrix} y_{dd} & y_{dk} & y_{df} \\ y_{kd} & y_{kk} & y_{kf} \\ y_{fd} & y_{fk} & y_{ff} \end{bmatrix} \begin{bmatrix} \psi_d \\ \psi_{kd} \\ \psi_f \end{bmatrix}$$

Where:

$$\begin{bmatrix} y_{dd} & y_{dk} & y_{df} \\ y_{kd} & y_{kk} & y_{kf} \\ y_{fd} & y_{fk} & y_{ff} \end{bmatrix} = \begin{bmatrix} x_d & x_{ad} & x_{ad} \\ x_{ad} & x_{kd} & x_{ad} \\ x_{ad} & x_{ad} & x_f \end{bmatrix}^{-1}$$

Similar expressions are used for the q-axis. Simulation is straightforward: I used the Matlab function `ode23`. See the appended scripts.

1 Terminal Fault

For the terminal fault we need to start the simulation with initial conditions as found in Problem Set 4. It is common to allow the simulation to run for a short time to see if, indeed, you have found the correct initial conditions. Here they are:

```
>>  ffrated
Initial  Conditions  for  Simulation:
psid0  =  0.773476    psiq0  =  -0.641583
psiad0  =  0.868191   psiaq0  =  -0.609504
psif0  =  1.43963   deltz  =  0.68942
Generator  id  =  0.94715   iq  =  0.320792
Motor      id  =  -0.94715   iq  =  -0.320792
omz  =  376.991   torque  =  -0.8558

This  should  look  familiar.
```

Some graphical results are shown in the next few pictures. Note I have run the simulation for only about 1/2 second past the fault, so some detail of currents are visible.To reconstruct phase A current the inverse Park's transform was used, with the machine angle being:

$$\theta = \omega_0 t + \delta$$

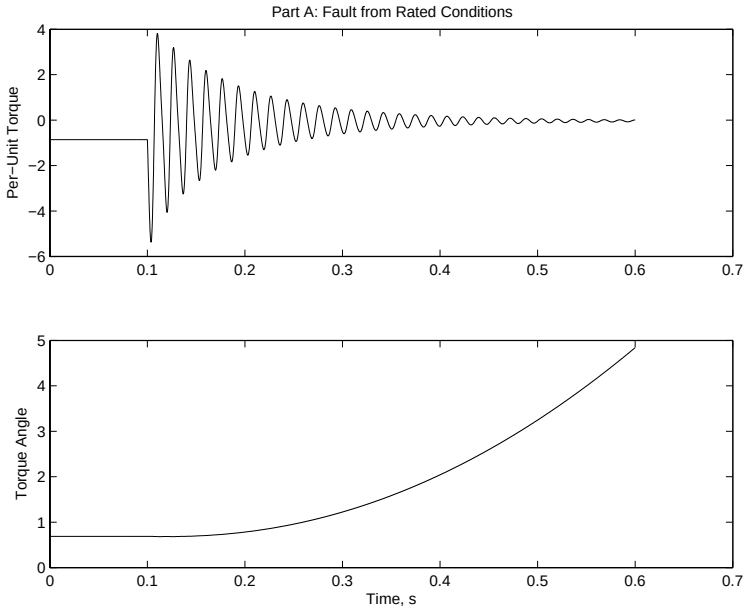

Figure 1: Fault Torque and Torque Angle

2 Transient Stability

In this part of the problem an *open* circuit was specified. Opening the machine sets $i_d = i_q = 0$ and this reduces the order of the machine model by two. A straightforward way of handling this is to ignore the first two state equations (in ψ_d and ψ_q) and, eliminating these two variables in the current equations to find:

$$i_{kd} \quad = \quad \left(y_{kk} - \frac{y_{dk}^2}{y_{dd}} \right) \psi_{kd} + \left(y_{kf} - \frac{y_{dk} y_{kf}}{y_d d} \right) \psi_f$$

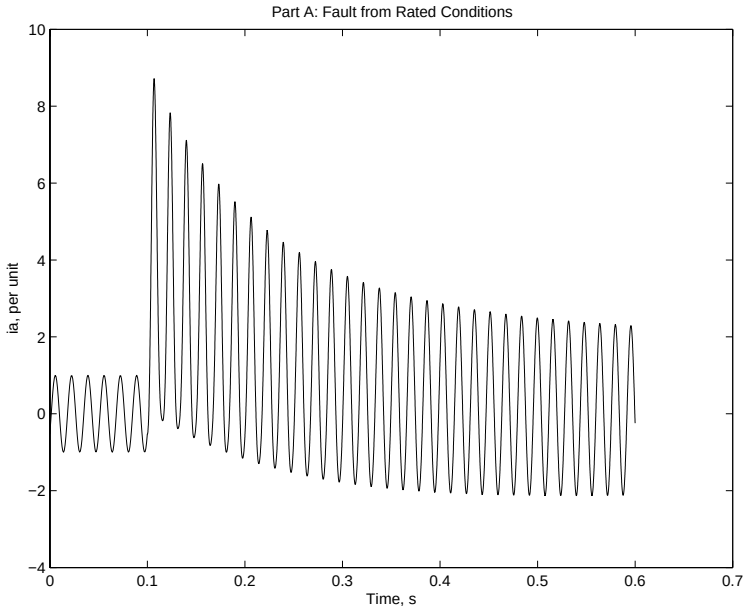

Figure 2: Phase A Current

$$
\begin{aligned}
i_f &= \left(y_{kf} - \frac{y_{df}y_{dk}}{y_d d}\right)\psi_{kd} + \left(y_{ff} - \frac{y_{df}^2}{y_{dd}}\right)\psi_f \\
i_{kq} &= \left(y_{kq} - \frac{y_{qk}^2}{y_{qq}}\right)\psi_{kq}
\end{aligned}
$$

Note that we are ignoring whatever transient was required to drive terminal current to zero: presumably a big voltage imposed by the opening switch. When the machine is reclosed, initial values for the two stator fluxes ψ_d and ψ_q can be established by using a voltage divider relationship between ψ_{kd} and ψ_f (for the direct axis) and ψ_{kq} (for the quadrature axis).

To start the simulation, we use the same initial conditions as in the first part. By trial and error we find a critical clearing time somewhere between 300 and 301 mS. (I didn't go finer than that and it makes no sense to do so since the time between individual current zeros of the 60 Hz waveform is a bit less than 3 mS. Figure 4 shows the transient at 301 mS reclosing time. Note the machine slips four poles and recovers. This would not be good for the generator. Figure 5 shows a swing that is just barely stable.

3 Bad Mistake

Synchronization 120 degrees out of phase is thought to be about the worst thing you can do to a generator. This could happen were the voltage sensors to be interchanged in what turns out to be a simple way. The simulation is started with mechanical torque equal to zero, direct axis stator and damper fluxes equal to unity, quadrature axis fluxes equal to zero and field flux calculated from steady conditions. The starting angle δ is set to 120 degrees. The results are shown in Figures 6, 7 and 8.

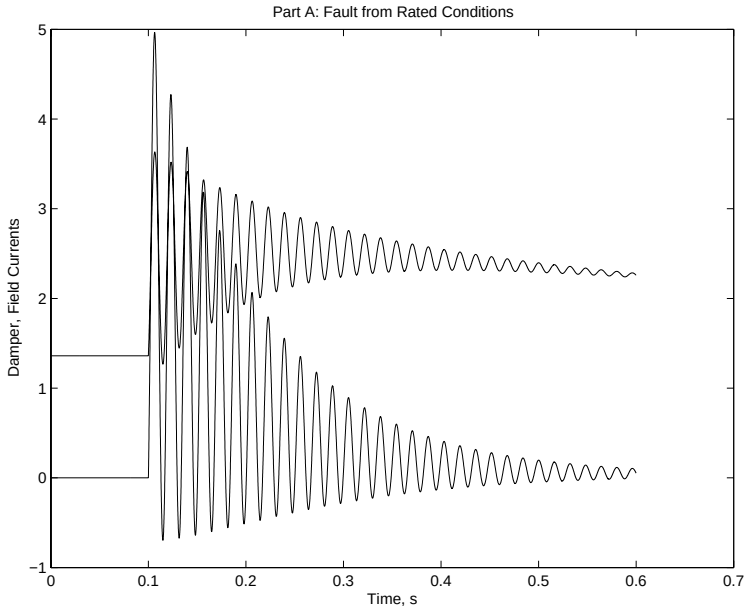

Figure 3: Damper and Field Currents

4 Small Signal Variation and Eigenvalues

The equations describing synchronous machine interaction may be linearized about the steady-state operating point. The equivalent set for the first-order variations is:

$$\begin{aligned}
\frac{d\psi_{d1}}{dt} &= \omega_0 V \cos\delta_0 \delta_1 + \omega_0 \psi_{q1} + \omega_1 \psi_{q0} - \omega_0 r_a i_{d1} \\
\frac{d\psi_{q1}}{dt} &= -\omega_0 V \sin\delta_0 \delta_1 - \omega_0 \psi_{d1} - \omega_1 \psi_{d0} - \omega_0 r_a i_{q1} \\
\frac{d\psi_{kd1}}{dt} &= -\omega_0 r_{kd} i_{kd1} \\
\frac{d\psi_{kq1}}{dt} &= -\omega_0 r_{kq} i_{kq1} \\
\frac{d\psi_{f1}}{dt} &= -\omega_0 r_f i_{f1} \\
\frac{d\omega_1}{dt} &= \frac{\omega_0}{2H}\left(T_{e1} + T_{m1}\right) \\
\frac{d\delta_1}{dt} &= \omega_1
\end{aligned}$$

$$T_{e1} = \psi_{d0} i_{q1} + \psi_{d1} i_{q0} - \psi_{q0} i_{d1} - \psi_{q1} i_{d0}$$

and for the purposes of this problem set we assume $T_{m1} = 0$.

The problem set describes how to find currents in terms of fluxes, and once those are filled in we have a seventh-order linear system of the form:

$$\underline{\dot{S}} = \underline{\underline{M}}\,\underline{S}$$

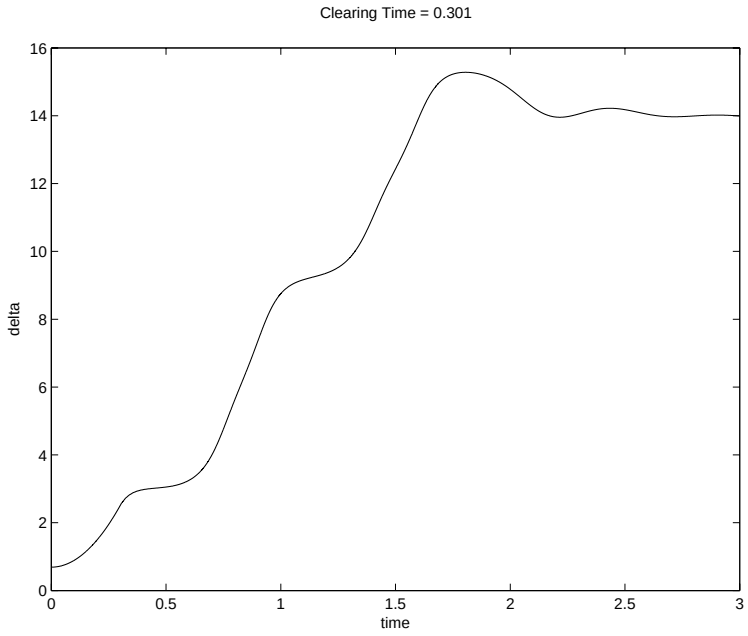

Figure 4: Unstable Swing

The details are shown in the MATLAB script which is appended. Note that I have pulled a small scale fraud here: While the machine is indeed driving a voltage source through some line reactance, I have assumed that the power factor in question is at the load end, not the machine. This will have a minor effect on the steady state operating point and through it the eigenvalues found.

To further illustrate the process and what is going on, note that the differential equation:

$$\frac{2H}{\omega_0}\frac{d^2\delta}{dt^2} + B\frac{d\delta}{dt} + T\delta = 0$$

is really the equivalent of (in fact is derived from) two first order differential equations:

$$\begin{aligned} \frac{2H}{\omega_0}\frac{d\omega}{dt} &= -B\omega - T\delta \\ \frac{d\delta}{dt} &= \omega \end{aligned}$$

The equivalent 'transition matrix' for this system is:

$$\underline{\underline{M}} = \begin{bmatrix} -B & -\frac{\omega_0}{2H}T \\ 1 & 0 \end{bmatrix}$$

It takes a little fiddling around to get the right parameters B and T, but eventually we find damping time constant and swing frequency to match the actual machine.

Results of the eigenvalue analysis are:

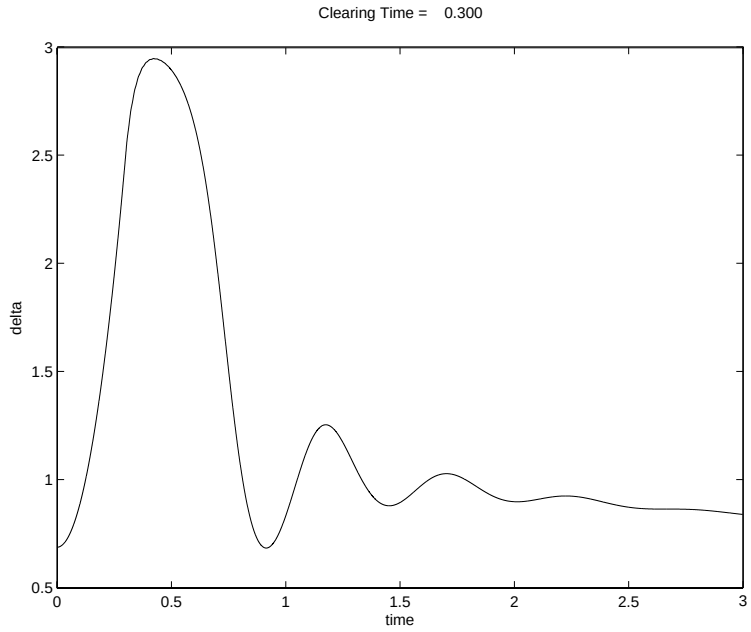

Figure 5: Stable Swing

```
>>  smeigs
Eigenvalues  of  a  Synchronous  Machine

S  =

   1.0e+02  *

  -0.0667  +  3.7625i
  -0.0667  - 3.7625i
  -0.9597
  -0.0122  +  0.0955i
  -0.0122  - 0.0955i
  -0.0739
  -0.0049

Eigenvalues  of  equivalent  second  order  model

S2  =

  -1.2200  +  9.5575i
  -1.2200  - 9.5575i
>>
```

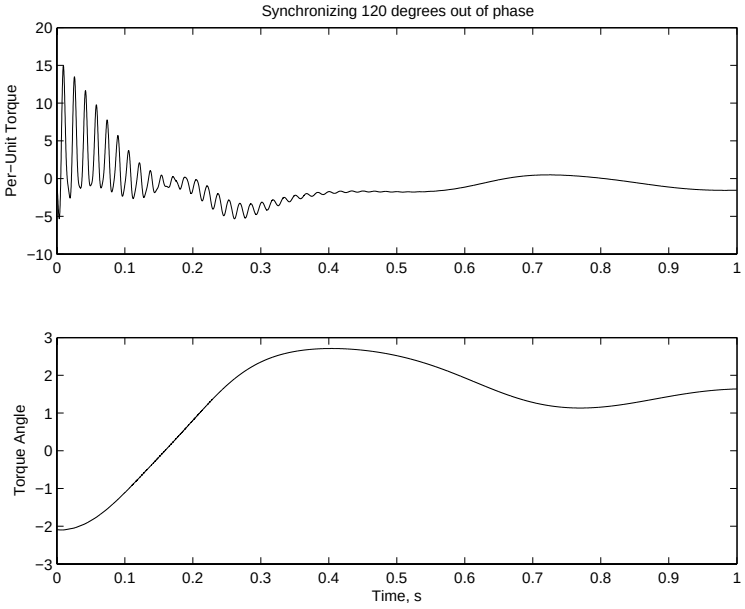

Figure 6: Out of Phase Synchronization

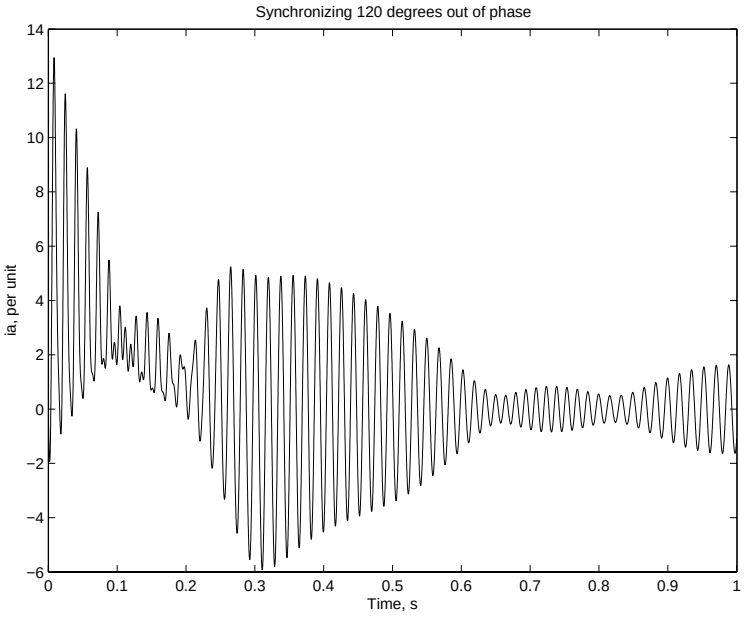

Figure 7: Out of Phase Synchronization

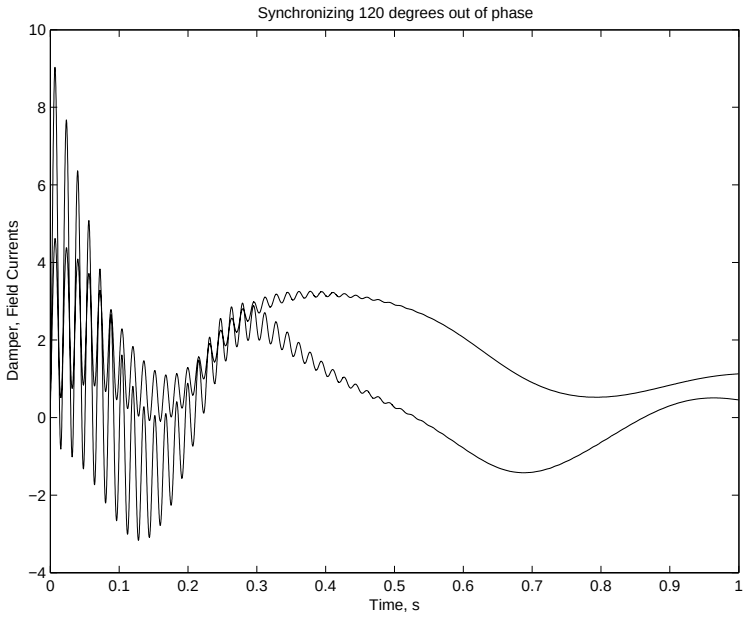

Figure 8: Out of Phase Synchronization

```
% 6.691 Problem Set 5 Fault from rated operation

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf H Tm Vt
t0 = 0;
ts = .1;       % run for a short time to confirm initial condiions
tf =.6;
H=4;
omz = 2*pi*60;
xal = .1;
xad = 2.1;
xaq = 1.9;
xkdl = .183;
xkql = .128;
xfl = .42;
rkd = .00706;
rkq = .079;
rf = .00134;
ra = .0058;
xd = xad + xal;
xq = xaq + xal;
xkd = xad + xkdl;
xkq = xad + xkql;
xf = xad + xfl;
vt = 1.0;
it = 1.0;
pf = 0.85;
% a couple of little things
xd = xad+xal;
xq = xaq+xal;


% first get initial conditions
ia = it*(pf - j * sqrt(1 - pf^2));       % this is the complex current
e1 = vt + ia*ra+j*xq*ia;       % this is on the q- axis
va = vt + ia*ra;               % this is voltage 'inside' resistance
tha = angle(va);               % pretty small angle
delta = angle(e1);             % this is the torque angle
psi = angle(ia);               % and this is power factor angle
idg = it*sin(delta - psi);         % since the psi we calculate is negative
iqg = it*cos(delta - psi);
ea = abs(e1) + (xd - xq) * idg;     % voltage behind synchronous reactance
vq = abs(va)*cos(delta - tha);       % voltage on q axis is d axis flux
vd = abs(va)*sin(delta - tha);
psid0 = vq;
psiq0 = - vd;
```

```
psiad0 = psid0 + xal*idg;        % air-gap flux
psiaq0 = psiq0 + xal*iqg;        % these will be damper fluxes too
i_f = ea/xad;                   % field current
psif0 = psiad0 + xfl*i_f;        % flux at field terminal
vf = i_f*rf;
deltz = delta;

xmd = [xd xad xad; xad xkd xad; xad xad xf];
ymd = inv(xmd);
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);
% initial state vector is [psid psiq psikd psikq psif omz deltz]
psi0 = [psid0 psiq0 psiad0 psiaq0 psif0 omz deltz];
fprintf('Initial Conditions for Simulation:\n')
fprintf('psid0 = %g   psiq0 = %g\n', psid0, psiq0)
fprintf('psiad0 = %g   psiaq0 = %g\n', psiad0, psiaq0)
fprintf('psif0 = %g   deltz = %g\n', psif0, deltz)
idm = ydd*psid0 + ydk*psiad0 + ydf*psif0;
iqm = yqq*psiq0 + yqk*psiaq0;
Trq = psid0*iqm - psiq0*idm;          % electrical torque
Tm = -Trq;
fprintf('Generator id = %g   iq = %g\n', idg, iqg)
fprintf('Motor     id = %g   iq = %g\n', idm, iqm)
fprintf('omz = %g   torque = %g\n', omz, Trq)
% initial time
dts = (ts - t0)/128;
time1 = t0:dts:ts;
Vt = vt;                              % starts out ok
[t1, psi1] = ode23('sf', time1, psi0);
dt = (tf-ts)/8192;
time2 = ts:dt:tf;
Vt = 0;                               % but now gets faulted
[t2,psi2] = ode23('sf',time2, psi0);
time = [time1 time2];
psi = [psi1; psi2];
```

```
id = ydd .* psi(:,1) + ydk .* psi(:,3) + ydf .* psi(:,5);
iq = yqq .* psi(:,2) + yqk .* psi(:,4);
Te = psi(:,1) .* iq - psi(:,2) .* id;
delt = psi(:,7);
om = psi(:,6);

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

iff = ydf .* psi(:,1) + ykf .* psi(:,3) + yff .* psi(:,5);
ikd = ydk .* psi(:,1) + ykd .* psi(:,3) + ykf .* psi(:,5);

figure(1)
clf
plot(time, ia)
title('Part A: Fault from Rated Conditions');
ylabel('ia, per unit');
xlabel('Time, s');

figure(2)
clf
subplot 211
plot(time, Te)
     title('Part A: Fault from Rated Conditions')
     ylabel('Per-Unit Torque')
subplot 212
plot(time, delt)
     ylabel('Torque Angle')
     xlabel('Time, s')

     figure(3)
     plot(time, ikd, time, iff)
     title('Part A: Fault from Rated Conditions');
ylabel('Damper, Field Currents')
xlabel('Time, s')
```

```
% 6.691 Problem Set 5 Critical Clearing Time

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf H Tm Vt
t0 = 0;
ts = .300;    % This will be the initial time: machine is open
tf =3;      % and we will carry the fault this long
H=4;
omz = 2*pi*60;
xal = .1;
xad = 2.1;
xaq = 1.9;
xkdl = .183;
xkql = .128;
xfl = .42;
rkd = .00706;
rkq = .079;
rf = .00134;
ra = .0058;
xd = xad + xal;
xq = xaq + xal;
xkd = xad + xkdl;
xkq = xad + xkql;
xf = xad + xfl;
vt = 1.0;
it = 1.0;
pf = 0.85;
% a couple of little things
xd = xad+xal;
xq = xaq+xal;


% first get initial conditions
ia = it*(pf - j * sqrt(1 - pf^2));       % this is the complex current
e1 = vt + ia*ra+j*xq*ia;       % this is on the q- axis
va = vt + ia*ra;               % this is voltage 'inside' resistance
tha = angle(va);               % pretty small angle
delta = angle(e1);             % this is the torque angle
psi = angle(ia);               % and this is power factor angle
idg = it*sin(delta - psi);         % since the psi we calculate is negative
iqg = it*cos(delta - psi);
ea = abs(e1) + (xd - xq) * idg;     % voltage behind synchronous reactance
vq = abs(va)*cos(delta - tha);       % voltage on q axis is d axis flux
vd = abs(va)*sin(delta - tha);
psid0 = vq;
psiq0 = - vd;
```

```
psiad0 = psid0 + xal*idg;        % air-gap flux
psiaq0 = psiq0 + xal*iqg;        % these will be damper fluxes too
i_f = ea/xad;                  % field current
psif0 = psiad0 + xfl*i_f;        % flux at field terminal
vf = i_f*rf;
deltz = delta;

xmd = [xd xad xad; xad xkd xad; xad xad xf];
ymd = inv(xmd);
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);
% calculation of initial conditions
%psi0 = [psid0 psiq0 psiad0 psiaq0 psif0 omz deltz];
fprintf('Initial Conditions for Simulation:\n')
fprintf('psid0 = %g   psiq0 = %g\n', psid0, psiq0)
fprintf('psiad0 = %g   psiaq0 = %g\n', psiad0, psiaq0)
fprintf('psif0 = %g   deltz = %g\n', psif0, deltz)
idm = ydd*psid0 + ydk*psiad0 + ydf*psif0;
iqm = yqq*psiq0 + yqk*psiaq0;
Trq = psid0*iqm - psiq0*idm;          % electrical torque
Tm = -Trq;
fprintf('Generator id = %g   iq = %g\n', idg, iqg)
fprintf('Motor     id = %g   iq = %g\n', idm, iqm)
fprintf('omz = %g   torque = %g\n', omz, Trq)
fprintf('Clearing time = %g\n', ts)
% initial time
dts = (ts - t0)/2048;
time1 = t0:dts:ts;
Vt = vt; % starts out ok
% initial time period is open circuited
psi0 = [psiad0 psiaq0 psif0 omz deltz];
[t1, psis] = ode23('sfopen', time1, psi0);
% now we need to re-assemble things
psid = -(ydk/ydd) .* psis(:,1) - (ydf/ydd) .* psis(:,3);
psid(length(psid));
psiq = - (yqk/yqq) .* psis(:,2);
```

```
psiq(length(psiq));
psi1 = [psid psiq psis];
psi0 = psi1(length(psi1(:,1)),:);
dt = (tf-ts)/8192;
time2 = ts:dt:tf;
Vt = vt;                                % but now gets reclosed
[t2,psi2] = ode23('sf',time2, psi0);
time = [time1 time2];
psi = [psi1; psi2];

id = ydd .* psi(:,1) + ydk .* psi(:,3) + ydf .* psi(:,5);
iq = yqq .* psi(:,2) + yqk .* psi(:,4);
Te = psi(:,1) .* iq - psi(:,2) .* id;
delt = psi(:,7);
om = psi(:,6);

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

iff = ydf .* psi(:,1) + ykf .* psi(:,3) + yff .* psi(:,5);
ikd = ydk .* psi(:,1) + ykd .* psi(:,3) + ykf .* psi(:,5);

titstring = sprintf('Clearing Time = %8.3f\n', ts)
figure(1)
plot(time, delt)
title(titstring)
ylabel('delta')
xlabel('time')
```

```
% 6.691 Problem Set 5 Bad Mistake

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf H
t0 = 0;
tf =1;
H=4;
omz = 2*pi*60;
%deltz=0.1;
deltz = -2*pi/3;
xal = .1;
xad = 2.1;
xaq = 1.9;
xkdl = .183;
xkql = .128;
xfl = .42;
rkd = .00706;
rkq = .079;
rf = .00134;
ra = .0058;
xd = xad + xal;
xq = xaq + xal;
xkd = xad + xkdl;
xkq = xad + xkql;
xf = xad + xfl;
xmd = [xd xad xad; xad xkd xad; xad xad xf];
ymd = inv(xmd);
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);
vf = rf/xad;
% initial state vector is [psid psiq psikd psikq psif omz deltz]
psi0 = [1 0 1 0 1+xfl/xad omz deltz];
dt = (tf-t0)/8192;
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);
Te = psi(:,1) .* iq - psi(:,2) .* id;
delt = psi(:,7);
om = psi(:,6);

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

iff = ydf .* psi(:,1) + ykf .* psi(:,3) + yff .* psi(:,5);
ikd = ydk .* psi(:,1) + ykd .* psi(:,3) + ykf .* psi(:,5);

figure(1)
clf
plot(t, ia)
title('Synchronizing 120 degrees out of phase');
ylabel('ia, per unit');
xlabel('Time, s');

figure(2)
clf
subplot 211
plot(t, Te)
     title('Synchronizing 120 degrees out of phase')
     ylabel('Per-Unit Torque')
subplot 212
plot(t, delt)
     ylabel('Torque Angle')
     xlabel('Time, s')

     figure(3)
     plot(t, ikd, t, iff)
     title('Synchronizing 120 degrees out of phase');
ylabel('Damper, Field Currents')
xlabel('Time, s')
```

```
% 6.691 Problem Set 5
% Eigenvalue Analysis of a synchronous machine
% Machine is assumed to be operating normally against a voltage
% source. No mechanical damping assumed
% Parameter Values
% Example for Synchronous Machine
xad = 2.1;
xaq = 1.9;
xal = 0.2;                    % add 0.1 for line reactance!
xkdl = 0.183;
xkql = 0.128;
xfl = 0.42;
rkd = .00706;
rkq = .079;
rf = .00134;
ra = .0058;
xd = xad+xal;
xq = xaq+xal;
xkd = xad+xkdl;
xkq = xaq+xkql;
xf = xad+xfl;
H = 4;
vt = 1;
it = 1;
pf = .85;
omz = 60*2*pi;

% steal the next section from Problem Set 4
% take advantage of MATLAB's ability to do complex numbers
ia = pf - j * sqrt(1 - pf^2);      % this is the complex current
e1 = vt + ia*ra+j*xq*ia;          % this is on the q- axis
va = vt + ia*ra;                  % this is voltage 'inside' resistance
tha = angle(va);                  % pretty small angle
delta = angle(e1);                % this is the torque angle
psi = angle(ia);                  % and this is power factor angle
idg = it*sin(delta - psi);          % since the psi we calculate is negative
iqg = it*cos(delta - psi);
ea = abs(e1) + (xd - xq) * idg;     % voltage behind synchronous reactance
vq = abs(va)*cos(delta - tha);      % voltage on q axis is d axis flux
vd = abs(va)*sin(delta - tha);
psid = vq;
psiq = - vd;
psiad = psid + xal*idg;           % air-gap flux
psiaq = psiq + xal*iqg;           % these will be damper fluxes too
i_f = ea/xad;                     % field current
```

```
psif  =  psiad  +  xfl*i_f;        %  flux  at  field  terminal

%  need  to  use  motor  convention  currents
i_d  =  - idg;
i_q  =  - iqg;

%  now  build  the  'transition  matrix'
M=zeros(7);                %  first  we  make  space  for  it
                  %  detailed  direct  axis  parameters
                  %  assemble  direct  axis  inductance  matrix
xmd  =  [xd  xad  xad;  xad  xkd  xad;  xad  xad  xf];
ymd  =  inv(xmd);          %  invert  to  get  admittances
                  %  now  do  the  same  for  quad  axis
xmq  =  [xq  xaq;  xaq  xkq];      %  assemble  quadrature  axis  inductance  matrix
ymq  =  inv(xmq);                  %  and  here  is  admittance
ydd  =  ymd(1,1);          %  elements  of  direct-axis  admittance
ydk  =  ymd(1,2);
ykd  =  ymd(2,2);
ydf  =  ymd(1,3);
yff  =  ymd(3,3);
ykf  =  ymd(2,3);
yqq  =  ymq(1,1);          %  elements  of  quad-axis  admittance
yqk  =  ymq(1,2);
ykq  =  ymq(2,2);
                                   %  put  together  the  transition  matrix
M(1,1)  =  - omz*ra*ydd;
M(1,2)  =  - omz*ra*ydk;
M(1,3)  =  - omz*ra*ydf;
M(1,4)  =  omz;
M(1,6)  =  psiq;
M(1,7)  =  omz*vt*cos(delta);
M(2,1)  =  - omz*rkd*ydk;
M(2,2)  =  - omz*rkd*ykd;
M(2,3)  =  - omz*rkd*ykf;
M(3,1)  =  - omz*rf*ydf;
M(3,2)  =  - omz*rf*ykf;
M(3,3)  =  - omz*rf*yff;
M(4,1)  =  -omz;
M(4,4)  =  - omz*ra*yqq;
M(4,5)  =  - omz*ra*yqk;
M(4,6)  =  - psid;
M(4,7)  =  - omz*vt*sin(delta);
M(5,4)  =  - omz*rkq*yqk;
M(5,5)  =  - omz*rkq*ykq;
C  =  (.5*omz/H);                  %  convenient  shorthand  notation
```

```
M(6,1) = C * (i_q - ydd * psiq);
M(6,2) = - C * ydk * psiq;
M(6,3) = - C * ydf * psiq;
M(6,4) = C * (yqq * psid - i_d);
M(6,5) = C * yqk * psid;
M(7,6) = 1;
fprintf('Eigenvalues of a Synchronous Machine\n');
S = eig(M)          % this step gets the eigenvalues

fprintf('Eigenvalues of equivalent second order model\n')
% now try to do the simple second order equivalent
T = 1.97;                          % synchronizing coefficient
B = 2.44;                          % damping coefficient
M2 = [-B -C*T; 1 0];               % second order transition matrix
S2 = eig(M2)
```