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

Problem Set 11 Solutions December 4, 2005

Problem 1 This is a straightforward linear problem: the current is the sum of 'particular' and 'homogeneous' parts:

$$i(t) = i_p(t) + i_h(t)$$

The *particular* solution is the driven part:

$$i_p(t) = \Re \frac{V}{Z} e^{j\omega t} = \frac{V}{\sqrt{R^2 + X^2}} \cos(\omega t - \phi)$$

where $X = \omega L$ and $\phi = \tan^{-1} X/R$.

The homogeneous part is the natural response to the undriven system with amplitude such that the initial current must be zero, so

$$i_h(t) = \frac{V}{\sqrt{R^2 + X^2}} \cos(\omega t_0 - \phi)\, e^{-\frac{R}{L}t}$$

where the switch is closed at $t = t_0$

As it turns out, the problem statement has a poor choice of parameters, since if the switch is closed at time zero there is very little offset, as is shown in Figure 1.

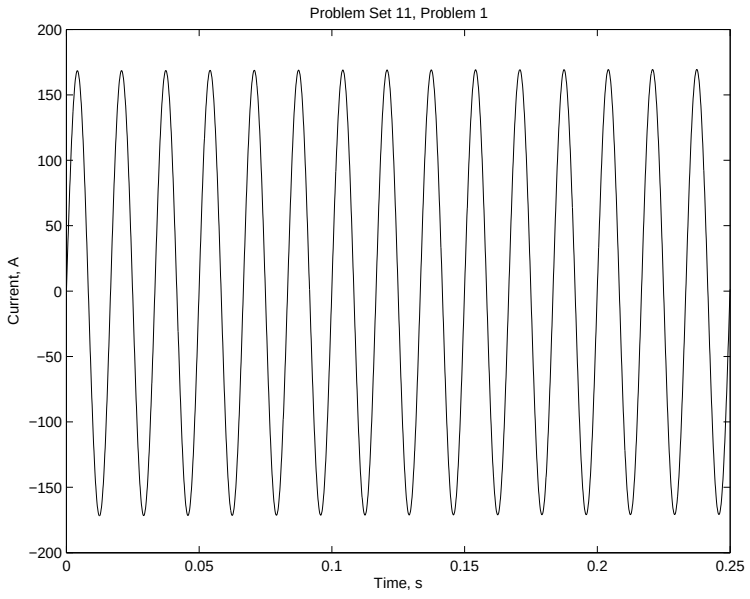

Figure 1: Switch closed at t=0

To have a zero offset, $\omega t_0 - \phi = \frac{\pi}{2}$ (or an integer times $\frac{\pi}{2}$. This is shown in Figure 2. If we use a different delay we can actually see the decaying offset, as in Figure 3.

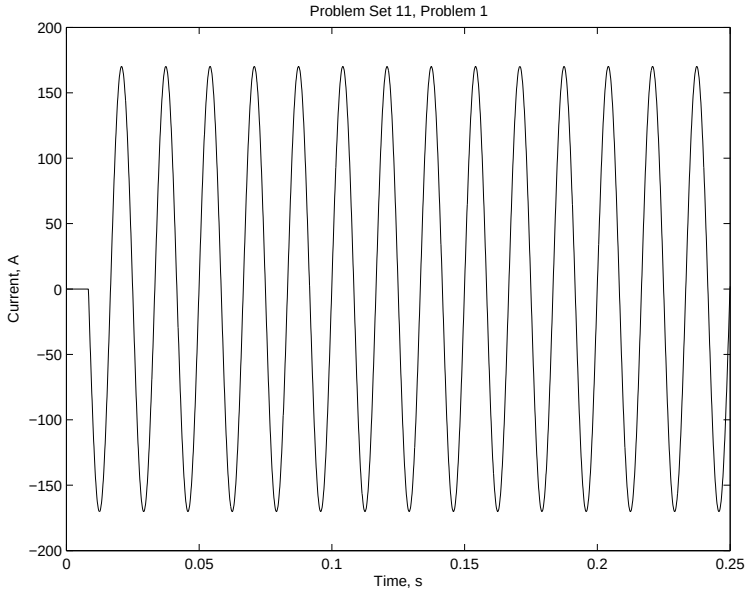

Figure 2: Switch Closed at t = .0083 to zero offset

Problem 2: Turbogenerator For reference we leave the original table of parameters here.

Synchronous, d- axis reactance	x_d	2.2	
Synchronous, q- axis reactance	x_q	2.0	
Transient, d- axis reactance	x'_d	0.45	
Subtransient, d- axis reactance	x_d"	0.22	
Subtransient, q- axis reactance	x_q"	0.22	
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}"	0.3	s
Inertial Constant	H	4.0	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:

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

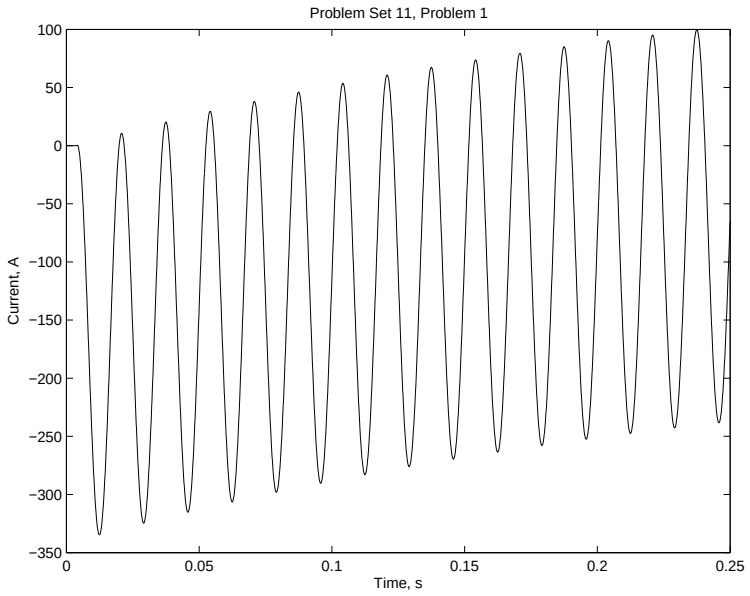

Figure 3: Switch Closed at t = .004

$$\begin{aligned} x_{aq} &= x_q - x_{a\ell} \\ x_{kq\ell} &= x_{aq}\frac{x_q" - x_{a\ell}}{x_q - x_q"} \end{aligned}$$

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 `p11.m` which is also appended. The results are, in both per-unit and ohms (referred to the stator: see below):

```
6.685 Problem Set 11: Basics
Parameter            Per Unit      Ohms
  xad                   2.1           1.29
```

xaq	1.9	1.17
xfl	0.42	0.258
xkdl	0.183	0.112
xkql	0.128	0.0787
rf	0.00134	0.000822
rkd	0.00706	0.00434
rkq	0.0179	0.011

2. The *base* impedance is:

$$Z_B = \frac{V_B^2}{P_B} = \frac{26^2}{1100} \approx .6145\Omega$$

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 4.

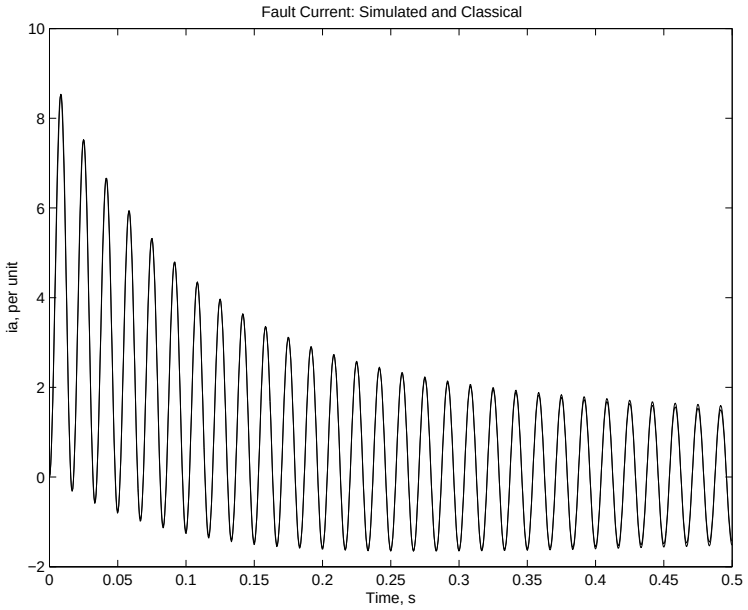

Figure 4: Phase A Fault Current: Simulated and Classical

The difference between the 'simulated' and 'classical' calculations are small enough that it is difficult to see in Figure 4. We can take the difference and plot it and do that in Figure 5.

5. The quantity 'voltage behind subrtransient reactance' is, for this fault, in 'classical' terms:

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

This quantity might also be described, from the simulation model, as:

$$e_q" = \psi_d - x_d" i_d$$

This is a side effect of the fault transient calculation and the 'classical' and simulated versions are shown in Figure 6.

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

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

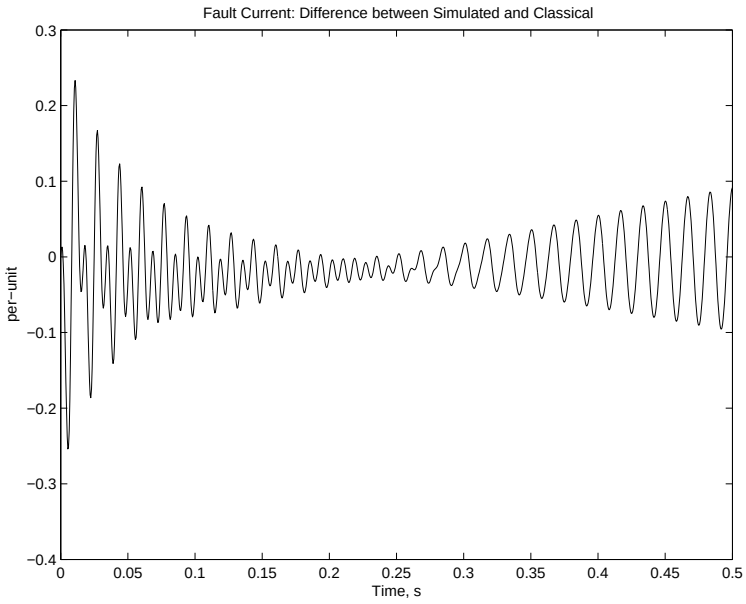

Figure 5: Phase A Fault Current: Difference between Simulated and Classical

and the transient and subtransient flux quantities are:

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

then:

Values have been computed in the appended script:

```
Problem 11.6
Torque Angle     =      1.107  Power Factor Angle =          0
D axis flux psid =     0.4472  Q axis flux psiq   =    -0.8944
D axis current Id=     0.8944  Q axis current Iq  =     0.4472
V behind xdp, eqp=     0.8497  V behind sync x eaf=      2.415
V beh xdpp, eqpp =      0.644  V behind xqpp edpp =    -0.9928
```

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

These results are shown in Figure 7

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

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

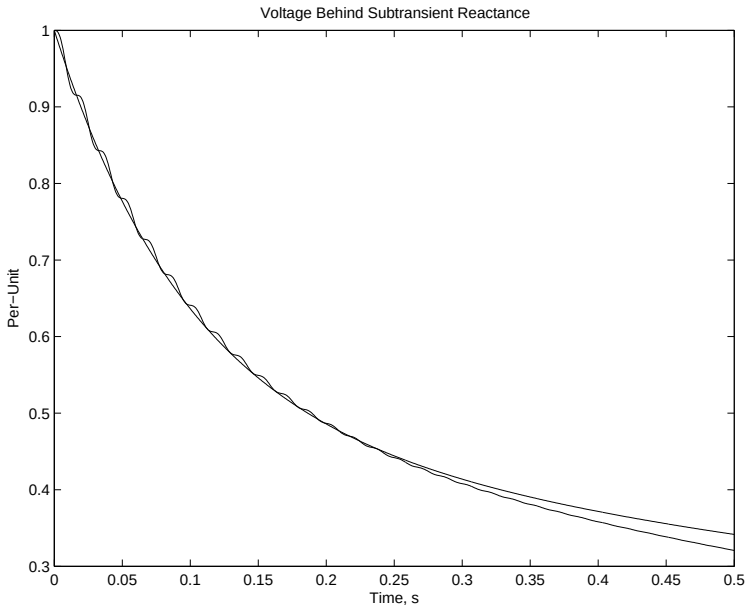

Figure 6: Voltage Behind Subtransient Reactance: Short

Values for the important parameters were calculated in conjunction with Problem 6. The results are plotted in Figure 8.

8. The equal area problem is formulated to be:

$$\int_{\delta_0}^{\delta_f} T_m d\delta + \int_{\delta_c}^{\delta_f} T_e d\delta = 0$$

where δ_0 is the original starting angle, δ_f is the maximum angle to be achieved during the transient (this is the angle on the unstable side of the torque-angle curve that has the same torque as the original starting angle) and δ_c is the 'critical clearing' angle: the angle by which the network connection must have been restored. Since we can formulate torque as:

$$T_e = -T_1 \sin\delta - T_2 \sin 2\delta$$

, this becomes:

$$T_m \left(\delta_f - \delta_0\right) + T_1 \left(\cos\delta_c - \cos\delta_f\right) + T_2 \left(\cos 2\delta_c - \cos 2\delta_f\right) = 0$$

I have avoided having to do any real trig here by using MATLAB's `fzero()` function to, first of all, find the initial and final torque angles and then finding the critical clearing angle. The critical clearing time is then just:

$$t_c = \sqrt{\frac{4H\left(\delta_c - \delta_0\right)}{\omega_0}}$$

This is automated in a script attached and the results for the two cases are:

```
Critical Clearing Time by Equal Area
xd =   2.2   xq =    2
```

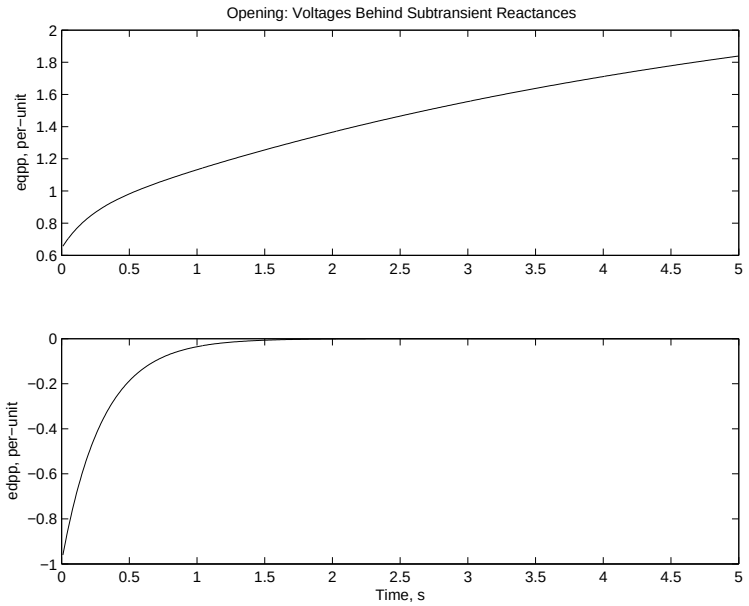

Figure 7: Voltage Behind Subtransient Reactance: Open

```
xdp =  0.45   H  =    4
          Salient Model    Round Model
Delt zero  =      1.107            0.5581
Delt final =      2.855             2.583
Crit Angle =      1.966             1.345
Crit Time  =      0.191            0.1827
```

Note the initial torque angle for the correct, or salient, case matches the torque angle computed earlier.

Finally, this problem has been simulated using straightforward but perhaps tedious expressions which are all in the attached scripts. One simulated case, very close to the critical clearing time, is shown in Figure 9. As it turns out, equal area turns out to be quite conservative. We find critical clearing time to be about 0.2245 seconds, as opposed to 0.191 seconds from the 'equal area' criterion.

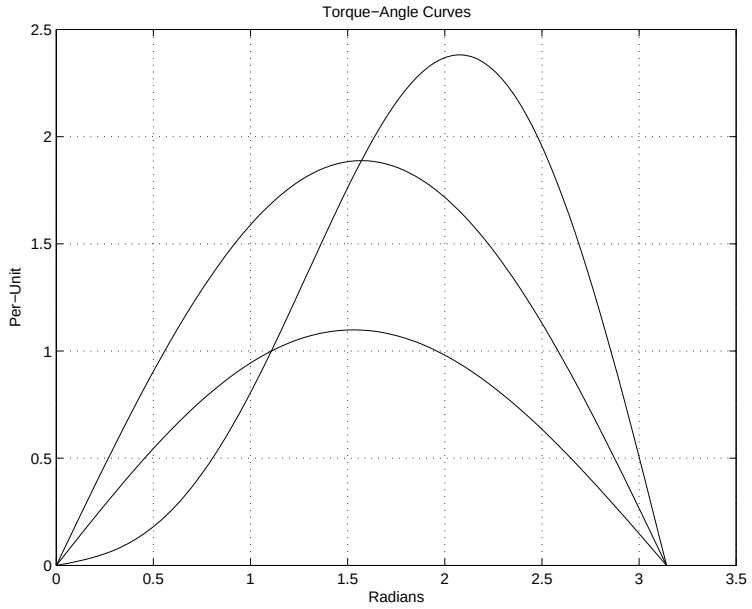

Figure 8: Torque-Angle Curves

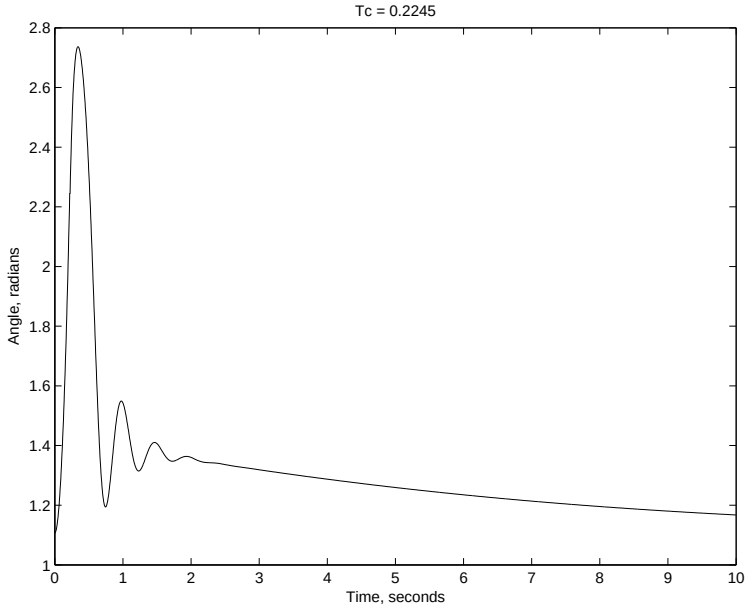

Figure 9: Near Critical (but stable) Swing

```
% 6.685 Problem Set 11, Problem 1
% we are using matlab to plot only
V = 170;
R = .01;
L = .00265;
om = 377;
X = om*L;
phi = atan(X/R);
Im = V/sqrt(R^2+X^2);

t = 0:.0001:.25;
t0 = input('starting time?')
i = zeros(size(t));
for k = 1:length(t)
    if t(k) < t0, i(k) = 0;
    else
      i(k) = Im * cos(om * t(k) - phi) - Im * cos(om*t0-phi) * exp(-(R/L) * (t(k)- t0) );
  end
end

figure(1)
plot(t, i)
title('Problem Set 11, Problem 1')
ylabel('Current, A')
xlabel('Time, s')
```

```
% 6.685 Problem Set 11, basic calculator
p11params           % 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;

fprintf('6.685 Problem Set 11: 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);

% Parameters for Problem Set 11:

xd=2.2;           % synchronous d- axis reactance
xq=2.0;           % synchronous q- axis reactance
xdp = .45;        % transient (d-axis) reactance
xdpp = .22;       % subtransient d- axis reactance
xqpp = .22;       % subtransient q- axis reactance
tdop = 5;         % transient (open circuit) time constant
tdopp = .2;       % subtransient d- axis time constant
tqopp = .3;       % subtransient q- axis time constant
xl = .1;          % armature leakage reactance
omz = 60*2*pi;    % base frequency
H = 4;            % rotor inertia constant
ta = .1;          % armature time constant
psi=0;            % power factor angle
Pb = 1.1e9;       % base power (rating)
Vb = 26e3;        % base voltage (rating)
```

```
% model elements from terminal parameters mi.m
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 miq.m
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);
```

```
% Problem Set 11, Problems 2, parts 3,4,5

global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz ra vf
p11params
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');

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

% estimate of "voltage behind subtransient reactance"
eqpp = psi(:,1) - xdpp .* id;
edpp = psi(:,2) + xqpp .* iq;
% and done by 'classical methods'
eqppc = xdpp/xd + (xdpp/xdp - xdpp/xd) .* exp(-t ./ tdp) ...
+ (1 - xdpp/xdp) .* exp(-t ./ tdpp);

figure(3)
clf
plot(t, eqpp, t, eqppc)
     title('Voltage Behind Subtransient Reactance')
     ylabel('Per-Unit')
     xlabel('Time, s')
```

```
% Massachusetts Institute of Technology
% Department of Electrical Engineering and Computer Science
% 6.685 Electric Machines
% Problem Set 11 Problem 2, part 6: Open Circuit voltage

p11params

% 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');
subplot 212
plot(t, edpp);
xlabel('Time, s');
ylabel('edpp, per-unit');

fprintf('Problem 11.6\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)
```

```
% Massachusetts Institute of Technology
% Department of Electrical Engineering and Computer Science
% 6.685 Electric Machines
% Problem Set 11, Problem 2 part 9
clear
global ydd ydk ykd ydf yff yqq yqk ykf ykq rf rkd rkq omz vf H Tm yado ydo yfo xkq
% parameters
p11params
t0 = 0;
tf = 10;
% find model parameters
% direct axis
[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);
% quadrature axis
[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);
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
delt = atan(xq);
psid = cos(delt);
psiq = - sin(delt);
id = sin(delt);
eaf = psid + xd * id;
ifield = eaf/xad;
vf = rf*ifield;
psikd0 = psid * xad/xd;
psikq0 = psiq * xaq/xq;
psiad0 = psid + xl * id;
psif0 = psiad0 + xfl * ifield;
```

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

tc = input('Clearing Time? ==> ');
dt = (tf - t0)/2048;
timeo = t0:dt:tc;

[tb,xb] = ode23('so', timeo, x0);

% tc is clearing time: needs to re-establish initial conditions
xc = xb(length(tb),:);

times = tc:dt:tf;
[ta, xa] = ode23('scl', times, xc);

t = [tb' ta'];
x = [xb' xa'];

figure(5)
clf

plot(t, x(5,:));

xlabel('Time, seconds');
ylabel('Angle, radians');
title(sprintf('Tc = %g',tc));
```

```
function dx = so(t, x);

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

psikd = x(1);
psikq = x(2);
psif = x(3);
om = x(4);
delt = x(5);

%psid = cos(delt);
%psiq = -sin(delt);

ikd = ydo * psikd + yado * psif;
iff = yado * psikd + yfo * psif;

ikq = psikq/xkq;

Te = 0;

xdot1 = -omz*rkd*ikd;
xdot2 = -omz*rkq*ikq;
xdot3 = omz*vf - omz*rf*iff;
xdot4 = (omz/(2*H)) * (1 + Te);
xdot5 = om - omz;

dx = [xdot1 xdot2 xdot3 xdot4 xdot5]';

function dx = scl(t, x);

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

psikd = x(1);
psikq = x(2);
psif = x(3);
om = x(4);
delt = x(5);

psid = cos(delt);
psiq = -sin(delt);

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;

xdot1 = -omz*rkd*ikd;
xdot2 = -omz*rkq*ikq;
xdot3 = omz*vf - omz*rf*iff;
xdot4 = (omz/(2*H)) * (1 + Te);
xdot5 = om - omz;

dx = [xdot1 xdot2 xdot3 xdot4 xdot5]';
```