Problem Set 1 Solutions   February 18, 2006

Problem 2.7 This one can be done almost by inspection. The capacitor delivers reactive power $Q_c = -5$, the inductor absorbs reactive power: $Q_L = 10$. The resistor absorbs real power: $P_R = .1$. So the total complex power delivered by the source is

$$S_s = P_R + jQ_c + jQ_L = .1 + j5$$

Problem 2.11 The delta-connected capacitors may be replaced by a wye connected capacitor with value $X_c = 1/3$. The delta connected voltage source can be replaced with a wye connected source with value $V = \frac{1}{\sqrt{3}}e^{-j\frac{\pi}{6}}$. then we can work a single phase problem. Voltage at the node a' is found through superposition. Note that the voltage divider relationship in both directions is between a reactance of value .1 and a parallel combination of the other reactance of value .1 and the capacitance with reactance of $-\frac{1}{3}$. The parallel impedance is inductive and has value:

$$j.1 || (-j\frac{1}{3}) = \frac{j}{7}$$

$$\frac{j.1 || \frac{-j}{3}}{\frac{j.1}{} | \frac{-j}{3} + .1} = \frac{10}{17}$$

The voltage at the node indicated is then:

$$v_a = \frac{10}{17}\left(1 + \frac{1}{\sqrt{3}}e^{-j\frac{\pi}{6}}\right)$$

Since the problem is balanced, the other two phase voltages are the same but rotated by $120°$

$$v_b = v_a e^{-j\frac{2\pi}{3}}$$
$$v_c = v_a e^{j\frac{2\pi}{3}}$$

Line-line voltage is simply:

$$v_a - v_b = v_a\sqrt{3}e^{j\frac{\pi}{6}}$$

Problems 4.1, 4.2 and 4.15 This can be worked right out of the textbook, using the formulae on Page 96 and 98. This is all worked out in the following MATLAB script:

```
%  Problem  4.1  and  4.2
%  basic  info  for  problem  4.1
z  =  .17  +  j  *  .79;
y  =  j  *  5.4e-6;
```

```
gamma1  =  sqrt(y*z);
Zc1  =  sqrt(z/y);
alpha1  =  real(gamma1);
beta1  =  imag(gamma1);
%  and  now  for  problem  4.2
z=.02+j  *  .54;
y  =  j*7.8e-6;
gamma  =  sqrt(y*z);
Zc2  =  sqrt(z/y);
alpha2  =  real(gamma);
beta2  =  imag(gamma);
%  Idiot  Check  Time:
C  =  186000;            %  remember  we  are  dealine  in  miles
beta_f  =  377/C;
fprintf('Problems  4.1  and  4.2\n')
fprintf('  138  kV  LineDecay  Rate  =  %g  Propagation  Constant  =  %g Char  Impedance  =  %g\n',
fprintf('  765  kV  LineDecay  Rate  =  %g  Propagation  Constant  =  %g Char  Impedance  =  %g\n',
fprintf('  Free  Space                            Propagation  Constant  =  %g\n', beta_f)
%  now  on  to  problem  4.15
l  =  400;               %  miles!
Pl  =  100e6;            %  receiving  end  real  power
Ql  =  Pl*sqrt(1-.95^2);
T11  =  cosh(gamma*l);   %  these  are  the  parameters  of  the  T-model
T22  =  T11;
T12  =  Zc2*sinh(gamma*l);
T21  =  sinh(gamma*l)/Zc2;
Vr  =  750000/sqrt(3);   %  line  to  neutral  voltage
Ir  =  Pl/(3*Vr)  - j*Ql/(3*Vr);        %  and  line  current
Vs  =  Vr  *  T11  +  Ir  *  T12;
Is  =  Vr  *  T21  +  Ir  *  T22;
delta  =  angle(Vs);
Ps  =  3*real(Vs  *  conj(Is));
Qs  =  3*imag(Vs  *  conj(Is));
eff  =  Pl/Ps;
fprintf('Receiving  End  Voltage  =  %g   Current  =  %g\n',  Vr,  abs(Ir))
fprintf('Sending  End  Voltage  =  %g    Current  =  %g\n',  abs(Vs),  abs(Is))
fprintf('Sending  End  Real  Power  =  %g   Reactive  Power  =  %g\n',  Ps,  Qs)
fprintf('Power  Angle  =  %g   Efficiency  =  %g\n',  delta,  eff)
%  another  idiot  check:  what  is  Surge  Impedance  Loading?
VI_si  =  3*Vr^2/Zc2;
fprintf('Surge  Impedance  Loading  P  =  %g  Q  =  %g\n',  real(VI_si),  imag(VI_si))
```

And the screen printout that results is:

```
 Problems  4.1  and  4.2
```

```
138 kV Line
Decay Rate = 0.000220969 Propagation Constant = 0.00207722 Char Impedance = 384.67
765 kV Line
Decay Rate = 3.79993e-05 Propagation Constant = 0.00205267 Char Impedance = 263.16
                 Free Space Propagation Constant = 0.00202688
Receiving End Voltage = 433013 Current = 80.6455
Sending End Voltage = 300852   Current = 1189.7
Sending End Real Power = 1.12271e+08 Reactive Power = -1.06789e+09
Power Angle = 0.0648288 Efficiency = 0.890702
Surge Impedance Loading P = 2.13673e+09 Q = 3.95555e+07
```

Note we have done a couple of things beyond the problem statement: the propagation coefficient of the transmission line is nearly the same as free space, so the two lines have propagation coefficients that are similar, but their decay coefficients are, as one would expect, quite a bit different. Note also this line is very lightly loaded, which is why the sending end current and voltage are markedly different from the receiving end conditions. Were we to drive the line with power closer to the surge impedance loading the two end conditions would be much closer together.

Problem 4.27 Well, we kind of wimped out here and let MATLAB do all the work. Here is the script that does it all:

```
% Problem 4.27
% data
Rl = .01;                       % line impedance
Xl = .1;
Pl = .5;                         % real power on load side
Ql = .5;
% despairing of closed form solution, set up Newton's method
delt = .1;                      % starting guess
not_done = 1;
while not_done == 1,
    P = real((exp(-j*delt)-1)/(Rl - j*Xl));      % real power at load bus for this delta
    J = real(-j*exp(-j*delt)/(Rl - j*Xl));       % one element jacobian
    if abs(P-Pl) < .0000001,                       % test
        not_done = 0;
    end
    delt = delt - (P-Pl)/J;                        % correction
end
Qb = imag((exp(-j*delt)-1)/(Rl - j*Xl));        % this is total reactive power at load
Qg = Qb + Ql;                                   % add reactive load
% now get the sending end conditions
Ss = (1-exp(j*delt))/(Rl - j*Xl);               % sending end line power
Sg = Ss + .5 + j*.5;                            % required at generator
% Now we do the idiot check
```

```
Il  =  abs(Ss);                                % this is line current
Pd  =  Rl * Il ^2;                             % line power dissipation
Pe  =  real(Sg) - .5 - Pl - Pd;                % should be zero
Qd  =  Xl * Il^2;                              % reactive power in line
Qe  =  imag(Sg) - .5 - Qd - Qb;                % should be zero

fprintf('Problem 4.27\n')
fprintf('Injected Reactive Power at Load Bus = %g\n', Qg)
fprintf('Load Angle = %g\n', delt)
fprintf('Generator Power = %g + j*%g\n', real(Sg), imag(Sg))
fprintf('Line Dissipation = %g, Reactive Power = %g\n', Pd, Qd)
fprintf('Check for errors = %g, %g\n', Pe, Qe)
```

Formal output from that script is: Note that we have done an 'idiot check' to see if both real and reactive powers balance (and they do, to the extent that $10^{-17} \approx 0$).

```
>>Problem 4.27
Injected Reactive Power at Load Bus = 0.437176
Load Angle = 0.0506499
Generator Power = 1.00254 + j*0.46257
Line Dissipation = 0.00253947, Reactive Power = 0.0253947
Check for errors = -3.42608e-17, -1.249e-16
```