Scilab Textbook Companion for Transmission Lines And Networks by U. Sinha[1]

Created by
Aakanksha Gupta
Bachelor of Engineering
Electrical Engineering
Thapar University
College Teacher
Dr.sunil Kumar Singla
Cross-Checked by
Lavitha Pereira

November 24, 2013

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website http://scilab.in

Book Description

Title: Transmission Lines And Networks

Author: U. Sinha

Publisher: Satya Prakashan, New Delhi

Edition: 5

Year: 1987

ISBN: 81-7684-017-3

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

List of Scilab Codes

Chapter 1

Transmission Lines

Scilab code Exa 1.1 Calculating the resistance and inductance and leakage and capacitance per km and velocity of propagation

```
clear;
clc;
Zo=710* exp(%i*(-%pi/(180/16)));f=1000;
w=2*%pi*f;
a=.01;b=.035;
P=a+%i*b;
Z=Zo*P;
R=real(Z);
r=round(R*100)/100;
printf('-Resistance R = %f ohms/km\n',r);
L=((imag(Z))/w)*10^3;
l=round(L*100)/100;
printf('-Inductance L = %f mH/km\n',l);
Y=P/Zo;
G=real(Y);
printf('-Conductance G = %f mhos/km\n',-G);
C=((imag(Y))/w)*10^6;
c=round(C*1000)/1000;
printf('-Capacitance C = %f microfarads/km\n',c);
Vp=round(w*1000/(b*10^5))/1000;
```

```
printf('-Velocity of propagation Vp = %f^* 10^5 km/
    sec',Vp);
```

Scilab code Exa 1.2 Calculating the values of the line constants

```
clear;
clc;
Zo=2039.6;f=800; //value of Zo as taken in solution
P=0.054* exp(%i*(%pi/(180/87.9)));
w=2*%pi*f;
Z=Zo*P;
R=real(Z);
printf('-Resistance R = %f ohms/km\n',R);
L=(imag(Z))/w;
printf('-Inductance L = %f mH/km\n',L*(10^3));
Y=P/Zo;
G=real(Y);
printf('-Conductance G = %f micromhos/km\n',G*(10^6)
    );
C=((imag(Y))/w)*(10^6);c=round(C*10000)/10000
printf('-Capacitance C = %f microfarads/km\n',c);
```

Scilab code Exa 1.3 Determining Zo and a and b

```
clear;
clc;
R=10;L=.0037;f=1000;G=.4*(10^-6);C=.0083*(10^-6);
w=2*%pi*f;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
Zo=sqrt(Z/Y);
C=round(real(Zo));
D=round(imag(Zo));
```

```
printf('-Zo = %f + j(%f) ohms\n',C,D);
P=sqrt(Z*Y);
a=real(P);
a1=round(a*10000)/10000;
printf('-Attenuation constant a = %f neper/km\n',a1)
    ;
b=imag(P);
b1=round(b*10000)/10000;
printf('-Phase constant b = %f radians/km',b1);
```

Scilab code Exa 1.4 Calculating terminating impedance and attenuation and phase constant

```
clear;
clc;
f=1000;R=6;L=.0022;G=.25*(10^-6);C=.005*(10^-6);l
    =100;
//value of C as taken in solution
w=2*%pi*f;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
Zo=sqrt(Z/Y);
C=real(Zo);
D=imag(Zo);
printf('(i)Terminating impedance for which there
    will be no reflection is Zo = %f /_%f ohms\n',fix
    (abs(Zo)),round(((atan(imag(Zo),real(Zo))*180/%pi
    ))*10)/10);
P=sqrt(Z*Y);
a=real(P);
b=imag(P);
a1=a*l*8.66;
printf('(ii)(a)Attenuation suffered while travelling
     = %f db\n',a1);
Vp=(w/b)*(10^-5);v=round(Vp*100)/100;
```

```
printf('(ii)(b)Phase velocity Vp = %f * 10^5 km/sec\n',v);
```

Scilab code Exa 1.5 Calculating phase velocity and characteristic impedance and propagation constant and power delivered at receiving end

```
clear;
clc;
f=1000;R=10.4;L=.0037;G=.8*(10^-6);C=.00835*(10^-6);
   l=1000;Vs=1;
w=2*%pi*f;
Z=R+round((%i*w*L));
Y=G+(%i*w*C);
Zo=sqrt(Z/Y);
printf('-Characteristic impedance is Zo = %f /_%f
   ohms\n',fix(abs(Zo)),round(((atan(imag(Zo),real(
   Zo))*180/%pi))));
P=sqrt(Z*Y);
printf('-Propagation constant P = %f + j(%f)\n',fix(
   real(P)*10^4)/10^4,round(imag(P)*10^4)/10^4);
b=imag(P);

Is=Vs/Zo;Vp=(w/b)*(10^-5);v=round(Vp*100)/100;
printf('-Phase velocity Vp = %f * 10^5 km/sec\n',v);
Ir=Is*exp(-P*l);
P=((abs(Ir))^2)*real(Zo);
printf("-Power delivered at receiving end = %f micro
   -watt",P*(10^6));
```

Scilab code Exa 1.6 Calculating characteristic impedance and propagation constant

```
clear;
```

```
clc;
f=5000/(2*%pi);R=196;C=.09*(10^-6);L=.71*(10^-3);G
    =0;
//value of C as taken in solution
w=2*%pi*f;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
Zo=sqrt(Z/Y);
printf('-Characteristic impedance is Zo = %f /_%f
    ohms\n',fix(abs(Zo)),round(((atan(imag(Zo),real(
    Zo))*180/%pi))*10)/10);
P=sqrt(Z*Y);F=fix(abs(P)*100)/100;
printf("-Propagation constant P =  %f /_%f ohms\n',F
    ,((atan(imag(P),real(P))*180/%pi)));
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Scilab code Exa 1.7 Calculating Zo and a and b and Vp

```
clc;
R=10.4;L=3.666*(10^-3);G=.08*(10^-6);C
    =.00835*(10^-6);w=5000;
//value of L and C as taken in solution
Z=R+(%i*w*L);
Y=G+%i*w*C;
Zo=sqrt(Z/Y);
printf('-Zo = %f /_%f ohms\n',fix(abs(Zo)),round(((
    atan(imag(Zo),real(Zo))*180/%pi))*10)/10);
P=sqrt(Z*Y);
a=real(P);
b=imag(P);
printf('-Attenuation constant a = %f neper/km\n',a);
printf("-Phase constant b = %f radians/km\n",b);
```

```
Vp=(w/b)*(10^-5);v=round(Vp*1000)/1000;
printf('(ii)(b)Phase velocity Vp = %f * 10^5 km/sec\
    n',v);
//the difference in result of Zo is due to erroneous
     value in textbook.
disp("The difference in result of Zo is due to
    erroneous value in textbook")
```

Scilab code Exa 1.8 Calculating characteristic impedance

```
clear;
clc;
R=65;L=1.6*(10^-3);C=.1*(10^-6);G=2.25*(10^-6);f
    =800;
w=5000;
Z=R+round(%i*w*L);
Y=G+%i*w*C;
Zo=sqrt(Z/Y);
printf('-Characteristic impedance is Zo = %f /_%f
    ohms\n',(abs(Zo)),(((atan(imag(Zo),real(Zo))*180/
    %pi))));
```

Scilab code Exa 1.9 Calculating phase velocity and attenuation

```
clear;
clc;
R=6;L=2*(10^-3);G=.5*(10^-6);C=.005*(10^-6);f=1000;l
    =100;
w=2*%pi*f;
Z=R+(%i*w*L);
Y=G+%i*w*C;
Zo=sqrt(Z/Y);
P=sqrt(Z*Y);
```

```
a=real(P);
b=imag(P);
a1=fix(a*8.66*l*10^3)/10^3;
printf("-Attenuation suffered while travelling = %f
   db\n",a1);
Vp=fix(w/b)/10^3;
printf("-Phase velocity Vp = %f km/sec",Vp );
```

Scilab code Exa 1.10 Calculating characteristic impedance

```
clc;
R=20;L=10*(10^-3);ins=0.1*(10^6);C=.1*(10^-6);w
   =5000; //ins=insulation resistance
G=1/ins;
Z=R+(%i*w*L);
Y=G+%i*w*C;
Zo=sqrt(Z/Y);
C=real(Zo);
D=imag(Zo);
printf('Input impedance Zo = %f /_%f ohms\n',(abs(Zo
   )),round(((atan(imag(Zo),real(Zo))*180/%pi))*10)
   /10);
//the difference in result of Zo is due to erroneous
    value in textbook.
disp("The difference in result of Zo is due to
   erroneous value in textbook")
```

Scilab code Exa 1.11 Calculating the voltage across load in terms of the sending end voltage

```
clear;
clc;
x=1;l=12;V=0.1;
```

```
V1=1-V;
//V1=ratio of voltage at 1km from the sending end to
    the voltage at sending end
P=-log(V1);
V2=exp(-P*l);
x=V2*100;x1=round(x*100)/100;
//x=ratio of voltage across the load impedance to
   the voltage at sending end
printf("Voltage across the load impedance is %f
   percentage of the sending end voltage",x1);
```

Scilab code Exa 1.12 Calculating dc resistance and inductance and capacitance and ac resistance

```
clear;
clc;
a=.1;d=30;Ur=1;s=5.57*(10^7);e=1;f=30000;
//ur= relative magnetic permeability of conductor
   material ,
//s=conductivity of material
//e=relative dielectric constant of the material
L=(Ur+9.21*log10(d/a))*10^-7;
printf("-Inductance L = %f mH/km\n",L*10^3);
C=12.07*e/(log10(d/a));
printf("-Capacitance C = %f micromicrofarads/km\n",
   round(C*100)/100);
Rdc=2/(%pi*a*a*10^-2*10^-2*s);
Rdc1=Rdc*10^3;
printf("-D.C. resistance of line Rdc = %f ohms/km\n",
   round(Rdc1*100)/100);
Uo=4*%pi*10^-7;
//Uo=absolute permeability
Rac=(1/(a*10^-2))*(sqrt(f*Uo/(%pi*s)));
Rac1=Rac*10^3;
printf("-A.C. resistance of line = %f ohms/km",round(
   Rac1*100)/100);
//the difference in result of innductance and
   capacitance is due to erroneous value in textbook
   .
disp("The difference in result of inductance and
   capacitance is due to erroneous value in textbook
   ")
```

Scilab code Exa 1.13 Calculating per km high and low frequency inductance and capacitance and dc and ac resistance

```
clear;
clc;
a=0.2;b=0.8;c=1;
a1=a*10^-2;b1=b*10^-2;c1=c*10^-2;s=5.57*(10^7);e
    =2.5;f=100*(10^3);
//e=relative dielectric constant of the material
//s=conductivity of copper
eo=(1/(36*%pi))*10^-9;
Uo=4*%pi*10^-7;
//Uo=absolute permeability
Llf=(((Uo/(2*%pi))*log(b1/a1))+((Uo/(8*%pi))*((((4*
    c1^4)/((((c1^2)-(b1^2))^2))*log(c1/b1))-((2*c1^2)
    /((c1^2)-(b1^2))))))*10^3;
printf("-Low frequency inductance = %f mH/km\n",
    round(Llf*(10^3)*10000)/10000);
Lhf=((Uo/(2*%pi))*log(b1/a1))*10^3;
printf("-High frequency inductance = %f mH/km\n",
    round(Lhf*(10^3)*1000)/1000);
C=(2*%pi*eo*e/(log(b1/a1)))*10^3;
printf("-Capacitance = %f microfarads/km\n",round(C
    *(10^6)*10000)/10000);
Rdc=((1/(%pi*s))*((1/(a1^2))+(1/((c1^2)-(b1^2)))))
    *10^3;
printf("-D.c.resistance = %f ohms/km\n",round(Rdc
    *1000)/1000);
Rac=((sqrt(f*Uo/(4*%pi*s)))*((1/a1)+(1/b1)))*10^3;
printf("—A.c.resistance = %f ohms/km",round(Rac*100)
    /100);
```

Scilab code Exa 1.14 Calculating line attenuation and velocity of propagation

```
clear;
clc;
Vd=10;l=2;b=20;f=796;
w=2*%pi*f;
//Vd=voltage drop(in percentage),b=phase change(in
    degrees)
V=(100-Vd)/100; //V=Vr/Vs;
a=(20*log10(1/V))/l;
printf("(i)Attenuation = %f db/km\n",round(a*1000)
    /1000);
b1=(b/l)*(%pi/180); //b1=phase constant/km(in
    radians)
Vp=w/b1;
printf("(ii)The velocity of propagation = %d km/sec"
    ,Vp);
```

Scilab code Exa 1.15 Calculating sending and receiving end voltages and currents and power

```
clear;
clc;
l=200;Vg=10;Zg=500;Zs=683-(%i*138);P=0.0074+(%i
    *0.0356);Zo=Zs;
Is=Vg/(Zg+Zs);
modIs=abs(Is);
```

```
printf("-Sending end current = %f mA\n",round(modIs
    *(10^3)*100)/100);
Vs=Is*Zs;
modVs=abs(Vs);
printf("-Sending end voltge = %f V r.m.s.\n",round(
    modVs*100)/100);
Rs=real(Zs);
Ps=((modIs)^2)*Rs;
printf("-Sending end power = %f mW\n",round(Ps
    *(10^3)*100)/100);
Vr=modVs*exp(-P*l);
A=imag(-P*l);
printf("-Receiving end voltage = %f /_ %f V\n",round
    (abs(Vr)*100)/100,A);
Zr=Zs;
Ir=Vr/Zr;
modIr=abs(Ir);
printf("-Receiving end current = %f mA\n",round(
    modIr*(10^3)*100)/100);
Rr=Rs;
Pr=((modIr)^2)*Rr;
printf("-Receiving end power = %f mW",fix(Pr*(10^3)
    *100)/100);
```

Chapter 2

Open and Short Circuited Lines

Scilab code Exa 2.1 Calculating Zo

```
clc;
Zoc=900*exp(%i*(-%pi/(180/30)));Zsc=400*exp(%i*(-%pi
   /(180/10)));f=1.6*(10^3);
Zo=sqrt(Zoc*Zsc);
A=real(Zo);
B=imag(Zo);
printf("Zo = %f /_ %f ohms",abs(Zo),atan(B,A)*180/
   %pi);
```

Scilab code Exa 2.2 Calculating the line constants

```
clear;
clc;
f=796;Zoc=328*exp(%i*(-%pi/(180/29.2)));Zsc=1548*exp
   (%i*(%pi/(180/6.8)));l=50;
Zo=sqrt(Zoc*Zsc);
C=real(Zo);
D=imag(Zo);
```

```
printf("-Zo = %f /_ %f ohms\n",fix(abs(Zo)),atan(D,C
   )*180/%pi);
w=2*%pi*f;
Z1=sqrt(Zsc/Zoc);
A=real(Z1);
B=imag(Z1);
D=(1+A+(%i*B))/(1-(A+(%i*B)));
r=abs(D);
theta=atan((imag(D))/(real(D)))-%pi;
n=1;
P=(1/(2*l))*((log(r))+(%i*(theta+(2*n*%pi))));
E=real(P);
F=imag(P);
printf("-P = %f /_ %f\n",round(abs(P)*10000)/10000,
   round(atan(F,E)*180*100/%pi)/100);
Z=P*Zo;
R=real(Z);
L=(imag(Z))/w;
printf("-R = %f ohms/km\n",round(R*100)/100);
printf("-L = %f mH/km\n",round(L*(10^3)*100)/100);
Y=P/Zo;
G=real(Y);
C=(imag(Y))/w;
printf("-G = %f micro-mhos/km\n",round(G*(10^6)));
printf("-C = %f microfarads/km",round(C*(10^6)
   *10000)/10000);
```

Scilab code Exa 2.3 Calculating the received current and voltage

```
clear;
clc;
Zo=600;a=0.1;b=0.05;x=10;Is=20*(10^-3);
Vr=0;
printf("-Receiving end voltage Vr=0 because the
   receiving end has been short ciruited\n");
```

```
P=a+(%i*b);
Ir=Is/(cosh(10*P));
A=real(Ir);
B=imag(Ir);
printf("−Received current is Ir = %f /_ %f mA ",
   round(abs(Ir)*(10^3)*100)/100,fix(atan(B,A)
   *180*10/%pi)/10);
```

Scilab code Exa 2.4 Determining Zo and a and b and primary constants of line

```
clear;
clc;
Zoc=1930*exp(%i*(%pi/(180/68.9)));Zsc=1308*exp(%i*(-
   %pi/(180/76.2)));l=8;w=5000;
Zo=sqrt(Zoc*Zsc);
C=real(Zo);
D=imag(Zo);
printf("−Zo = %f /_ %f ohms\n",round(abs(Zo)),round(
   atan(D,C)*180*100/%pi)/100);
Z1=sqrt(Zsc/Zoc);
A=round(real(Z1)*1000)/1000;
B=round(imag(Z1)*1000)/1000;
D=(1+A+(%i*B))/(1-(A+(%i*B)));K=round((1+A+(%i*B))
   *100)/100;J=round((1-(A+(%i*B)))*10)/10;
phi1=atan(imag(K),real(K))+(%pi/4);
phi2=atan(imag(J),real(J));
phi3=phi1-phi2;
P=(1/(2*8))*(log(round(abs(K/J)*10)/10)+(%i*(round(
   phi3*10)/10+(6.28))));
printf(" −a = %f neper/km\n",fix(real(P)*100)/100);
printf(" −b = %f radians/km\n",fix(imag(P)*1000)
   /1000);
Z=P*Zo;
R=real(Z);
```

```
L=(imag(Z))/w;
printf(" -R = %f ohms/km\n",round(R*1000)/1000);
printf(" -L = %f mH/km\n",round(L*(10^6))/1000);
Y=P/Zo;
G=real(Y);
C=(imag(Y))/w;
printf(" -G = %f micro-mhos/km\n",-round(G*(10^9))
    /1000);
printf(" -C = %f microfarads/km",round(C*(10^9))
    /1000);
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Scilab code Exa 2.5 Calculating the line constants

```
clear;
clc;
w=5000;Zoc=1300*exp(%i*(%pi/(180/80)));Zsc=3200*exp(
    %i*(-%pi/(180/80)));l=40;
Zo=sqrt(Zoc*Zsc);
Z1=sqrt(Zsc/Zoc);
A=real(Z1);
B=imag(Z1);
D=(1+A+(%i*B))/(1-(A+(%i*B)));K=(1+A+(%i*B));J=(1-(A
    +(%i*B)));
r=round(abs(D)*1000)/1000;
theta=round(atan((imag(D))/(real(D)))*10)/10;
P=(1/(2*l))*((log(r))+(%i*(theta+6.28)));
b=imag(P);
Z=P*Zo;
R=real(Z);
L=(imag(Z))/w;
printf("R = %f ohms\n",fix(R*100)/100);
printf(" L = %f mH/km\n",round(L*(10^6))/1000);
Y=P/Zo;
G=real(Y);
C=(imag(Y))/w;
printf(" G = %f micro-mhos/km\n",round(G*(10^8))
   /100);
printf(" C = %f micro-farads/km",round(C*(10^9))
   /1000);
//the difference in result of L and C is due to
   erroneous value in textbook.
disp("The difference in result of L and C is due to
   erroneous value in textbook")
```

Scilab code Exa 2.6 Calculating Zo and P and line constants

```
clear;
clc;
Zoc=286*exp(%i*(-%pi/(180/40)));Zsc=1520*exp(%i*(%pi
   /(180/16)));l=50;f=700;
w=2*%pi*f;
Zo=sqrt(Zoc*Zsc);
S=real(Zo);
T=imag(Zo);
printf("-Characteristic impedance of transmission
   line is Zo = %f /_ %f ohms\n",round(abs(Zo)*100)
   /100,atan(T,S)*180/%pi);
A=atanh(sqrt(Zsc/Zoc));
P=A/l;
U=real(P);
V=imag(P);
printf("-Propagation constant = %f /_ %f\n",fix(abs(
   P)*1000)/1000,round(atan(V,U)*180*100/%pi)/100)
Z=P*Zo;
R=real(Z);
L=(imag(Z))/w;
```

```
printf("-R = %f ohms/km\n",round(R*100)/100);
printf("-L = %f mH/km\n",fix(L*(10^3)*100)/100);
Y=P/Zo;
G=real(Y);
C=(imag(Y))/w;
printf("-G = %f micromhos/km\n",fix(G*(10^6)*100)
    /100);
printf("-C = %f microfarads/km",fix(C*(10^6)*10^5)
    /10^5);
```

Scilab code Exa 2.7 Calculating the series impedance and shunt admittance per km of line

```
clear;
clc;
Zoc=2500*exp(%i*(-%pi/(180/70)));Zsc=49*exp(%i*(%pi
    /(180/25)));l=1;
f=(8000)/(2*%pi);
w=2*%pi*f;
Zo=sqrt(Zoc*Zsc);
A=atanh(sqrt(Zsc/Zoc));
P=A/l;
P1=P*l/1000;
Z=P1*Zo;
printf("Series impedance of line is Z = %f /_ %f
    ohms/km\n",round(abs(Z)*1000)/1000,atan(imag(Z),
    real(Z))*180/%pi);
Y=P1/Zo;
printf("Shunt admittance of line is Y = %f /_ %f
    mhos/km",round(abs(Y)*10^6*10)/10,atan(imag(Y),
    real(Y))*180/%pi);
```

Scilab code Exa 2.8 Calculating Zo and a and b

```
clear;
clc;
Zoc=2000*exp(%i*(-%pi/(180/80)));Zsc=20*exp(%i*(%pi
    /(180/20)));l=0.5;w=10000;
//value of length of cable as taken in solution
Zo=sqrt(Zoc*Zsc);
C=real(Zo);
D=imag(Zo);
printf("-Zo = %f /_ %f ohms\n",abs(Zo),atan(D,C)
    *180/%pi);
A=atanh(sqrt(Zsc/Zoc));
P=A/l;
a=real(P);
printf("-a = %f neper/km\n",fix(a*10000)/10000);
b=imag(P);
printf("-b = %f henry/km",round(b*10000)/10000);
```

Scilab code Exa 2.9 Calculating output current when far end is short circuited

```
clear;
clc;
Vs=1;f=1000;l=150;Zo=692*exp(%i*(-%pi/(180/12)));a
    =0.0047;b=0.022;Vr=0;
P=a+(%i*b);
Ir=1/(Zo*sinh((a*l)+(b*l)));
C=real(Ir);
D=imag(Ir);
printf("Output current(in amperes)= %f /_ %f mA",
    round(abs(Ir)*10^6)/1000,atan(D,C)*180/%pi);
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Scilab code Exa 2.10 Calculating Zo and a and b

```
clear;
clc;
f=20*(10^6);l=32;Zsc=17+(%i*19.4);Zoc=115-(%i*138);
Zo=sqrt(Zoc*Zsc);
X=real(Zo);
Y=imag(Zo);
printf("-Characteristic impedance = %f /_ %f ohms\n"
   ,round(abs(Zo)),round(atan(Y,X)*180*10/%pi)/10);
Z1=sqrt(Zsc/Zoc);
A=real(Z1);
B=imag(Z1);
D=(1+A+(%i*B))/(1-(A+(%i*B)));
r=sqrt(((real(D))^2)+((imag(D))^2));
theta=atan((imag(D))/(real(D)));
n=6;
P=(1/(2*l))*((log(r))+(%i*(theta+(2*n*%pi))));
b=imag(P);
a=real(P);
printf("-Attenuation factor = %f neper/m\n",round(a
   *10000)/10000);
printf("-Phase propagation factor = %f radians/m",
   round(b*100)/100);
```

Scilab code Exa 2.11 Calculating Zo and a and b

```
clear;
clc;
f=20*(10^6);Zoc=1390;Zsc=4.61;l=5;
lo=(3*(10^8))/f; //lo=wavelength
Zo=sqrt(Zoc*Zsc);
```

```
printf("-Characteristic impedance = %f ohms\n",round
   (Zo));
er=(lo/l)^2; //er=relative permittivity of
   dielectric
printf("-Relative permittivity of dielectric = %f\n"
   ,er);
P=fix(((atanh(sqrt(Zsc/Zoc)))/l)*10000)/10000;
a=P*8.686;
printf("-a = %f db/m\n",fix(a*10000)/10000);
Vp=(3*(10^8))/(sqrt(er)*10^8);
printf("-Velocity of propagation = %f * 10^8 m/sec",
   Vp);
//the difference in result of attenuation constant
   is due to erroneous value in textbook.
disp("The difference in result of the value of
   attenuation constant is due to erroneous value in
    textbook")
```

Chapter 3

Line with any Termination

Scilab code Exa 3.1 Calculating sending end voltage

```
clear;
clc;
l=10;a=0.03;b=0.03;Vr=4;f=1000;
P=a+(%i*b);
Vs=Vr*(exp(P*l));
C=real(Vs);
D=imag(Vs);
printf("Sending end voltage is Vs = %f /_ %f volts",
   fix(abs(Vs)*100)/100,round(atan(D,C)*180/%pi));
```

Scilab code Exa 3.2 Calculating rms value voltage and current

```
clear;
clc;
Zo=50;Vr=10;Ir=0;a=0;
B=%pi/4;
V=(Vr*cos(B))+(%i*(Ir*Zo*sin(B)));
I=(Ir*cos(B))+(%i*((Vr*sin(B))/Zo));
```

```
C=real(I);
D=imag(I);
printf("–R.m.s.voltage at the required distance is V
   = %f volts\n",round((V)*100)/100);
printf("–R.m.s.current at required distance is I =
   %f /_ %f  Amps",round(abs(I)*1000)/1000,atan(D,C)
   *180/%pi);
```

Scilab code Exa 3.3 Calculating the rms value of voltage and current

```
clc;
Zo=500*exp(%i*(-%pi/(180/43)));l=10;P=0.07+(%i*0.08)
   ;Vs=5;
V=Vs*(exp(-P*l));
I=V/Zo;
C=real(V);
D=imag(V);
E=real(I)*(10^3);
F=imag(I)*(10^3);
printf("–R.m.s.voltage at the required distance is V
   = %f /_ %f V\n",round(abs(V)*10)/10,round(atan(D
   ,C)*180*10/%pi)/10);
printf("–R.m.s.current at required distance is I =
   %f/_ %f  mA",round(abs(I)*10^3),round(atan(F,E)
   *180*10/%pi)/10);
```

Scilab code Exa 3.4 Calculating the input impedance of line

```
clear;
clc;
Zo=710*exp(%i*(%pi/(180/14)));l=100;P=0.007+(%i
   *0.028);Zr=300;
```

```
K=round(sinh(P*l)*100)/100;J=round(cosh(P*l)*100)
   /100;
Zin=Zo*((Zr*J)+(Zo*K))/((Zo*J)+(Zr*K));
C=real(Zin);
D=imag(Zin);
printf("Input impedance of line is Zin = %f /_ %f
   ohms",abs(Zin),round(atan(D,C)*180*10/%pi)/10);
A=((Zr*J)+(Zo*K));
B=((Zo*J)+(Zr*K));
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Scilab code Exa 3.5 Calculating sending end impedance

```
clear;
clc;
Zo=55;Zr=115+(%i*75);n=1.183;
//value of Zr as taken in solution
m=(2*%pi*n)-(2*%pi);
Zin=Zo*(Zr+(%i*Zo*tan(m)))/(Zo+(%i*Zr*tan(m)));
C=real(Zin);
D=imag(Zin);
printf("Sending end impedance is Zin = %f + j(%f)
   ohms",fix(C*10)/10,fix(D*10)/10);
```

Scilab code Exa 3.6 Calculating input impedance

```
clear;
clc;
Zo=50*exp(-%i*(%pi/(180/5)));a=0.001;b=%pi/1.8;Vr
   =0.5*exp(%i*(%pi/(180/30)));
```

```
l=100;x=4; //Vr=voltage reflection coefficient
modVr=abs(Vr);
A=modVr*exp(-2*a*(l-x));
Zin=Zo*(1+A)/(1-A);
C=real(Zin);
D=imag(Zin);
printf("Input impedance of transmission line = %f /_
    %f ohms",fix(abs(Zin)*100)/100,atan(D,C)*180/%pi
   );
```

Scilab code Exa 3.7 Calculating the value of voltage reflection coefficient

```
clear;
clc;
Ka=0.2*exp(%i*(-%pi/(180/30)));d=1/12;
Kb=Ka/exp(%i*%pi*4*d);
printf("Desired value of voltage reflection
   coefficient = %f /_ %f",abs(Kb),atan(imag(Kb),
   real(Kb))*180/%pi);
```

Scilab code Exa 3.8 Calculating the ratio of transmitter voltage to receiver voltage

```
clear;
clc;
Zo=710*exp(%i*(-%pi/(180/16)));a=0.01;b=0.035;l=100;
   Zr=300;
P=a+(%i*b);
V=(cosh(P*l))+((Zo/Zr)*sinh(P*l)); //V=Vs/Vr
modV=abs(V);
V1=log(modV)*8.686;
V2=log(modV);
```

```
printf("-Ratio of the transmitter voltage and the
   receiver voltage in nepers = %f\n",round((V2)
   *100)/100);
printf("-Ratio of the transmitter voltage and the
   receiver voltage in decibels = %f\n",round((V1)
   *100)/100);
```

Scilab code Exa 3.9 Calculating the value of voltage reflection coefficient

```
clear;
clc;
Zo=692*exp(%i*(-%pi/(180/12 )));Zr=200;l=100;V=1;f
   =1000;
K=(Zr-Zo)/(Zr+Zo);
C=real(K);
D=imag(K);
printf("Voltage reflection coefficient = %f /_ %f",
   round(abs(K)*100)/100,round(atan(D,C)*180*10/%pi)
   /10);
```

Scilab code Exa 3.10 Calculating power delivered at receiving end

```
clear;
clc;
f=1000;l=1000;R=10.4;L=0.00367;G=0.8*(10^-6);C
   =0.00835*(10^-6);Es=10;
//value of Es as taken in solution
w=2*%pi*f;
Z=R+round((%i*w*L));
Y=G+(%i*w*C);
Zo=sqrt(Z/Y);
P=sqrt(Z*Y);
Is=Es/Zo;
```

```
Ir=Is*exp(-P*l);
P=((abs(Ir))^2)*real(Zo);
printf("—Power delivered at receiving end = %f micro
    —watt",P*(10^6));
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Scilab code Exa 3.11 Calculating sending end current and receiving end current and voltage and efficiency of transmission

```
clear;
clc;
l=50;Zo=692*exp(%i*(-%pi/(180/12)));a=0.00755;b
    =0.0355;Zr=300;f=1000;Vs=10;
P=a+(%i*b);
Zin=Zo*((Zr*cosh(P*l))+(Zo*sinh(P*l)))/((Zo*cosh(P*l
    ))+(Zr*sinh(P*l)));
Is=Vs/Zin*(10)^-1;
A=real(Is);
B=imag(Is);
printf("—Sending end current = %f /_ %f Amp\n",round
    (abs(Is)*10^5)/10^5,round(atan(B,A)*180*10/%pi)
    /10);
Vr=Vs*((cosh(P*l))-((Zo*sinh(P*l))/Zin));
C=real(Vr);
D=imag(Vr);
printf("—Receiving end voltage = %f /_ %f volts\n",
    round(abs(Vr)*10)/10,round(atan(D,C)*180/%pi));
Ir=Vr/Zr;
E=real(Ir);
F=imag(Ir);
printf("—Receiving end current = %f /_ %f Amp\n",
    round(abs(Ir)*1000)/1000,round(atan(F,E)*180/%pi)
    );
modIr=abs(Ir);
Rr=Zr;
Pr=((modIr)^2)*Rr;
theta=atan((imag(Zin))/(real(Zin)));
modIs=abs(Is);
Ps=Vs*modIs*cos(theta);
n=(Pr/Ps)*10;
printf("-Efficiency of transmission(in percentage)=
    %f",round(n));
```

Scilab code Exa 3.12 Calculating the resistance and inductance of the series and shunt elements

```
clear;
clc;
l=10;Zo=280*exp(-%i*(%pi/(180/30)));P=0.08*exp(%i*(
    %pi/(180/40)));f=5000/(2*%pi);
w=2*%pi*f;
Z1=Zo*tanh(P*l/2);
Rs=real(Z1);
Ls=imag(Z1)/w;
printf("-Resistance of series element = %f ohms\n",
    fix(Rs));
printf("-Inductance of series element = %f mH\n",
    round(Ls*(10^3)*100)/100);
Z2=Zo/(sinh(P*l));
Rsh=real(Z2);
Csh=-imag(Z2)/w;
printf("-Resistance of shunt element = %f ohms\n",
    round(Rsh*10)/10);
printf("-Capacitance of shunt element = %f
    microfarads",fix(Csh*1000)/1000);
```

Scilab code Exa 3.13 Determining the characteristic impedance and propagation constant

```
clear;
clc;
l=5;f=5000/(2*%pi);Rs=175;Ls=10*(10^-3);Rsh=270;Csh
   =0.2*(10^-6);
w=2*%pi*f;
Z1=(Rs+(%i*w*Ls)); //Z1=Z1/2
Z2=Rsh-(%i/(w*Csh));
t=sqrt((Z1)/((Z1)+(2*Z2)));
P=2*(atanh(t))/l;
A=real(P);
B=imag(P);
printf("Propagation constant = % f + %f per loop km\
   n",round(A*100)/100,round(B*100)/100);
Zo=Z1/(tanh((P*l)/2));
C=real(Zo);
D=imag(Zo);
printf("Characteristic impedance = %f /_ %f ohms",
   round(abs(Zo)*1000)/1000,round(atan(D,C)*180/%pi)
   );
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Scilab code Exa 3.14 Calculating voltage and current

```
clear;
clc;
a=0;Zo=50;Vr=100;l=50;y=10;Ir=0;f=20*(10^6);
```

```
lo=300/f; //lo=lambda
b=2*%pi/lo;
P=a+(%i*b);
V=(Vr*cosh(P*y))+(Ir*Zo*sinh(P*y));
printf("-Voltage at the required point = %f /_ %f
   volts\n",abs(V),atan(imag(V),real(V))*180/%pi);
I=((Vr/Zo)*sinh(P*y))+(Ir*cosh(P*y));
A=real(I);
B=imag(I);
printf("-Current at required point = %f /_ %f Amps",
   round(abs(I)*1000)/1000,atan(B,A)*180/%pi);
```

Scilab code Exa 3.15 Calculating input impedance

```
clear;
clc;
Zo=600;Zr=%i*150;l=0.25;f=300*(10^6);
lo=300/f; //lo=lambda
b=2*%pi/lo;
Zin=Zo*Zo/Zr;
A=real(Zin);
B=imag(Zin);
printf("Zin = %f + j(%f) ohms",A,B);
```

Scilab code Exa 3.16 Calculating power consumed in load

```
clear;
clc;
f=7.5*(10^3);R=2.6;L=2.4*(10^-3);C=.0078*(10^-6);G
   =.11*(10^-6);l=50;Vs=10;
w=2*%pi*f;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
```

```
Zo=sqrt(Z/Y);
P=sqrt(Z*Y);
Is=Vs/Zo;
Ir=Is*(exp(-P*l));
Ir1=abs(Ir);
Rr=real(Zo);
Pr=Ir1*Ir1*Rr;
printf("Power consumed in load is Pr = %f mW",round(
   Pr*(10^3)*10)/10);
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Scilab code Exa 3.17 Calculating the exact and approximate distance to fault

```
clear;
clc;
R=10.4;L=0.0036;C=0.0083*(10^-6);G=0;f1=1420;f2
   =1860;
Favg=(f1+f2)/2;
w=2*%pi*Favg;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
P=sqrt(Z*Y);
a=real(P);
b=imag(P);
v=w/b;
d=v/(2*(f2-f1));
printf("-Approximate distance = %f km\n",round(d));
w1=2*%pi*f1;
Z1=R+(%i*w1*L);
Y1=G+(%i*w1*C);
P1=sqrt(Z1*Y1);
```

```
a1=real(P1);
b1=imag(P1);
v1=w1/b1;
w2=2*%pi*f2;
Z2=R+(%i*w2*L);
Y2=G+(%i*w2*C);
P2=sqrt(Z2*Y2);
a2=real(P2);
b2=imag(P2);
v2=w2/b2;
D=(v1*v2)/(2*((v1*f2)-(v2*f1)));
printf("-Exact distance = %f km",round(D*10)/10);
```

Scilab code Exa 3.18 Calculating the reflection factor and reflection loss

```
clear;
clc;
Z1=300;Zo=650*exp(%i*(-%pi/(180/12)));Z2=Zo;
Z=Z1+Z2;
K=(2*sqrt(Z1*abs(Z2)))/abs(Z);
printf("-Reflection factor = %f\n",round(K*1000)
   /1000);
L=20*(log10(1/K));
printf("-Reflection loss = %f db",round(L*100)/100);
//the difference in result of reflection loss is due
    to erroneous value in textbook.
disp("The difference in result of reflection loss is
    due to erroneous value in textbook")
```

Scilab code Exa 3.19 Calculating reflection coefficient and input impedance

```
clear;
clc;
```

```
Zo=600;a=0.01;b=0.03;l=100;Zr=200;
P=a+(%i*b);
K=(Zr-Zo)/(Zr+Zo);
printf("Reflection coefficient = %f /_- %f\n",abs(K)
    ,atan(imag(K),real(K))*180/%pi);
k=((abs(K))*exp(-2*P*l));
b=(1+k)/(1-k);
Zin=Zo*b;
C=real(Zin);
D=imag(Zin);
printf("Input impedance Zin = %f + j(%f) ohms",round
    (C*1000)/1000,round(D*100)/100);
//Zin has been wrongly calculated in the book
```

Scilab code Exa 3.20 Finding reflection loss

```
clc;
Zo=632.8;Zr=80;
F=20*log10((2*Zr*Zo)/(Zo+Zr));
printf("Reflection loss = %f db",round(F*100)/100);
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Chapter 4

Power Lines

Scilab code Exa 4.2 Determining equivalent pi network

```
clear;
clc
l=225;Zo=401-(%i*29);P=(0.148+(%i*2.06))*(10^-3);
Zs=Zo*sinh(P*l);
A=real(Zs);
B=imag(Zs);
printf("−Series branch of the equivalent network
   will have the impedance = %f + j(%f) ohms\n",
   round(A),round(B));
Zsh=Zo*coth(P*l/2);
C=real(Zsh);
D=imag(Zsh);
printf("−Shunt branch of the equivalent network will
    have the impedanc = j(%f) ohms",round(D))
```

Scilab code Exa 4.3 Calculating inductance per km of line

```
clear;
```

```
clc;
r=1.25;d1=75;d2=150;
r1=0.7718*r;
ds=nthroot((r1*r1*d1*d1),4);
dm=sqrt(d2*(sqrt((d2*d2)+(d1*d1))));
L=0.9212*log10(dm/ds);
printf("Total loop inductance = %f mH/loop km",round
   (L*100)/100);
```

Scilab code Exa 4.4 Calculating characteristic impedance and propagation constant

```
clear;
clc;
l=440;L=2.2*(10^-3);C=0.0136*(10^-6);R=0.120;G=0;f
   =60;
w=2*%pi*f;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
Zo=sqrt(Z/Y);
A=real(Zo);
B=imag(Zo);
printf("-Characteristic impedance = %f + j(%f) ohms\
   n",round(A),round(B));
P=sqrt(Z*Y);
E=real(P)*10^4;
F=imag(P)*10^3;
printf("-Propagation constant = %f * 10^-4 + j(%f) *
    10^-3 per km",fix(E*100)/100,fix(F*100)/100);
```

Scilab code Exa 4.5 Calculating inductive reactance

```
clear;
```

```
clc;
f=50;r=5*(10^-3);x=.5;y=3;z=4.5;t=6;s=5;
r1=0.7788*r; //r1=GMR
Dab=round(sqrt((y^2)+(x^2))*1000)/1000;
Dab1=round(sqrt((y^2)+(s^2))*1000)/1000;
Daa=sqrt((t^2)+(z^2));
Dab2=Dab;
Dab3=Dab1;
dab=round(nthroot((Dab1*Dab3*Dab*Dab2),4)*100)/100;
dca=fix(nthroot((t*t*z*z),4)*100)/100;
ds1=nthroot((r1*r1*7.5*7.5),4);
ds2=nthroot((r1*r1*5.5*5.5),4);
ds3=ds1;
ds=round(nthroot((ds1*ds2*ds3),3)*1000)/1000;
La=fix(0.4606*log10(dca/ds)*100)/100;
X=2*3*f*La*10^-3;
printf("Inductive reactance = %f ohm/km/phase",X);
```

Scilab code Exa 4.6 Calculating capacitive susceptance

```
clear;
clc;
dia=2*(10^-2);f=50;d=3.5;
r=dia/2;
Cab=0.01207/(log10(d/r))*(10^-6);
Xc=1/(2*%pi*f*Cab);
Bc=(1/Xc)*10^6;
printf("Capacitive susceptance per km = %f * 10^-6
    mho/km",round(Bc*100)/100);
```

Scilab code Exa 4.7 Determining the capacitance per km and charging current

```
clear;
clc;
dia=2*(10^-2);f=50;d1=3.5;V=66000;
r=dia/2;
Cn1=0.02414/(log10(d1/r))*(10^-6);
printf("(a)Capacitance per km = %f microfarads/km,to
    neutral\n",fix(Cn1*(10^6)*10^5)/10^5);
Vn=V/sqrt(3);
Ic1=Vn*2*%pi*f*Cn1;
printf("\tCharging current = %f mA/km\n",fix(Ic1
   *(10^3)*10)/10);
d1=3.5;d2=4.5;d3=5.5;
Deq=nthroot(d1*d2*d3,3);
Cn2=0.02414/(log10(Deq/r))*(10^-6);
printf("(b)Capacitance per km =%f microfarads/km,to
   neutral\n",fix(Cn2*(10^6)*10^5)/10^5);
Ic2=Vn*2*%pi*f*Cn2;
printf("\tCharging current = %f mA/km",round(Ic2
   *(10^3)));
```

Chapter 5

Telegraph and Telephone Lines

Scilab code Exa 5.1 Calculating decrease in attenuation and cut off frequency

```
clear;
clc;
R=28;C=0.04*(10^-6);G=0;L=0;f=1600;s=2000;r=3.7;l
   =88*(10^-3);
Rc=R+(r/(s*(10^-3)));
Lc=L+(l/(s*(10^-3)));
Fc=1/(%pi*(sqrt(Lc*C)));
printf("-Cutoff frequency = %f Hz\n",round(Fc));
w=2*%pi*f;
a=sqrt(w*C*R/2);
al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
A=(a-al)*8.686;
printf("-Decrease in attenuation = %f db/km",round(A
   *100)/100);
```

Scilab code Exa 5.2 Calculating Zo and a and b and Vp

```
clear;
clc;
R=10.15;L=3.93*(10^-3);G=0.29*(10^-6);C
    =0.008*(10^-6);w=5000;r=7.3;l=246*(10^-3);
s=7.88;
Rc=R+(r/s);
Lc=L+(l/s);
al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
printf("-a = %f neper/km\n",round(al*10000)/10000);
b=w*(sqrt(Lc*C));
printf("-b = %f radians/km\n",round(b*10^4)/10^4);
lo=2*%pi/b; //lo=lambda
printf("-lo = %f km\n",round(lo*100)/100);
Vp=(w/b)*10^-4;
printf("-Vp = %f * 10^4 km/sec\n",round(Vp*100)/100)
    ;
Zo=(sqrt(Lc/C))*10^-3;
printf("-Zo = %f * 10^3 ohms",fix(Zo*100)/100);
```

Scilab code Exa 5.3 Calculating modified Zo and a and Vp

```
clear;
clc;
R=10.4;L=3.67*(10^-3);G=0.8*(10^-6);C
    =0.00835*(10^-6);w=5000;r=7.3;l=246*(10^-3);
s=7.88;
Rc=R+(r/s);
Lc=L+(l/s);
Z=fix((Rc+(%i*w*Lc))*100)/100;
Y=G+(%i*w*C);
Zo=sqrt(Z/Y);
A=real(Zo);
B=imag(Zo);
P=sqrt(Z*Y);
a=real(P);
```

```
b=imag(P);
Vp=w/b;
phi=(round(atan(imag(Z),real(Z))*180*10/%pi)/10)-
   round(atan(imag(Y),real(Y))*180*10/%pi)/10;
printf("(i)Using exact method Zo = %f /_ %f ohms\n",
   round(abs(Zo)),phi);
printf("\t\t\ta = %f neper/km\n",round(a*10^4)/10^4)
   ;
printf("\t\t\tVp = %f * 10^4 km/sec\n",round(Vp
   *10^-2)/10^2);
Zo1=sqrt(Lc/C);
printf("(ii)Using approximate methos Zo = %f ohms\n"
   ,round(Zo1));
al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
printf("\t\t\ta = %f neper/km\n",round(al*10^5)
   /10^5);
b1=w*(sqrt(Lc*C));
Vp1=1/(sqrt(Lc*C));
printf("\t\t\tVp = %f * 10^4 km/sec",round(Vp1
   *10^-2)/10^2);
```

Scilab code Exa 5.4 Calculating the phase velocity before and after velocity and decrease in attenuation and cut off frequency

```
clear;
clc;
R=10;C=0.04*(10^-6);L=0;G=0;f=1000;l=100*(10^-3);r
   =12;d=0.9;
w=2*%pi*f;
a=sqrt(w*C*R/2);
b=sqrt(w*C*R/2);
Vp=(w/b)*10^-5;
printf("-Phase velocity before loading = %f * 10^5
   km/sec\n",fix(Vp*10)/10);
Rc=R+(r/d);
```

```
Lc=L+(l/d);
al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
Vpl=(1/(sqrt(Lc*C)))*10^-4;
printf("-Phase velocity after loading = %f * 10^4 km
   /sec\n",Vpl);
A=a-al;
printf("-Decrease in attenuation = %f neper/km\n",
   round(A*10000)/10000);
Fc=1/(%pi*(sqrt(Lc*C*d)));
printf("-Cutoff frequency = %f kHz",round(Fc*(10^-3)
   *1000)/1000);
```

Scilab code Exa 5.5 Calculating attenuation and cut off frequency

```
clear;
clc;
R=40;C=0.06*(10^-6);L=0;G=0;r=15;l=80*(10^-3);d=1.2;
   f=(5/(2*%pi))*10^3;
Rc=R+(r/d);
Lc=L+(l/d)
w=2*%pi*f;
Z=Rc+(%i*w*Lc);
Y=G+(%i*w*C);
P=sqrt(Z*Y);
a=real(P);
printf("(a)Attenuation constant a = %f neper/km\n",
   round(a*1000)/1000);
Fc=1/(%pi*(sqrt(l*C*d)));
printf("(b)Cutoff frequency = %f kHz",round(Fc
   *(10^-3)*1000)/1000);
```

Scilab code Exa 5.6 Calculating attenuation and highest frequency of transmission

```
clear;
clc;
R=25;C=0.04*(10^-6);L=0.6*(10^-3);G=0;l=45*(10^-3);r
    =20;d=1;f=1000;
Rc=R+(r/d);
Lc=L+(l/d)
w=2*%pi*f;
Z=Rc+(%i*w*Lc);
Y=G+(%i*w*C);
P=sqrt(Z*Y);
a=real(P);
printf("-Attenuation per km line = %f neper/km\n",
    round(a*10^5)/10^5);
Fc=1/(%pi*(sqrt(Lc*C*d)));
printf("-Highest frequency of transmission will be
    cutoff frequency = %f kHz",round(Fc*(10^-3)*10^3)
    /10^3);
```

Scilab code Exa 5.7 Calculating the value of Zo and a and Vp after loading

```
clear;
clc;
R=42.1;L=1*(10^-3);G=1.5*(10^-6);C=0.062*(10^-6);f
    =796;r=2.7;l=31*(10^-3);
d=1.135;
Rc=R+(r/d);
Lc=round((L+(l/d))*1000)/1000;
Zo=sqrt(Lc/C);
printf("-Zo after loading = %f ohms\n",round(Zo));
al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
printf("-a after loading = %f neper/km\n",round(al
    *10^4)/10^4);
Vp=1/(sqrt(Lc*C));
printf("-Approximate value of Vp = %f km/sec",fix(Vp
));
```

Scilab code Exa 5.8 Calculating cut off frequency and attenuation

```
clear;
clc;
R=30;L=0;G=0;C=7000*(10^-12);f=3/(2*%pi)*(10^3);r
   =35;l=70*(10^-3);
d=1;
Rc=R+(r/d);
Lc=L+(l/d);
Fc=1/(%pi*(sqrt(Lc*C*d)));
printf("-Approximate value of highest frequency = %f
    kHz\n",round(Fc*(10^-3)*100)/100);
al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
printf("-Approximate value of attenuation per km
   after = %f neper/km",round(al*10^4)/10^4);
```

Scilab code Exa 5.9 Calculating Zo and a and b

```
clear;
clc;
R=44;L=0.001;G=1*(10^-6);C=0.065*(10^-6);f=1600;r
   =3.7;l=88*(10^-3);
d=1.136;
w=2*%pi*f;
Rc=R+(r/d);
Lc=L+(l/d);
Zo=sqrt(Lc/C);
printf("-Approximate value of Zo = %f ohms\n",round(
   Zo*100)/100);
al=(((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C)))*8.686;
```

```
printf("-Approximate value of a = %f db/km\n",fix(al
    *1000)/1000);
b=w*(sqrt(Lc*C));
printf("-Approximate value of b = %f radians/km",fix
    (b*1000)/1000);
//the difference in result of Zo is due to erroneous
     value in textbook.
disp("The difference in result of Zo is due to
    erroneous value in textbook")
```

Scilab code Exa 5.10 Calculating length of line for a given attenuation

```
clear;
clc;
a=0.005;f=1000;al=0.05;att=10 //att=given
    attenuation in db
lo=att/(a*8.686);
printf("-For the open wire line the required length
    is lo = %f km\n",round(lo*10)/10);
lc=att/(al*8.686);
printf("-For the cable the required length is lc =
    %f km",fix(lc*100)/100);
```

Scilab code Exa 5.11 Calculating cutoff frequency and attenuation and phase velocity

```
clear;
clc;
R=20;C=0.03*(10^-6);L=0;G=0;r=18.2;l=78*(10^-3);d
    =2.1;f=3000/(2*%pi);
Rc=R+(r/d);
Lc=L+(l/d);
Fc=1/(%pi*(sqrt(Lc*C*d)));
```

```
printf("(i)Cutoff frequency = %f kHz\n",round(Fc
   *(10^-3)*100)/100);
al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
printf("(ii)Attenuation per km = %f neper/km\n",fix(
   al*10000)/10000);
Vp1=(1/(sqrt(Lc*C)))*10^-4;
printf("(iii)Phase velocity = %f * 10^4 km/sec",fix(
   Vp1*1000)/1000);
//the difference in result of attenuation per km and
    phase velocity is due to erroneous value in
   textbook.
disp("The difference in result of attenuation per km
    and phase velocity is due to erroneous value in
   textbook")
```

Scilab code Exa 5.12 Finding the value of the loading coil inductance and the interval at which it needs to be placed

```
clear;
clc;
R=80;C=0.06*(10^-6);L=0;G=0;f=2500;al=0.04;ratio=40;
fc=2*f;
//ratio=ratio off resistance to inductance of
   loading coil
A=2/(3.14*3.14*C*4*f*f); //A=Lc*d
B=al/(%pi*f*1.414*C/2);
C=%pi*f*1.414*C/2;
y=poly([ratio*C*100*A -al*100 80*100*C],"x","coeff")
   ;
a=roots(y);
b=round(a(1,1)*100)/100;
Lo=round((A/b)*1000)/1000;
printf("Adding %f Henry coils at intervals of %f km
   is the practical and economically possible
   solution.",Lo,b)
```

Scilab code Exa 5.13 Calculating rate of transmission of information

```
clear;
clc;
B=3000; //B=band width
D=30; //D=ratio of signal power to noise power in db
P=10^(D/10); // P=converting D into a numerical
   ratio
A=fix(log2(1+P)*100)/100
C=B*A;
printf("Rate of transmission of information = %f
   bits/sec",C);
```

Scilab code Exa 5.14 Calculating a and b and cut off frequency

```
clear;
clc;
R=55;L=0.6*(10^-3);G=1*(10^-6);C=0.04*(10^-6);f=800;
   r=8;l=0.1;d=2.5; //value of l(loading coil
   inductance) as taken in solution
w=2*%pi*f;
Z=round(R+(%i*w*L));
Y=G+(%i*w*C);
Zo=sqrt(Z/Y);
P=round(sqrt(Z*Y)*10^4)/10^4;
Zc=r+(%i*w*l);
A=fix(((cosh(P*d))+(Zc*(sinh(P*d))/(2*Zo)))*10^3)
   /10^3;
Pl=(acosh(A))/d;
al=real(Pl);
bl=imag(Pl);
```

```
printf("New value of attenuation constant = %f neper
    /km\n",round(al*100)/100);
printf(" New value of phase constant = %f rad/km\n",
    round(bl*1000)/1000);
Lc=L+(1/(d*10^3));
Fc=1/(3.14*(sqrt(Lc*C*d)));
printf(" Cutoff frequency = %f * 10^4 Hz",round(Fc
    *10^-2)/10^2);
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Scilab code Exa 5.15 Calculating wavelength of line

```
clear;
clc;
R=10;C=0.008*(10^-6);l=60; //value of R as taken in
    solution
Rt=R*l; //Rt=total resistannce
Ct=C*l; //Ct=total capacitance
f=1600 //assupmtion
w=2*%pi*f;
b=sqrt(f*Rt*Ct/2);
lo=2*%pi/b;
printf("Wavelength of line = %f metres",fix(lo*100)
    /100);
```

Scilab code Exa 5.16 Calculating the value of attenuation

```
clear;
clc;
```

```
R=45;L=1.2*(10^-3);G=5*(10^-6);C=0.065*(10^-6);w
   =20000;l=22*(10^-3);s=1.1;
pf=0.005//pf=power factor //value of pf as taken in
   solution
r=pf*w*L;
Rc=R+(r/s);
Lc=L+(l/s);
P=sqrt((Rc+(%i*w*Lc))*(G+(%i*w*C)));
theta=round(atan(imag(P),real(P))*180/%pi);
a=abs(P)*cos(theta*%pi/180);
printf("Attenuation constant of line = %f neper/km",
   fix(a*10^4)/10^4);
```

Scilab code Exa 5.17 Calculating potential at mid point

```
clear;
clc;
l=100;R=10;G=10^-5;Vs=40;Zr=0;
Zo=sqrt(R/G);
P=sqrt(R*G);
Zin=Zo*(Zr+(Zo*tanh(P*l)))/(Zo+(Zr*tanh(P*l)));
Is=Vs/Zin;
V=(Vs*(cosh(P*l)))-(Is*Zo*(sinh(P*l)));
Vm=2*V;
printf("Potential at mid point = %f volts",Vm);
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Chapter 6

Ultrahigh Frequency Lines

Scilab code Exa 6.1 Calculating complex reflection coefficient and terminal impedance

```
clear;
clc;
f=200*(10^6);s=4.48;ymin=6;Zo=300; //s=standing wave
    ratio
lo=300/(f*(10^-6)); //where f is in megahertz ,lo=
   wavelength of wave in air
b=2*%pi/lo;
phi=(2*b*ymin*(10^-2))-%pi;
ampK=(s-1)/(s+1); //ampK=amplitude of the reflection
    coefficient
K=ampK*(exp(%i*phi));
A=real(K);
B=imag(K);
printf("-Complex reflection coefficient= %f /_ %f\n"
   ,round(abs(K)*1000)/1000,round(atan(B,A)*180/%pi)
   );
ZR=(Zo*(1+K))/(1-K);
C=real(ZR);
D=imag(ZR);
printf("-Terminating impedance of line = %f /_ %f
ohms",abs(ZR),round(atan(D,C)*180*10/%pi)/10);
```

Scilab code Exa 6.2 Calculating value of load impedance

```
clear;
clc;
Zo=75;s=3;d=1/5;
B=2*%pi*d; //B=b*ymax where ymax=position of the
   current maxima which is 1/5th
   wavelength away from the load(here)
phi=2*B;
ampK=(s-1)/(s+1); //ampK=amplitude of the reflection
    coefficient
K=ampK*(exp(%i*phi));
ZR=round(((Zo*(1+K))*100)/100)/(round((1-K)*1000)
   /1000);
C=real(ZR);
D=imag(ZR);
printf("Load impedance = %f /_ %f ohms",round(abs(ZR
   )*10)/10,round(atan(imag(ZR),real(ZR))*180*100/
   %pi)/100);
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Scilab code Exa 6.3 Calculating VSWR and position of voltage minimum nearest to load

```
clear;
clc;
Zo=50;f=300*(10^6);ZR=50+(%i*50);
```

```
lo=300/(f*(10^-6)); //where f is in megahertz ,lo=
    wavelength of wave in air
K=(ZR-Zo)/(ZR+Zo);
ampK=sqrt((real(K)^2)+(imag(K)^2));
S=(1+ampK)/(1-ampK);
printf("—VSWR = %f\n",round(S*100)/100);
phi=atan(imag(K)/real(K));
ymax=phi*lo/(2*2*%pi);
ymin=ymax+(lo/4);
printf("—Position of voltage minimum nearest load =
    %f metres",round(ymin*10000)/10000);
```

Scilab code Exa 6.4 Calculating standing wave ratio

```
clear;
clc;
Zo=400;ZRa=70,ZRb=800;ZRc=650-(%i*475);
Ka=(ZRa-Zo)/(ZRa+Zo);
ampKa=sqrt((real(Ka)^2)+(imag(Ka)^2));
Sa=(1+ampKa)/(1-ampKa);
printf("(a)Standing wave ratio = %f\n",round(Sa*100)
    /100);
Kb=(ZRb-Zo)/(ZRb+Zo);
ampKb=sqrt((real(Kb)^2)+(imag(Kb)^2));
Sb=(1+ampKb)/(1-ampKb);
printf("(b)Standing wave ratio = %f\n",Sb);
Kc=(ZRc-Zo)/(ZRc+Zo);
ampKc=sqrt((real(Kc)^2)+(imag(Kc)^2));
Sc=(1+ampKc)/(1-ampKc);
printf("(c)Standing wave ratio = %f",round(Sc*1000)
    /1000);
```

Scilab code Exa 6.5 Calculating value of load impedance

```
clear;
clc;
s=2;f=300*(10^6);lo=1;ymin=0.8; //lo=wavelength
ampK=(s-1)/(s+1);
b=2*%pi/lo;
phi=(2*b*ymin)-%pi;
K=ampK*(exp(%i*phi));
Zr=(1+K)/(1-K);
A=real(Zr);
B=imag(Zr);
printf("Value of load impedance = %f /_ %f ohms",
   round(abs(Zr)*10)/10,fix(atan(B,A)*180*100/%pi)
   /100);
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Scilab code Exa 6.11 Calculating point of attachment and length of stub

```
clear;
clc;
ZR=100;Zo=600;f=100*(10^6);
lo=300/(f*(10^-6)); //lo=wavelength
Ls=(lo/(2*%pi))*(atan(sqrt(ZR/Zo)));
printf("-Point of attachment = %f cms\n",round(Ls
   *(10^2)*10)/10)
Lt=(lo/(2*%pi))*(%pi+(atan((sqrt(ZR*Zo))/(ZR-Zo))));
printf("-Length of the short circuited stub = %f cms
   ",round(Lt*(10^2)));
```

Scilab code Exa 6.14 Calculating point of attachment and length of stub

```
clear;
clc;
f=10^9;K=0.5*exp(%i*(30)/(180/%pi));
lo=300/(f*(10^-6)); //lo=wavelength
ampK=abs(K);
phi=atan(imag(K)/real(K));
Ls=(lo/(4*%pi))*(phi+%pi-acos(ampK));
printf("-Position of stub = %f cm\n",Ls*(10^2));
Lt=(lo/(2*%pi))*(atan(sqrt(1-(ampK*ampK)))/(2*ampK))
    ;
printf("-Length of the stub = %f cm",round(Lt*(10^2)
    *100)/100);
```

Scilab code Exa 6.15 Calculating point of attachment and length of stub

```
clear;
clc;
Zo=400;ZR=200-(%i*100);lo=3; //lo=wavelength
//value os Zo as taken in solution
K=(ZR-Zo)/(ZR+Zo);
ampK=abs(K);
phi=%pi + atan(imag(K)/real(K));
Ls=(lo/(4*%pi))*(phi+%pi-acos(ampK));
printf("Shortest distance from the lead to the stub
    location = %f metres\n",round(Ls*100)/100);
Lt=(lo/(2*%pi))*(atan(sqrt(1-(ampK*ampK)))/(2*ampK))
    ;
printf("Length of the short circuited stub = %f
    metres",fix(Lt*10)/10);
```

Scilab code Exa 6.17 Calculating the dimensions of a quarter wave line

```
clear;
```

```
clc;
ZR=300;s=9;d=0.1
r=d/2;
Zof=276*log10(s/r);
Zoq=sqrt(ZR*Zof);
do=(s*2)/10^(Zoq/276);
printf("Diameter of wire used = %f cm",fix(do*10)
    /10);
```

Scilab code Exa 6.18 Designing a quarter wave transformer

```
clear;
clc;
Zin=36;Zt=500;f=40;x=0.97
Zo=sqrt(Zin*Zt);
A=10^(Zo/276);
lo=300/f;
l1=lo*x/4;
printf("-The characteristic impedance of the
    transmission line = %f ohms\n",round(Zo));
printf("-The spacing between the conductors shud be
    %f times the radius of the conductor\n",round(A))
    ;
printf("-The length of the quarter wavelength
    transformer must be %f metres",round(l1*100)/100)
    ;
```

Scilab code Exa 6.19 Designing a single stub matching given system to eliminate standing wave ratio

```
clear;
clc;
f=150;S=4.48;Ymin=6*(10^-2);
```

```
lo=300/(f); //lo=wavelength
b=(2*%pi)/lo;
phi=round(((2*b*Ymin)-%pi)*100)/100;
phi1=-phi;
ampK=round(((S-1)/(S+1))*10)/10;
Ls=(lo/(4*%pi))*(phi1+%pi-round(acos(ampK)));
printf("Point of attachment = %f cm\n",round(Ls
    *(10^4))/100);
Lt=(lo/(2*%pi))*(atan(sqrt(1+(ampK*ampK)))/(2*ampK))
    ;
printf("Length of the stub = %f cm",round(Lt*(10^4))
    /100);
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Scilab code Exa 6.20 calculating frequency and terminated impedance

```
clear;
clc;
s=3.3;Zo=300;l=15;
ampK=round(((s-1)/(s+1))*100)/100;
Zr=Zo*(1+ampK)/(1-ampK);
printf("-Terminated impedance = %f ohms\n",fix(Zr));
lo=(2*2*%pi*l*(10^-2))/%pi; //lo=wavelength
f=300/lo;
printf("-Frequency = %f MHz",f);
```

Scilab code Exa 6.21 Calculating load and input impedance

```
clear;
clc;
```

```
Ymin=18*(10^-2);S=2.5;dmin=20*(10^-2);l=52*(10^-2);
   Zo=300;
//dmin=distance betweeen adjacent voltage minimas
ampK=round(((S-1)/(S+1))*100)/100;
ZR=fix(Zo*(1+ampK)/(1-ampK));
printf("Input impedance = %f ohms\n",ZR);
lo=2*dmin; //lo=wavelength
b=(2*%pi)/lo;
phi=(2*b*Ymin)-%pi;
theta=-fix((phi-(2*b*l)));
Zm=Zo*(round((1+(ampK*exp(%i*theta)))*100)/100)/(
   round((1-(ampK*exp(%i*theta)))*100)/100);
printf("Load impedance = %f + j(%f) ohms",round(real
   (Zm*100))/100,round(imag(Zm*100))/100);
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Chapter 7

Waveguides

Scilab code Exa 7.1 Calculating critical and guide wavelengths

```
clear;
clc;
c=3*(10^8);
f=3000*(10^8);
lo=c/f;
l=lo*(10^4);
m=1;n=0;a=7.62;
lc=2*a;
printf("—Critical wavelength = %f cm\n",lc);
lg=sqrt((l*l*lc*lc)/((lc*lc)-(l*l)));
printf("—Guide wavelength = %f cm",round(lg*10)/10);
```

Scilab code Exa 7.2 Finding the value of the dominant mode

```
clear;
clc
a=3;
lc=2*a;
```

```
Zs=500;n=377;c=3*(10^8);
lo=sqrt(1-((n/Zs)^2))*lc;
f=c/lo;
f1=f/(10^7);
printf("Frequency of dominant mode = %f GHz",round(
    f1*100)/100);
```

Scilab code Exa 7.3 Calculating the cut off wavelength and guide wavelength and group and phase velocities and the characteristic wave impedance

```
clear;
clc
a=4.5;b=3;f=9*(10^9);c=3*(10^8);n=377
lo=c/f;
l=lo*(10^2);
lc=2*a;
printf("(i)Cutoff wavelegth = %f cm\n",lc);
lg=l /(sqrt(1-((l/lc)^2)));
printf("(ii)Guide wavelength = %f cm\n",fix(lg*100)
    /100);
Vp=(lg/l)*c*10^-8;
printf("(iii)Phase velocity = %f * 10^8 m/sec\n",fix
    (Vp*100)/100);
Vg=(l/lg)*c*10^-8;
printf("    Group velocity = %f * 10^8 m/sec\n",round
    (Vg*100)/100);
Z=n/(sqrt(1-((l/lc)^2)));
printf("(iv)Characteristic impedance = %f ohm",fix(Z
    ));
```

Scilab code Exa 7.4 Calculating voltage attenuation

```
clear;
```

```
clc;
a=1;c=3*(10^8);f=(10^9);d=25;
lc=2*a;
lo=c/f;
l=lo/(10^2);
att=(54.55/lc)*d;
printf("Total attenuation = %f db",round(att*100)
   /100);
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Scilab code Exa 7.5 Calculating group and phase velocities and phase constant

```
clear;
clc;
c=3*(10^8);f=3000*(10^6);a=.0722;
lo=c/f;
lc=2*a;
lg=lo/(sqrt(1-((lo/lc)^2)));
Vp=(lg/lo)*c*10^-8;
printf("-Phase velocity Vp = %f * 10^8 m/sec\n",
   round(Vp*10)/10);
Vg=(lo/lg)*c*10^-8;
printf("-Group velocity Vg = %f * 10^8 m/sec\n",
   round(Vg*10)/10);
b=(2*%pi)/lg;
printf("-Phase constant = %f radians/m",round(b));
```

Scilab code Exa 7.6 Calculating cut off frequencies

```
clear;
clc
d=5;c=3*(10^8);
lo=1.706*d;
f=c/lo;
ff=f/(10^7);
printf("(i)Cutoff frequency for TE11 = %f GHz\n",
    round(ff*100)/100);
l=1.306*d;
fc=c/l;
ffc=fc/(10^7);
printf("(ii)Cutoff frequency for TE01 = %f GHz",
    round(ffc*10)/10);
```

Scilab code Exa 7.7 Calculating cut off wavelength and guide wavelength and characteristic wave impedance

```
clear;
clc;
c=3*(10^8);f=8*(10^9);r=2.5;h=1.84;n=377;
l=c/f;
lo=l*(10^2);
lc=2*%pi*r/h;
printf("-Cutoff wavelength = %f cm\n",round(lc*100)
    /100);
lp=lo/(sqrt(1-((lo/lc)^2)));
printf("-Guide wavelength = %f cm\n",round(lp*100)
    /100);
Zo=n/(sqrt(1-((lo/lc)^2)));
printf("-Characteristic wave impedance = %f ohm",fix
    (Zo*10)/10);
```

Chapter 8

Transmission Lines Measurements

Scilab code Exa 8.1 Calculating the value of load impedance

```
clear;
clc;
K=0.5*exp(%i*(30)/(180/%pi));
Zo=100;
Zl=Zo*(1+K)/(1-K);
A=real(Zl);
B=imag(Zl);
printf("Load impedance = %f /_ %f ohms",fix(abs(Zl))
    ,round(atan(B,A)*180/%pi));
```

Scilab code Exa 8.2 Calculating insertion loss

```
clear;
clc;
P1=67;P2=30;
L=10*log10(P1/P2);
```

```
printf("Insertion loss = %f db",fix(L*100)/100);
```

Scilab code Exa 8.3 Calculating insertion loss

```
clear;
clc;
S=2;
Lr=10*log10(((S+1)^2)/(4*S));
Ld=10*log10((S+1)/(S-1));
L=Ld+Lr;
printf("Insertion loss = %f db",round(L*100)/100);
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Chapter 9

Artificial Lines

Scilab code Exa 9.1 Designing an artificial line

```
clear;
clc;
R=10.4;L=3.67*(10^-3);G=0.8*(10^-6);C
    =0.00835*(10^-6);bmax=0.1;
f=5.5*(10^3); //artificial line will be designed fr
    highest frequecy of operation
w=2*%pi*f;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
P=sqrt(Z*Y);
b=imag(P);
l=bmax/b;
Zs=Z*l/2;
Zsh=1/(Y*l);
Zr=Y*l;
R1=real(Zs);
printf("−R1/2 = %f ohms\n",round(R1*10)/10);
L1=imag(Zs)/w;
printf("−L1/2 = %f mH\n",fix(L1*(10^3)*100)/100);
C2=imag(Zr)/w;
printf("−C2 = %f microfarads\n",fix(C2*(10^6)*10^4)
    /10^4);
G2=real(Zr);
R2=1/(round(G2*10^6*10)/10);
printf("-R2 = %f m ohms",R2);
```

Scilab code Exa 9.2 Designing an artificial line

```
clear;
clc;
R=16.64;L=5.87*(10^-3);G=1.28*(10^-6);C
    =0.0134*(10^-6);bmax=0.1;f2=5500;f1=30;
w=2*%pi*f2;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
P=fix(sqrt(Z*Y)*10^4)/10^4;
a=-f1*((-3*real(P)*real(P)*imag(P))-((imag(P))^3))
    /24;
a1=round(a*1000)/1000;
v=sqrt(bmax/a1);
l=f1/v;
R1=R*f1/(2*l);
L1=L*f1*10^3/(2*l);
C1=C*f1*10^6/l;
G1=G*f1/l;
Rg=1/G1;
printf("The elements of the artificial line are:\n")
    ;
printf(" R/2 = %f ohms\n",fix(R1*100)/100);
printf(" L/2 = %f mH\n",fix(L1*100)/100);
printf(" C = %f microfarads\n",fix(C1*1000)/1000);
printf(" Rg = %f k ohms\n",round((Rg*0.1)/100));
```

Scilab code Exa 9.3 Designing a delay line using T section

```
clear;
clc;
Ro=500;Td=1*(10^-6);Tr=0.3*(10^-6);
n=1.1*((Td/Tr)^(3/2));
N=round(n);
printf("-Number of T-sections required = %f\n",N);
C=Td/(1.07*N*Ro);
printf("-C = %f microfarads\n",C*(10^6));
L=(Ro*Td)/(1.07*n);
printf("-L = %f mH",L*(10^3));
//the difference in result of L is due to erroneous
   value in textbook.
disp("The difference in result of L is due to
   erroneous value in textbook")
```

Scilab code Exa 9.4 Designing an artificial line

```
clear;
clc;
R=10.4;L=3.66*(10^-3);G=0.8*(10^-6);C
   =0.00835*(10^-6);bmax=0.1;s=7.88;f=5000;w=2*%pi*f
   ;
Z=R+(%i*w*L);
Y=G+(%i*w*C);
P=sqrt(Z*Y);
b=imag(P);
lmax=bmax/b;
l=s/20; //l<lmax
Zs=Z*l/2;
Zsh=1/(Y*l);
Zr=Y*l;
R1=real(Zs);
printf("-R1 = %f ohms\n",round(R1*100)/100);
L1=imag(Zs)/w;
printf("-L1 = %f mH\n",round(L1*(10^3)*100)/100);
```

```
C2=imag(Zr)/w;
printf("—C2 = %f microfarads\n",C2*(10^6));
G2=real(Zr);
printf("—G2 = %f micromhos",G2*(10^6));
```

Chapter 11

Networks

Scilab code Exa 11.1 Calculating image and iterative impedance

```
clear;
clc;
Za=200;
Zb=400;
Zc=500;
Zi1=sqrt((Za+Zc)*((Za*Zc)+(Za*Zb)+(Zc*Zb))/(Zb+Zc));
printf("(a)Image impedance Zi1 = %f ohms\n",round(
   Zi1*10)/10);
Zi2=sqrt((Zb+Zc)*((Za*Zc)+(Za*Zb)+(Zc*Zb))/(Za+Zc));
printf("  Image impedance Zi2 = %f ohms\n",round(Zi2
   ));
Zt1=(1/2)*((Za-Zb)+sqrt(((Za-Zb)^2)+(4*((Za*Zb)+(Za*
   Zc)+(Zb*Zc)))));
printf("(b)Iterative impedances Zt1 = %f ohms\n",
   round(Zt1*10)/10);
Zt2=(1/2)*((Zb-Za)+sqrt(((Zb-Za)^2)+(4*((Za*Zb)+(Za*
   Zc)+(Zb*Zc)))));
printf("  Iterative impedances Zt2 = %f ohms",round(
   Zt2*10)/10);
```

Scilab code Exa 11.2 Calculating iterative impedance

```
clear;
clc;
Za=%i*200;Zc=-%i*500;
Zt1=(Za/2)+(sqrt((Za*Za/4)+(Za*Zc)));
A=real(Zt1);
B=imag(Zt1);
printf("Iterative impedances Zt1 = %d + j(%d) ohms\n
    ",A,B);
Zt2=(-Za/2)+(sqrt((Za*Za/4)+(Za*Zc)));
C=real(Zt2);
D=imag(Zt2);
printf("Iterative impedances Zt2 = %d + j(%d) ohms",
   C,D);
```

Scilab code Exa 11.3 Calculating iterative and image impedance

```
clear;
clc;
Z1=30+(%i*7.5);Z2=50+(%i*10);Z3=-%i*3229;
Za=Z1;Zb=Z2;Zc=Z3;
a=Za+Zc;
b=Zb+Zc;
s=(Za*Zb)+(Zb*Zc)+(Zc*Za);
Zi1=sqrt(a*s/b);
printf("Image impedances Zi1 = %f /_ %f ohms\n",
   round(abs(Zi1)*10)/10,round(atan(imag(Zi1),real(
   Zi1))*180*100/%pi)/100);
Zi2=sqrt(b*s/a);
printf(" Image impedances Zi1 = %f /_ %f ohms\n",
   round(abs(Zi2)*10)/10,round(atan(imag(Zi2),real(
   Zi2))*180*100/%pi)/100);
Zt1=(1/2)*((Za-Zb)+sqrt(((Za-Zb)^2)+(4*s)));
printf(" Iterative impedances Zt1 = %f + j(%f) ohms\
   n",round(real(Zt1)*100)/100,round(imag(Zt1)*100)
   /100);
Zt2=(1/2)*((Zb-Za)+sqrt(((Za-Zb)^2)+(4*s)));
printf(" Iterative impedances Zt1 = %f + j(%f) ohms\
   n",round(real(Zt2)*100)/100,round(imag(Zt2)*100)
   /100);
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Scilab code Exa 11.4 Calculating image and iterative impedances and transfer constants

```
clear;
clc;
Za=%i*300;Zc=-%i*700;
Zoc1=Za+Zc;
Zsc1=Za;
Zoc2=Zc;
Zsc2=(Za*Zc)/(Za+Zc);
Zi1=sqrt(Zoc1*Zsc1);
printf("-Image impedance Zi1 = %f ohms\n",round(Zi1)
   );
Zi2=sqrt(Zoc2*Zsc2);
printf("-Image impedance Zi2(in ohms)= %f ohms\n",
   round(Zi2));
Zt1=(Za/2)+(sqrt((Za*Za/4)+(Za*Zc)));
A=real(Zt1)
B=imag(Zt1);
printf("-Iterative impedance Zt1 = %f + j(%f) ohms\n
   ",round(A),B);
```

```
Zt2=(-Za/2)+(sqrt((Za*Za/4)+(Za*Zc)));
C=real(Zt2);
D=imag(Zt2);
printf("-Iterative impedance Zt2 = %f + j(%f) ohms\n
   ",round(C),D);
I=(1+(sqrt(Zsc1/Zoc1)))/(1-(sqrt(Zsc1/Zoc1)));
Qi=(log(I))/2;
E=real(Qi);
F=imag(Qi);
printf("-Image transfer constant = %f + j(%f)\n",E,
   round(F*1000)/1000);
I1=(Zt1+Zc);
I2=Zc;
Q2=log(I1/I2);
G=real(Q2);
H=imag(Q2);
printf("-Iterative transfer constant = %f +j(%f)",G,
   round(H*180*10/%pi)/10);
```

Scilab code Exa 11.5 Computing the insertion ratio and loss

```
clear;
clc;
Zg=100;Zl=500;b=63.4;a=0;
theta=a+(%i*b);
Fr=2*(sqrt(Zg*Zl))/(Zg+Zl);
IR=Fr*exp(theta);
A=real(IR);
B=imag(IR);
printf("-Insertion ratio = %f /_ %f\n",fix(abs(IR)
   *100)/100,theta/%i);
IL=-20*log10(Fr);
printf("-Insertion loss = %f db",round(IL*10)/10);
```

Scilab code Exa 11.6 Determining the components of a symmetrical T section network

```
clear;
clc;
Zoc=800;Zsc=600;
R2=sqrt((Zoc*Zoc)-(Zsc*Zoc));
R1=2*(Zoc-R2);
printf("The components of the network are:\n");
printf("   R1/2 = %d ohms\n",R1/2);
printf("   R2 = %d ohms",R2);
```

Scilab code Exa 11.7 Calculating current

```
clear;
clc;
Z1=20;Z2=10;Vrms=10;
Zot=sqrt(((Z1*Z1)/4)+(Z1*Z2));
I=Vrms/Zot;
a=[30 27.32;1 -3.732];
b=[0.577;0];
b=inv(a)*b;
printf("Current flowing through the terminating
   impedance = %f mA",round(b(2,1)*(10^4)*100)/100);
```

Scilab code Exa 11.8 Calculating characteristic impedance

```
clear;
clc;
```

```
Z1=50+(%i*125);Z2=200-(%i*100);
Zot=sqrt((Z1/4)*(Z1+(4*Z2)));
A=real(Zot);
B=imag(Zot);
printf("Characteristic impedance = %f + j(%f) ohms",
   round(A),round(B*100)/100);
```

Scilab code Exa 11.9 Designing a L section

```
clear;
clc;
Zl=20+(-%i*5);w=5*(10^6);Rg=600;
Rl=real(Zl);
Xc=-%i*imag(Zl);
A=imag(Xc);
printf("Compensating reactance = j(%d) ohms\n ",A);
X21=-sqrt(Rl*(Rg-Rl));
X22=-X21;
X31=-Rg*sqrt(Rl/(Rg-Rl));
X32=-X31;
X2a=X22+(Xc/%i);
L2=X2a/w;
C3=-1/(w*X31);
printf("(a)In the first case X2 is inductive, X2=L2
   = %f micro-henry\n",round(L2*(10^6)*100)/100);
printf("\t\t\t\t\tX3=C3 = %f pf\n",round(C3*(10^12))
   );
X2b=X21+(Xc/%i);
C2=-1/(w*X2b);
L3=X32/w;
printf("(b)In the second case X2 is capacitive,X2=C2
    = %f pf\n",round(C2*(10^12)));
printf("\t\t\t\t\tX3=L3 = %f micro-henry",round(L3
   *(10^6)*10)/10);
```

Scilab code Exa 11.10 Designing a reactive T network

```
clear;
clc; //solved using the value of w used in the
   solution
R1=1000;Xg=250+(%i*200);w=2*(10^6);
Rg=real(Xg);
X1=sqrt(Rg*R1);
X2=X1;
X3=X1;
C3=1/(w*X3);
printf("-C3 = %d pf\n",C3*(10^12));
L1=X1/w;
printf("-L1 = %d micro-henry\n",L1*(10^6));
L2=X2/w;
Xc=-%i*imag(Xg);
X21=X2+(Xc/%i);
L21=X21/w;
printf("-L2 = %d micro-henry",L21*(10^6));
```

Scilab code Exa 11.11 Determining the equivalent T network

```
clear;
clc;
Zin=400+(%i*4000);Zout1=100+(%i*1000);Zout2=38+(%i
   *380);
Zoc1=Zin;Zoc2=Zout1;Zsc2=Zout2;
Z3=sqrt(Zoc1*(Zoc2-Zsc2));
Z1=Zoc1-Z3;
Z2=Zoc2-Z3;
A=real(Z1);
B=imag(Z1);
```

```
C=real(Z2);
D=imag(Z2);
E=real(Z3);
F=imag(Z3);
printf("−Z1 = %f + j(%f) ohms\n",round(A*10)/10,
    round(B*10)/10);
printf("−Z2 = %f + j(%f) ohms\n",fix(C*10)/10,fix(D)
    );
printf("−Z3 = %f + j(%f) ohms\n",fix(E*10)/10,fix(F)
    );
```

Scilab code Exa 11.12 Finding the elements of the arms of a loss less T network

```
clear;
clc;
w=5*(10^6);Rg=800;Rl=200;b=-12;
X3=-(sqrt(Rg*Rl))/sin(b*%pi/180);
L3=X3/w;
X1=(-Rg/tan(b*%pi/180))+((sqrt(Rg*Rl)/sin(b*%pi/180)
    ));
L1=X1/w;
X2=(-Rl/tan(b*%pi/180))+((sqrt(Rg*Rl)/sin(b*%pi/180)
    ));
C2=-1/(X2*w);
printf("−L3 = %f micro−henry\n",fix(L3*(10^6)));
printf("−L1 = %f micro−henry\n",fix(L1*(10^6)));
printf("−C2 = %f pf",round(C2*(10^12)));
```

Scilab code Exa 11.13 Determining the image and iterative impedance and equivalent T network

```
clear;
```

```
clc;
Ya=40*(10^-3);Yb=50*(10^-3);Yc=20*(10^-3);
Za=1/Ya;Zb=1/Yb;Zc=1/Yc;
Z1=Zb*Zc/(Za+Zb+Zc);
Z2=Za*Zc/(Za+Zb+Zc);
Z3=Zb*Za/(Za+Zb+Zc);
Zi1=sqrt(((Z3+Z1)/(Z3+Z2))*((Z1*Z2)+(Z2*Z3)+(Z1*Z3))
    );
printf("(a)Zi1 = %f ohms\n",round(Zi1));
Zi2=sqrt(((Z3+Z2)/(Z3+Z1))*((Z1*Z2)+(Z2*Z3)+(Z1*Z3))
    );
printf("    Zi2 = %f ohms\n",round(Zi2*100)/100);
Zt1=(1/2)*((Z1-Z2)+(sqrt(((Z1-Z2)^2)+(4*((Z1*Z2)+(Z2
    *Z3)+(Z1*Z3))))));
printf("    Zt1 = %f ohms\n",fix(Zt1*100)/100);
Zt2=(1/2)*((Z2-Z1)+(sqrt(((Z1-Z2)^2)+(4*((Z1*Z2)+(Z2
    *Z3)+(Z1*Z3))))));
printf("    Zt2 = %f ohms\n\n",fix(Zt2*100)/100);
Zb1=Za*Zb/(Za+Zb);
Z11=Zb*Zc/(Zb+Zc+Zb1);
Z21=Zb1*Zc/(Zb+Zc+Zb1);
Z31=Zb1*Zb/(Zb+Zc+Zb1);
Zr=Zc+Z21;
Zs=Z21+Zb;
Z12=Z31*Zs/(Z31+Zr+Za);
Z22=Zr*Za/(Z31+Zr+Za);
Z32=Z31*Za/(Z31+Zr+Za);
Z121=Z12+Z11;
printf("   The desired T network will be as:\n");
printf("   Z1 = %f ohms\n",round(Z121*100)/100);
printf("   Z2 = %f ohms\n",fix(Z22*10)/10);
printf("   Z3 = %f ohms\n",round(Z32*10^4)/10^4);
```

Scilab code Exa 11.14 Calculating the characteristic impedance and propagation function of given network

```
clear;
clc;
L=20*(10^-3);C=0.064*(10^-6);f=400;
w=2*%pi*f;
Z1=round(2*%i*w*L*10)/10;
Z2=1/(%i*w*C);
Zo=sqrt((Z1*Z1/4)+(Z1*Z2));
printf("Characteristic impedance = %f ohms\n",round(
   Zo*100)/100);
Pf=(1+(Z1/(2*Z2)));
printf(" Propagation constant = %f",round(Pf*100)
   /100);
//the difference in result  is due to erroneous
   value in textbook.
disp("The difference in result is due to erroneous
   value in textbook")
```

Scilab code Exa 11.15 Calculating characteristic impedance and propagation function

```
clear;
clc;
Za=300;Zc=600;R=1000;
Zi1=sqrt((Za*Za)+(Za*Zc));
Zi2=Za*Zc/sqrt((Za*Za)+(Za*Zc));
Zt1=(Za/2)+sqrt((Za*Za/4)+(Za*Zc));
Zt2=(-Za/2)+sqrt((Za*Za/4)+(Za*Zc));
printf("The image impedances are:\n ");
printf("Zi1 = %f ohms\n",round(Zi1*10)/10);
printf(" Zi2 = %f ohms\n",round(Zi2*10)/10);
printf(" Zt1 = %f ohms\n",Zt1);
printf(" Zt2 = %f ohms\n\n",Zt2);
I=(((R+Zt1+Zt2)*(R+Zt1)/(Zt1))-Zt1)*(1/(R+R));
Ir=20*log10(round((I*1000))/1000);
printf(" Insertion loss = %f db",Ir);
```

Scilab code Exa 11.16 Designing a L matching loss less network

```
clear;
clc;
Rg=100;Rl=50;f=5*(10^6);
w=2*%pi*f;
X21=sqrt(Rl*(Rg-Rl));
X22=-X21;
X31=-Rg*sqrt(Rl/(Rg-Rl));
X32=-X31;
L2=X21/w;
printf("(i)X2 is inductive and X3 is capacitive
    where\n  X2=L2 = %f microhenry\n",round(L2*(10^6)
    *1000)/1000);
C2=-1/(w*X31);
printf("  X3=C3 = %f pf\n",round(C2*(10^12)*10)/10);
L31=X32/w;
printf("(ii)X3 is inductive and X2 is capacitive
    where\n  X3=L3 = %f microhenry\n",round(L31
    *(10^6)*1000)/1000);
C21=-1/(w*X22);
printf("  X2=C2 = %f pf",round(C21*(10^12)));
```

Scilab code Exa 11.17 Calculating different losses and total insertion loss

```
clear;
clc;
Zl=100;Zsh=500;Zg=300;
Zoc=Zl+Zsh;
Zsc=Zl+(1/((1/Zl)+(1/Zsh)));
Zi1=sqrt(Zoc*Zsc);
```

```
Zi2=Zi1;
theta=atanh(sqrt(Zsc/Zoc));
att=theta*8.686;
printf("-Attenuation loss = %f db\n",round(att*10)
   /10);
inp=20*log10(round((Zi1+Zg))/(2*sqrt(Zi1*Zg)));
printf("-Loss due to mismatch at the input = %f db\n
   ",round(inp*100)/100);
out=20*log10((Zi2+Zl)/(2*(sqrt(Zi2*Zl))));
printf("-Loss due to mismatch at output = %f db\n",
   round(out*100)/100);
Ki1=(Zi1-Zg)/(Zi1+Zg);
Ki2=(Zi2-Zl)/(Zi2+Zl);
inte=-20*log10(1-(Ki1*Ki2*exp(-2*theta)));
printf("-Loss due to interaction = %f db\n",fix(inte
   *100)/100);
ext=20*log10((Zg+Zl)/(2*sqrt(Zg*Zl)));
printf("-External reflection loss = %f db\n",round(
   ext*100)/100);
tot=att+inp+out-inte-ext;
printf("-Total insertion loss = %f db\n",fix(tot
   *100)/100);
```

Scilab code Exa 11.18 Calculating the elements of a L section network

```
clear;
clc;
Rg=8000;Zl=500+(%i*500);f=5*(10^6);
//value of f as taken in solution
w=2*%pi*f;
Xc=-%i*imag(Zl);
Rl=real(Zl);
X21=sqrt(Rl*(Rg-Rl));
X22=-X21;
X31=-Rg*sqrt(Rg/(Rg-Rl));
```

```
X32=-X31;
X2a=X21+(Xc/%i);
L2a=X2a/w;
C3a=-1/(w*X31);
printf("(a)X2 is inductive and X3 is capacitive
   where\n    X2=L2 = %f mH\n",round(L2a*(10^3)*1000)
   /1000);
printf("    X3=C3 = %f pf\n",round(C3a*(10^12)*1000)
   /1000);
X2b=X22+(Xc/%i);
C2b=-1/(w*X2b);
L3b=X32/w;
printf("(b)X2 is capacitive and X3 is inductive
   where\n    X2=C2 = %f pf\n",round(C2b*(10^12)*100)
   /100);
printf("    X3=L3 = %f mH",round(L3b*(10^3)*1000)
   /1000);
```

Scilab code Exa 11.19 Calculating image impedance and transfer constant

```
clear;
clc;
Zoc1=-%i*400;Zoc2=-%i*600;Zsc1=%i*267;Zsc2=%i*400;
A=Zoc1/Zsc1;
B=Zoc2/Zsc2;
printf("(a)Since Zoc1/Zsc1 = Zoc2/Zsc2 = %f the
   results are consistant\n",round(A*10)/10);
Zi1=sqrt(Zoc1*Zsc1);
Zi2=sqrt(Zoc2*Zsc2);
printf("The image impedances are:\n Zi1 = %f ohms\n
   Zi2 = %f ohms\n",round(Zi1*10)/10,round(Zi2*10)
   /10);
C=(1+sqrt(B))/(1-sqrt(B));
phi=round(atan(imag(C),real(C))*180*10/%pi)/10;
theta=round(%i*phi*%pi*1000/(2*180))/1000;
```

```
printf("Image transfer constant = j %f",theta/%i);
```

Scilab code Exa 11.20 Calculating reduction in power

```
clear;
clc;
Zg=300;Zi=400;Zi1=600;theta=10;
thetai=theta/8.686;
Zi2=Zi1;
Ir=thetai + log(((Zi1+Zg)/(2*sqrt(Zi1*Zg)))*((Zi1+Zi
    )/(2*sqrt(Zi1*Zi)))*((Zi+Zg)/(2*sqrt(Zi*Zg)))
    *((1-(((Zg-Zi1)/(Zg+Zi1))*((Zi-Zi1)/(Zi+Zi1))*exp
    (-2*thetai)))));
Ir1=(round(Ir*10)/10)*8.686;
printf("The reduction in power will be = %f db",
    round(Ir1*100)/100);
```

Scilab code Exa 11.21 Calculating characteristic impedance and attenuation constant and phase shift constant

```
clear;
clc;
R1=200;L1=100*(10^-3);R2=200;L2=100*(10^-3);C2
    =2.5*(10^-6);w=2000;
Z1=R1+(%i*w*L1);
Z2=1/(%i*w*C2);
Zoc=Z1+Z2;
Zsc=Z1+(1/((1/Z1)+(1/Z2)));
Zo=sqrt(Zoc*Zsc);
printf("-Characteristic impedance = %f ohms\n",round
    (Zo));
P=atanh(sqrt(Zsc/Zoc));
a=real(P);
```

```
printf("-Attenuation constant = %f nepers\n",round(a
    *100)/100);
b=(imag(P))*180/%pi;
printf("-Phase shift constant = %d degrees",b);
```

Scilab code Exa 11.22 Calculating image impedance and propagation constant and elements of T network

```
clear;
clc;
Z1o=1260*(exp(%i*30/(180/(%pi))));Z2o=2430*(exp(-%i
    *34/(180/(%pi))));
Z1s=318*(exp(%i*72/(180/(%pi))));Z2s=613*(exp(%i
    *8/(180/(%pi))));
Zi1=sqrt(Z1o*Z1s);
A=real(Zi1);
B=imag(Zi1);
printf("(i)Image impedance Zi1 = %f /_ %f ohms\n",
    round(abs(Zi1)),atan(B,A)*180/%pi);
Zi2=sqrt(Z2o*Z2s);
C=real(Zi2);
D=imag(Zi2);
printf("   Image impedance Zi2 = %f /_ %f ohms\n",
    round(abs(Zi2)),atan(D,C)*180/%pi);
Z3=sqrt(Z2o*(round(Z1o)-round(Z1s)));
Z1=Z1o-Z3;
Z2=Z2o-Z3;

P=atanh(sqrt(Z1s/Z1o));
printf(" (ii)Propagation constant = %f /_ %f\n",
    round(abs(P*100))/100,round(atan(imag(P),real(P))
    *100)/100);
printf(" (iii)The elements of the T network are:\n")
    ;
printf("      Z1 = %f + j(%f) ohms\n",round(real(Z1))
    ,round(imag(Z1)));
printf("      Z2 = %f + j(%f) ohms\n",round(real(Z2))
    ,round(imag(Z2)));
printf("      Z3 = %f + j(%f) ohms",round(real(Z3)),
    round(imag(Z3)));
```

Chapter 12

Network Theorems

Scilab code Exa 12.1 Finding the rms value of current

```
clear;
clc;
w=2*%pi*(10^6);C=100*(10^-12);V=10;L=100*(10^-6);
Zc=1/(w*C);
Ic=(V/sqrt(2))/Zc;
printf("-R.m.s. value of current flowing through
   capacitor C = %f mA\n",round(Ic*(10^3)*100)/100);
Zl=w*L;
Il=(V/sqrt(2))/Zl;
printf("-R.m.s. value of current flowing through
   inductor L = %f mA",fix(Il*(10^3)*10)/10);
```

Scilab code Exa 12.2 Calculating voltage

```
clear;
clc;
Z1=10;Z2=5;Z3=2;I1=2;I2=4;
Vab1=I1*(1/((1/Z2)+(1/(Z1+Z3))));
```

```
I5=I2*Z3/(Z1+Z2+Z3); //I5=current through the 5 ohm
    resistor
Vab2=Z2*I5;
Vab=Vab1+Vab2;
printf("Vab = %f volts",round(Vab*100)/100);
```

Scilab code Exa 12.3 Calculating current

```
clear;
clc;
R1=5;R2=%i*5;R3=3+(%i*4);V1=50*exp(%i*90/(180/%pi));
    V2=50;
Z1=R1+(1/((1/R2)+(1/R3)));
I1=V1/Z1;
Iz1=I1*R2/(R2+R3);
Z2=R2+(1/((1/R1)+(1/R3)));
I2=V2/Z2;
Iz2=-I2*R1/(R1+R3);
Iz=Iz1+Iz2;
printf("Total current in the (3+j4)ohm branch = %f /
    _ %f Amp",round(abs(Iz)*10)/10,round(atan(imag(Iz
    ),real(Iz))*180*10/%pi)/10);
```

Scilab code Exa 12.5 Determining Thevenins equivalent generator

```
clear;
clc;
Isc=10;Voc=120;Rl=8;Il=6;
Zeq=Voc/Isc; //Zeq=impedance of the equivalent
    thevenin's generator Z=R+jX
t=Voc/Isc; //(R^2)+(X^2)=(t^2)
R=(((Voc/Il)^2)-(t^2)-(Rl^2))/(2*Rl);
X=sqrt((t^2)-(R^2));
```

```
printf("The equivalent Thevenin generator has an emf
     of %d volts and an internal impedance of (%d +
   j%d) ohms",Voc,R,X);
```

Scilab code Exa 12.6 Determining Thevenins equivalent circuit

```
clear;
clc;
V=10;R1=5;R2=%i*5;R3=3+(%i*4);
Zab=R2+(1/((1/R1)+(1/R3)));
A=real(Zab);
B=imag(Zab);
I=V/(R1+R3);
Voc=I*R3;
C=real(Voc);
D=imag(Voc);
printf("The equivalent Thevenin circuit has an emf
   of %f /_ %f volts and an internal impedance of (
   %f + j%f) ohms",fix(abs(Voc)*100)/100,fix(atan(D,
   C)*180/%pi),A,B);
```

Scilab code Exa 12.7 Deriving the equivalent voltage generator circuit

```
clear;
clc;
Voc=150;Rl=10;Il=2.65;Isc=3;
t=Voc/Isc; //(R^2)+(X^2)=(t^2)
R=(((Voc/Il)^2)-(t^2)-(Rl^2))/(2*Rl);
X=sqrt((t^2)-(R^2));
printf("−The equivalent voltage generator circuit
   has an emf of %d volts and an internal impedance
   of (%f + j%f) ohms\n",Voc,fix(R*100)/100,round(X
   *100)/100);
```

```
if X>0 then
    printf("-Since X has a positive value the
       circuit will be inductive");
else
    disp("-Since X has a negative value the circuit
       will be capacitve");
end
```

Scilab code Exa 12.8 Determining the Nortons equivalent circuit

```
clear;
clc;
Za=-%i*5;Zb=5+(%i*5);V=50;Z1=5-(%i*5);Z2=10;
Isc=V/Za;
A=real(Isc);
B=imag(Isc);
printf("-Short circuit current = %d /_ %d Amps\n",
   abs(Isc),atan(B,A)*180/%pi);
Zab=1/((1/Za)+(1/Zb));
C=real(Zab);
D=imag(Zab);
printf("-Equivalent impedance = %d + j(%d) ohms\n",C
   ,D);
I1=Isc*(Zab/(Zab+Z1));
E=real(I1);
F=imag(I1);
printf("-Current through impedance Z1(=5-j5) = %d /_
    %d Amps\n",abs(I1),atan(F,E)*180/%pi);
I2=Isc*(Zab/(Zab+Z2));
G=real(I2);
H=imag(I2);
printf("-Current through impedance Z2(=10) = %f /_
   %f Amps",round(abs(I2)*100)/100,round(atan(H,G)
   *100*180/%pi)/100);
```

Scilab code Exa 12.9 Obtaining Nortons equivalent circuit

```
clear;
clc;
V1=10;V2=20;R1=5;R2=15;
Isc=(V1/R1)+(V2/R2);
Zab=1/((1/R2)+(1/R1));
printf("-Short circuit current = %f Amp\n",round(Isc
   *100)/100);
printf("-Equivalent impedance = %f ohm",Zab);
```

Scilab code Exa 12.10 Verifying reciprocity theorem

```
clear;
clc;
I=5*exp(%i*(90)/(180/%pi));Z1=5+(%i*5);Z2=2;Z3=-%i
   *2;
I2=I*Z1/(Z1+Z2+Z3);
V2=I2*Z3;
I1=I*Z3/(Z1+Z2+Z3);
V1=I1*Z1;
printf("The reciprocity theorem has been verified.")
   ;
```

Scilab code Exa 12.11 Calculating value of compensation source

```
clear;
clc;
Z1=%i*10;Z2=3+(%i*4);Z=5;V=20;
```

```
Zeq=1/((1/Z1)+(1/Z2));
Zi=Z+Zeq;
I=V/Zi;
Vc=I*Zeq;
A=real(Vc);
B=imag(Vc);
printf("Compensation source Vc = %f /_ %f volts",
   round(abs(Vc)*10)/10,round(atan(B,A)*180*10/%pi)
   /10);
```

Scilab code Exa 12.12 Calculating power in load and turns ratio for maximum power transfer

```
clear;
clc;
Zg=10000;Zl=24+(%i*7);Vrms=100;
Rl=real(Zl);
Xl=imag(Zl);
Zeq=sqrt(((Rl+Zg)^2)+((Xl)^2));
Irms=Vrms/Zeq;
P=Irms*Irms*Rl;
printf("(i)Power in the load if connected directly
   to the generator = %f mW\n",round(P*(10^3)*10)
   /10);
ampZl=sqrt((Rl*Rl)+(Xl*Xl));
n=1/sqrt(ampZl/Zg);
printf("(ii)The desired turn ratio of transformer is
    1:%d\n",n);
Pl=Vrms*Vrms/(4*Zg);
printf("(iii)Power transferred under ideal load
   conditions =%f mW",Pl*(10^3));
```

Scilab code Exa 12.13 Determining value of maximum power

```
clear;
clc;
Zg=10+(%i*20);V=50;
Rg=real(Zg);Xg=imag(Zg);
ampZg=sqrt((Rg*Rg)+(Xg*Xg));
Rl=ampZg;
I=V/(Zg+Rl);
ampI=round(sqrt((real(I))^2+(imag(I))^2)*100)/100;
P=ampI*ampI*round(Rl*10)/10;
printf("Power delivered to the load = %f Watt",round
    (P*10)/10);
```

Scilab code Exa 12.14 Calculating value of R and X which result in maximum power transfer and value of maximum power tranferrred

```
clear;
clc;
V=50*exp(%i*45/(180/%pi));Z1=3;Z2=2+(%i*10);Xl=2;
Voc=V*Z2/(Z1+Z2);
Zab=1/((1/Z1)+(1/Z2));
Zg=Zab
Rl1=Zg-(%i*Xl);
Rl=abs(Rl1);
Z=Zab+Rl-(%i*Xl);
I1=Voc/Z;
I=abs(I1);
P=I*I*Rl;
printf("-Rl = %f ohms\n",round(Rl*100)/100);
printf("-Xl = %f ohms\n",Xl);
printf("-Maximum power delivered to load = %f Watts"
    ,round(P));
```

Scilab code Exa 12.17 Calculating effective resistance of coil and inductance of the effective resistance

```
clear;
clc;
f=5*(10^6);C=400*(10^-12);R=10*(10^3);
w=2*%pi*f;
L=2/(w*w*C);
r=1/(w*w*C*C*R);
printf("-Effective resistance of the coil = %f ohms\
   n",round(r*100)/100);
printf("-Inductance of effective resistance of the
   coil = %f mH",round(L*(10^3)*1000)/1000);
```

Scilab code Exa 12.19 Determining the elements of a series type Foster network

```
clear;
clc;
w1=2*(10^6);w2=3*(10^6);w3=4*(10^6);w=1*(10^6);Z=%i
   *100;
F=((w*w)-(w2*w2))/(((w*w)-(w1*w1))*((w*w)-(w3*w3)));
H=Z/(%i*w*F);H1=round(H*10^-8*100)/(100*10^-8);
A=((w1*w1)-(w2*w2))/((w1*w1)-(w3*w3));
B=((w3*w3)-(w2*w2))/((w3*w3)-(w1*w1));
C2=-1/(H1*A);
printf("Elements of the series type Foster network
   are:\n");
printf("-C2 = %f pf\n",fix(C2*(10^12)))
L2=1/(w1*w1*C2);
printf("-L2 = %f microhenry\n",round(L2*(10^6)*10)
   /10);
C4=-1/(H1*B);
printf("-C4 = %f pf\n",fix(C4*(10^12)));
L4=1/(w3*w3*C4);
```

```
printf("—L4 = %f microhenry",round(L4*(10^6)*10)/10)
    ;
```

Scilab code Exa 12.20 Determining the elements of series type Foster network

```
clear;
clc;
w1=1000;w2=1500;w3=2000;w4=3000;w5=5000;w=100;Z=%i
    *100;
F=((w*w)-(w2*w2))*((w*w)-(w4*w4))/(((w*w)-(w1*w1))
    *((w*w)-(w3*w3))*((w*w)-(w5*w5)));
H=Z/(%i*w*F);
A=((w1*w1)-(w2*w2))*((w1*w1)-(w4*w4))/(((w1*w1)-(w3*
    w3))*((w1*w1)-(w5*w5)));
B=((w3*w3)-(w2*w2))*((w3*w3)-(w4*w4))/(((w3*w3)-(w5*
    w5))*((w3*w3)-(w1*w1)));
C=((w5*w5)-(w2*w2))*((w5*w5)-(w4*w4))/(((w5*w5)-(w1*
    w1))*((w5*w5)-(w3*w3)));
C2=-1/(H*A);
printf("—C2 = %f microfarads\n",round(C2*(10^6)*100)
    /100)
L2=1/(w1*w1*C2);
printf("—L2 = %f henry\n",round(L2*100)/100);
C4=-1/(H*B);
printf("—C4 = %f microfarads\n",round(C4*(10^6)*100)
    /100);
L4=1/(w3*w3*C4);
printf("—L4 = %f henry\n",round(L4*100)/100);
C6=-1/(H*C);
L6=1/(w5*w5*C6);
printf("—C6 = %f microfarads\n",round(L6*100)/100);
printf("—L6 = %f henry\n",round(C6*(10^6)*100)/100)
```

Scilab code Exa 12.21 Finding current

```
clear;
clc;
V=10;Rl=10;Z1=5;Z2=5;
Zab=1/((1/Z1)+(1/Z2));
I1=V/(Z1+Z2);
Voc=I1*Z1;
I=Voc/(Zab+Rl);
printf("Current in the 10 ohm resistor = %d mA",I
    *(10^3));
```

Scilab code Exa 12.22 Using Thevenins theorem to calculate power

```
clear;
clc;
Rl=10;V=100;Z1=-%i*30;Z2=-%i*30;Z3=20+(%i*10);
I=V/(Z1+Z2);
Voc=I*Z2;
Zab=1/((1/Z1)+(1/Z2))+Z3;
Z=Zab+Rl;
Il=Voc/Z;
ampIl=sqrt(real(Il)^2+imag(Il)^2);
Pl=ampIl*ampIl*Rl;
printf("Power in the load = %f Watts",round(Pl*100)
    /100);
```

Scilab code Exa 12.23 Finding Thevenins equivalent generator

```
clear;
clc;
V=100;R1=10;R2=10;R3=10;R4=10;
I=V/(R1+R2);
Vab=I*R2;
Zab=1/((1/R1)+(1/R2));
V1=Vab*R4/(Zab+R3+R4);
Z1=1/((1/(Zab+R3))+(1/R4));
printf("The equivalent Thevenin circuit has an emf
    of %d volts and an internal impedance of %f ohms"
    ,V1,Z1);
```

Scilab code Exa 12.25 Determining the value of load impedance for which maximum power is consumed in load and the corresponding power

```
clear;
clc;
V=100;Z1=20;Z2=-%i*100;Z3=100;
Zab=1/((1/(Z2))+(1/Z3));
Voc=V*Z3/(Z2+Z3);
I=Voc/(Zab+Voc);
P=(abs(I))^2 * real(Zab);
Z=conj(Zab);
A=real(Z);
B=imag(Z);
printf("Value of load value for maximum power = %d +
     j(%d) ohms\n",A,B);
printf("  Maximum Power = %d Watts",P);
```

Scilab code Exa 12.26 Determining Thevenins equivalent circuit

```
clear;
clc;
```

```
V=100;R1=50;R2=60;R3=40;R4=60;R5=40;
Rac=1/((1/(R2+R3))+(1/(R4+R5)));
Rt=Rac+R1;
I=V/Rt;
V1=I*R1;
Vac=V-V1;
Vab=R2*Vac/(R2+R3);
Vad=R5*Vac/(R4+R5);
Vbd=Vab-Vad;
Voc=Vbd;
R1y=R5*R1/(R1+R4+R5);
R2y=R1*R4/(R1+R4+R5);
R3y=R4*R5/(R1+R4+R5);
Rbd=R3y+(1/((1/(R1y+R2))+(1/(R2y+R3))));
printf("The equivalent Thevenin circuit has an emf
   of %d volts and an internal impedance of %d ohms"
   ,Vbd,Rbd);
```

Scilab code Exa 12.27 Calculating current

```
clear;
clc;
V1=6;V2=6;R1=2;R2=1;R3=2;
Y1=1/R1;Y2=1/R2;
Vm=((V1*Y1)+(V2*Y2))/(Y1+Y2);
Zm=1/(Y1+Y2);
I=Vm/(Zm+R3);
printf("Current in resistor R3 = %f Amp",I);
```

Scilab code Exa 12.28 Using Millmans theorem to find current and voltage

```
clear;
```

```
clc;
V1=5;V2=4;R1=5;R2=4;I3=1;R3=10;R4=5;
V3=I3*R3;
Y1=1/R1;Y2=1/R2;Y3=1/R3;
V4=((V1*Y1)+(V2*Y2)+(V3*Y3))/(Y1+Y2+Y3);
I4=V4/R4;
printf("Voltage across resistor R4 = %f volts\n",V4)
    ;
printf("Current in resistor R4 = %f Amp",I4);
//the difference in result  is due to erroneous
    value in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Scilab code Exa 12.29 Verifying Tellegens theorem

```
clear;
clc;
V=120;R1=6;R2=8;R3=3;R4=5;
a=[7 -4;-1 2];
b=[60;0];
b=inv(a)*b;
I1=b(1,1);
I2=b(2,1);
VI1=(R1*I1)+(R2*(I1-I2))-V;
printf("Summation V*I for first loop = %d\n",VI1);
VI2=(R3*I2)+(R4*I2)-(R2*(I1-I2));
printf("Summation V*I for second loop = %d\n",VI2);
printf("Hence Tellegens theorem has been verified.")
    ;
```

Scilab code Exa 12.30 Verifying Tellegens theorem

```
clear;
clc;
V1=4;V2=2;V3=2;V4=3;V5=-1;V6=-5;i1=2;i2=2;i3=4;i4
   =-2;i5=-6;i6=4;
Va=V2+V3-V1;
printf("V2+V3-V1= %d volts\n",Va);
Vb=V2+V4+V6;
printf("V2+V4+V6= %d volts\n",Vb);
Vc=V4+V5-V3;
printf("V4+V5-V6= %d volts\n",Vc);
printf("-Hence KVL is satisfied for all three loops
   PQTU,PRYX and QRST respectively.\n\n");
Ia=i1+i2-i6;
printf("i1+i2-i6= %d Amp\n",Ia);
Ib=i3+i4-i2;
printf("i3+i4-i2= %d Amp\n",Ib);
Ic=i5+i6-i4;
printf("i5+i6-i2= %d Amp\n",Ic);
printf("-Hence KCL is satisfied at all three nodes P
   ,Q and R  respectively.\n\n");
VI=(V1*i1)+(V2*i2)+(V3*i3)+(V4*i4)+(V5*i5)+(V6*i6);
printf("-Summation V*I= %d\n\n",VI);
printf("-This proves Tellegens theorem.");
```

Scilab code Exa 12.31 Finding current

```
clear;
clc;
Vrms=10;R1=5;Z1=20;Z2=20;Z3=10;Z4=10;
Zab=(1/((1/Z1)+(1/Z2)))+(1/((1/Z3)+(1/Z4)));
Voc=R1;
I=Voc/(Zab+R1);
printf("Current I = %f Amps",I);
```

Scilab code Exa 12.32 Calculating current and voltage using Thevenins theorem

```
clear;
clc;
I1=6;I2=8;R1=4;R2=4;R3=6;R4=4;Rl=3;
V1=I1*R1;V2=I2*R4;
Voc1=V1*(R3+R4)/(R1+R2+R3+R4);
Voc2=V2*(R1+R2)/(R1+R2+R3+R4);
Voc=Voc1+Voc2;
Zab=1/((1/(R1+R2))+(1/(R3+R4)));
Il=Voc/(Zab+Rl);
printf("-il = %f Amp\n",round(Il*10)/10);
el=Il*Rl;
printf("-el = %f volts",round(el*10)/10);
```

Chapter 13

Network Parameters

Scilab code Exa 13.1 Determining z and y parameters

```
clear;
clc;
Z1=%i*20;Z2=%i*25;Z3=30;
Z11=Z1+Z3; //Z11=V1/I1 when I2=0
Z12=Z3; //Z12=V1/I2 when I1=0
Z21=Z3; //Z21=V2/I1 when I2=0
Z22=Z2+Z3; //Z22=V2/I2 when I1=0
printf("(a)The z-parameters are\n");
printf("  Z11 = %f + j(%f) ohms\n",real(Z11),imag(
   Z11));
printf("  Z12 = %f ohms\n",Z12);
printf("  Z21 = %f ohms\n",Z21);
printf("  Z22 = %f + j(%f) ohms\n",real(Z22),imag(
   Z22));
deltaz=(Z11*Z22)-(Z12*Z21);
y11=Z22/deltaz;
y12=-Z12/deltaz;
y21=-Z21/deltaz;
y22=Z11/deltaz;

printf("(b)The y-parameters are\n");
printf("   y11 = %f + j(%f) mhos\n",real(y11),imag(
    y11));
printf("   y12 = %f + j(%f) mhos\n",real(y12),imag(
    y12));
printf("   y21 = %f + j(%f) mhos\n",real(y21),imag(
    y21));
printf("   y22 = %f + j(%f) mhos\n",real(y22),imag(
    y22));
```

Scilab code Exa 13.2 Obtaining the open circuit impedance parameters

```
clear;
clc;
L=1;C=1;R=1;
printf("z11 = s + %d ohms\n",C);
printf(" z21 = %d ohms\n",C);
printf(" z22 = 1/s + %d ohms\n",C);
printf(" z12 = %d ohms\n",C);
```

Scilab code Exa 13.3 Finding driving point and transfer impedances and loop equations and voltage

```
clear;
clc;
Voc2=100;I1=10;V2=25;
Voc1=100;I2=20;V1=50;
z11=Voc1/I1;
z21=V2/I1;
z22=Voc2/I2;
z12=V1/I2;
printf("(a)The driving point impedances are:\n");
printf("   z11 = %f ohms\n",z11);
printf("   z22 = %f ohms\n\n",z22);
printf("   The transfer impedances are:\n");
printf("   z12 = %f ohms\n",z12);
printf("   z21 = %f ohms\n\n",z21);
printf("   The loop equations are:\n");
printf("   V1= %f*I1 + %f*I2\n",z11,z12);
printf("   V2= %f*I1+ %f*I2\n\n",z21,z22);
Rl=10;
a=[z11 z12 ; z21 (z22+Rl)];
b=[100 ; 0];
b=inv(a)*b;
I2=b(2,1);
Vl=-I2*Rl;
printf("(b)Voltage across resistor = %f volts",round
   (Vl*10)/10);
```

Scilab code Exa 13.4 Computing y parameters

```
clear;
clc;
z11=10;z22=12;z12=5;z21=5;
deltaz=(z11*z22)-(z12*z21);
y11=z22/deltaz;
printf("The y-parameters are:\n");
printf("-y11 = %f mho\n",round(y11*10^4)/10^4);
y22=z11/deltaz;
printf("-y22 = %f mho\n",round(y22*10^4)/10^4);
y12=-z12/deltaz;
printf("-y12 = %f mho\n",round(y12*10^3)/10^3);
y21=y12;
printf("-y21 = %f mho",round(y21*10^3)/10^3);
```

Scilab code Exa 13.5 Finding the z parameters

```
clear;
clc;
Y1=1;Y2=1;Y3=2;V3=3;
Z1=1/Y1;Z2=1/Y2;Z3=1/Y3;
V1=1;I1=-1;
z11=V1/I1;
V2=1;I2=3;
z22=V2/I2;
z21=V2/I1;
printf("z11 = %f ohms\n",z11);
printf(" z22 = %f ohms\n",z22);
printf(" z21 = %f ohms\n",z21);
printf(" z11 = %f ohms\n",0);
```

Scilab code Exa 13.6 Computing h parameters

```
clear;
clc;
Vs1=25;Is1=1;Is2=2; //values with output terminal
   short circuited
Vo1=10;Vo2=50;Io2=2; //values with input terminal
   open circuited
Vs2=0;
h11=Vs1/Is1;
printf("The h-parameters are:\n");
printf("-h11 = %f ohms\n",h11); //with output
   terminals short circuited
h21=Is2/Is1;
printf("-h21 = %f\n",h21); //with input terminals
   open circuited
h12=Vo1/Vo2;
printf("-h12 = %f\n",h12); //with input terminals
   open circuited
h22=Io2/Vo2;
printf("-h22 = %f mho",h22); //with output terminals
```

```
   short circuited
//the difference in result of h22 is due to
   erroneous value in textbook.
disp("The difference in result of h22 is due to
   erroneous value in textbook")
```

Scilab code Exa 13.7 Calculating transmission parameters and loop equations

```
clear;
clc;
z11=40;z22=30;z12=20;z21=20;
deltaz=(z11*z22)-(z12*z21);
A=z11/z12;
printf("The transmission parameters are:\n");
printf("  A = %f\n",A);
B=deltaz/z21;
printf("  B = %f ohms\n",B);
C=1/z21;
printf("  C = %f mho\n",C);
D=z22/z21;
printf("  D = %f\n",D);
printf(" The network equations using z-parameter are
   \n");
printf("  V1 = %fI1 + %fI2\n",z11,z12);
printf("  V2 = %fI1 + %fI2\n",z21,z22);
printf(" The network equations using ABCD parameter
   are\n");
printf("  V1 = %fV2 - %fI2\n",A,B);
printf("  I1 = %fV2 - %fI2\n",C,D);
```

Scilab code Exa 13.8 Obtaining the transmission parameters

```
clear;
clc;
V1=1;R1=1;R2=2;R3=1;R4=2;I2=0;
//I2=0 because port 2-2' has been open circuited
a=[R1+R2 -R2;-2 5];
b=[1;0];
b=inv(a)*b;
I1=b(1,1);
I3=b(2,1);
V2=I3*R4;
A=V1/V2;
C=I1/V2;
V21=0; //because port 2-2' has been short circuited
c=[3 -2;-2 3];
d=[1;0];
d=inv(c)*d;
I11=d(1,1);
I21=d(2,1);
B=V1/I21;
D=I11/I21;
R=(A*D)-(B*C);
printf("The transmission parameters are:\n");
printf("   A = %f\n",A);
printf("   B = %f ohms\n",B);
printf("   C = %f mhos\n",C);
printf("   D = %f\n\n",D);
printf(" AD-BC = %f . Hence the circuit is
    reciprocal.",R);
```

Scilab code Exa 13.9 Calculating input impedance

```
clear;
clc;
Rl=3;z11=5;z12=2;z21=2;z22=1;
Zi=z11-(z12/z21);
```

```
printf("Input impedance = %f ohms",Zi);
```

Scilab code Exa 13.10 Finding ABCD parameters and the equivalent T network

```
clear;
clc;
z11=10;z22=20;z12=5;z21=5;
deltaz=(z11*z22)-(z12*z21);
A=z11/z12;
printf("(a)The ABCD parameters are:\n");
printf("  A = %f\n",A);
B=deltaz/z21;
printf("  B = %f ohms\n",B);
C=1/z12;
printf("  C = %f mho\n",C);
D=z22/z21;
printf("  D = %f\n\n",D);
Z3=z21;
Z1=z11-Z3;
Z2=z22-Z3;
printf("(b)The equivalent T-network:\n");
printf("  Z1 = %f ohms\n",Z1);
printf("  Z2 = %f ohms\n",Z2);
printf("  Z13 = %f ohms\n",Z3);
```

Scilab code Exa 13.11 Obtaining the parameters Ya and Yb and Yc

```
clear;
clc;
V1=10;I1=2.5;I2a=-0.5; //values with terminals 3-4
    short-circuited
```

```
V2=V1;I2b=1.5;  //values with terminals 1-2 short
    circuited
Yb=-I2a/V1;
Ya=(I1/V1)-Yb;
Yc=(I2b/V2)-Yb;
printf("-Ya = %f mho\n",Ya);
printf("-Yb = %f mhp\n",Yb);
printf("-Yc = %f mho",Yc);
```

Scilab code Exa 13.12 Finding the equivalent pi network

```
clear;
clc;
Za=2;Zb=2.5;Zc=5;
Ya=1/Za;Yb=1/Zb;Yc=1/Zc;
Y1=(Ya*Yc)/(Ya+Yb+Yc);
Z1=1/Y1;
Y2=(Yb*Yc)/(Ya+Yb+Yc);
Z2=1/Y2;
Y3=(Ya*Yb)/(Ya+Yb+Yc);
Z3=1/Y3;
printf("The equivalent pi network is: \n");
printf(" Z1 = %f ohms\n",Z1);
printf(" Z2 = %f ohms\n",Z2);
printf(" Z3 = %f ohms\n",Z3);
```

Scilab code Exa 13.13 Obtaining the z and y and ABCD parameters and the equivalent pi network

```
clear;
clc;
y11=0.5;y12=-0.2;y21=-0.2;y22=1;
Y2=-y12;
```

```
Y1=y11-Y2;
Y3=y22-Y2;
deltay=(y11*y22)-(y12*y21);
z11=y22/deltay;
z22=y11/deltay;
z12=y12/deltay;
z21=z12;
A=-y22/y12;
B=-1/y12;
C=-deltay/y12;
D=-y11/y12;
printf("The y parameters are\n");
printf(" y11 = %f mho\n",y11);
printf(" y12 = %f mho\n",y12);
printf(" y21 = %f mho\n",y21);
printf(" y22 = %f mho\n\n",y22);
printf(" The z parameters are\n");
printf(" z11 = %f ohm\n",round(z11*1000)/1000);
printf(" z12 = %f ohm\n",round(z12*10^4)/10^4);
printf(" z21 = %f ohm\n",round(z21*10^4)/10^4);
printf(" z22 = %f ohm\n\n",round(z22*1000)/1000);
printf(" The ABCD parametrs are\n");
printf(" A = %f\n",A);
printf(" B = %f ohm\n",B);
printf(" C = %f mho\n",C);
printf(" D = %f\n\n",D);
printf(" The equivalent pi network is\n");
printf(" Y1 = %f mho\n",Y1);
printf(" Y2 = %f mho\n",Y2);
printf(" Y3 = %f mho",Y3);
```

Scilab code Exa 13.14 Finding y parameters

```
clear;
clc;
```

```
Y1=1;Y2=2;Y3=1;
y11=Y1+Y2;
y12=-Y2;
y21=y12;
y22=Y2+Y3;
Y11=y11+y11;
Y12=y12+y12;
Y21=Y12;
Y22=y22+y22;
printf("The y-parameters are:\n");
printf(" Y11 = %f mho\n",Y11);
printf(" Y12 = %f mho\n",Y12);
printf(" Y21 = %f mho\n",Y21);
printf(" Y22 = %f mho\n",Y22);
```

Scilab code Exa 13.15 Determining the ABCD parameters and image impedance

```
clear;
clc;
Za=10;Zb=10;Zc=5;
Ya=1/Za;Yb=1/Zb;Yc=1/Zc;
A=1+(Za*Yb);
printf("The ABCD parameters are:\n");
printf(" A = %f\n",A);
B=Za+Zb+(Za*Zb*Yc);
printf(" B = %f ohm\n",B);
C=Yc;
printf(" C = %f mho\n",C);
D=1+(Zb*Yc);
printf(" D = %f \n\n",D);
Zi1=sqrt((A*B)/(C*D));
printf(" Image impedances of the two port network
   are:\n");
printf(" Zi1 = %f ohms\n",round(Zi1*10)/10);
Zi2=sqrt((D*B)/(A*C));
```

```
printf(" Zi2 = %f ohms",round(Zi2*100)/100);
```

Scilab code Exa 13.17 Determining the open and short circuit impedance

```
clear;
clc;
R1=2;R2=1;R3=1;R4=2;
A=[R1+R2 R2 -R4;R2 R2+R4 R4;-R1 R1 R4+R3+R1];
B=det(A);
C=[1 R2 -R1;0 R2+R1 R1;0 R4 R4+R3+R1];
D=det(C);
E=[1 -R1;0 R4+R3+R1];
F=[R2+R1 -R1;-R1 R4+R3+R1];
G=[R4+R2 0 -R4;R1 1 R4;-R4 0 R4+R2];
H=[1 R1;0 R4+R3+R1];
I=[R2+R1 R1;R1 R4+R3+R1];
printf("Z1s = %d/%d ohms\n",B,D);
printf(" Z1o = %d/%d ohms\n",det(F),det(E));
printf(" Z2s = %d/%d ohms\n",det(A),det(C));
printf(" Z2o = %d/%d ohms\n",det(I),det(H));
```

Scilab code Exa 13.18 Calculating y parameters

```
clear;
clc;
R1=2;R2=1;R3=2;R4=1;R5=2;
R3=R3/2;
z11=R2+R1;
z22=R3+R2;
z12=R2;
z21=R2;
Z11=z11+z11;
Z12=z12+z12;
```

```
Z21=z21+z21;
Z22=z22+z22;
deltaZ=(Z11*Z22)-(Z12*Z21);
y11=Z22/deltaZ;
y12=-Z12/deltaZ;
y21=y12;
y22=Z11/deltaZ;
printf("The y-parameters are:\n");
printf("-y11 = %f mho\n",y11);
printf("-y12 = %f mho\n",y12);
printf("-y21 = %f mho\n",y21);
printf("-y22 = %f mho\n",y22);
```

Scilab code Exa 13.19 Determining the image impedances

```
clear;
clc;
R1=3;R2=2;R3=2;
A=(R3+R3+R2+R2)/(R1+R3+R1+R2);
Zi1=sqrt((R2*(R1+R3)+(R1*R2))/A);
printf("Zi1 = %f ohms\n",round(Zi1*10)/10);
Zi2=R2+(R3*(R1+Zi1)/(R3+R1+Zi1));
printf(" Zi2 = %f ohms\n",round(Zi2*10)/10);
```

Chapter 14

Filters

Scilab code Exa 14.1 Calculating values of inductor and capacitor

```
clear;
clc;
Rk=600;fc=3000;
L=Rk/(%pi*fc);
printf("-Desired value of inductor L = %f mH\n",
   round(L*(10^3)*10)/10);
C=1/(%pi*Rk*fc);
printf("-Desired value of capacitor C = %f
   microfarads",round(C*(10^6)*10^4)/10^4);
```

Scilab code Exa 14.2 Calculating cut off frequency and attenuation and phase shift

```
clear;
clc;
L=20*(10^-3);C=0.32*(10^-6);f=15*(10^3);
fc=(1/(3.14*(sqrt(L*C))));
printf("Cutoff frequency fc = %f kHz\n",fix(fc*0.1)
   /100);
```

```
Rk=sqrt(L/C);
printf(" Value of nominal terminating impedance Rk =
    %f ohms\n",fix(Rk*10)/10);
a=2*(acosh(f/fc));
printf(" Value of attenuation a = %f db\n",fix(a
   *8.686*100)/100);
printf(" Phase shift of the low pass filter = pi
   radians");
```

Scilab code Exa 14.3 Calculating value of L and C

```
clear;
clc;
fc=10*(10^3);Rk=600;
L=Rk/(4*%pi*fc);
printf("-Value of L = %f mH\n",fix(L*(10^3)*10^3)
   /10^3);
C=1/(4*%pi*fc*Rk);
printf("-Value of C = %f microfarads",fix(C*(10^6)
   *10^5)/10^5);
```

Scilab code Exa 14.4 Determining iterative impedance and cut off frequency and ratio of output voltage to input voltage

```
clear;
clc;
L=60*(10^-3);C=0.2*(10^-6);
fc=1/(%pi*(sqrt(L*C)));
printf("(i)Cut off frequency fc for the low pass
   filter = %f kHz\n\n",round(fc*(10^-1))/100);
fa=1000;
Za=(sqrt(L/C))*(sqrt(1-((fa/fc)^2)));
```

```
printf(" (ii)Iterative impedance at f=1 KHz = %f
   ohms\n",fix(Za));
fb=5000;
Zb=(sqrt(L/C))*(sqrt(1-((fb/fc)^2)));
printf("   Iterative impedance at f=5 KHz = j(%f)
   ohms\n\n",fix(Zb/%i));
aa=0; //attenuation at frequency fa
ba=2*(asind(fa/fc)); //phase shift at frequency fa
ab=2*acosh(fb/fc); //attenuation at frequency fb
bb=180; //attenuation at frequency fb
V1=exp(aa); //V1=(Vin/Vout) at frequency of 1kHz
V2=exp(ab); //V2=(Vin/Vout) at frequency of 5kHz
printf(" (iii)At 5kHz,the voltage ratio = %f and
   phase difference = %f degrees\n",round(V2*10)/10,
   bb);
printf("     At 1kHz,the voltage ratio = %f and
   phase difference = %f degrees\n",V1,fix(ba*10)
   /10);
```

Scilab code Exa 14.5 Determining the elements of a prototype HP T section filter

```
clear;
clc;
fc=1000;Rk=600;
L=Rk/(4*%pi*fc);
C=1/(4*%pi*fc*Rk);
printf("Thus,the series elements are two capacitors
   of value %f microfarad each and shunt inductance
   of value %f mH.",round(C*(10^3)*10^6)/10^5,fix(L
   *(10^3)*100)/100);
```

Scilab code Exa 14.6 Calculating frequency

```
clear;
clc;
a=1.15;
F=cosh(a/2);
printf("The frequency at which low pass filter will
    havean attenuation 10 db will be %f times the cut
     off frequency",round(F*100)/100);
```

Scilab code Exa 14.7 Designing a m derived T section low pass filter

```
clear;
clc;
Rk=600;fc=1000;fi=1050;
L0=Rk/(%pi*fc);
C0=1/(%pi*fc*Rk);
m=round(sqrt(1-((fc/fi)^2))*10)/10;
L1=m*L0/2;
printf("-mL/2 = %f mH\n",round(L1*(10^3)*100)/100);
C=m*C0;
printf("-mC = %f microfarads\n",round(C*(10^6)*1000)
    /1000);
L2=L0*((1-(m*m))/(4*m));
printf("-(1-m^2)L/(4m) = %f mH",L2*(10^3));
//the difference in result is due to erroneous value
     in textbook.
disp("The difference in result is due to erroneous
    value in textbook")
```

Scilab code Exa 14.8 Designing an m derived T section high pass filter

```
clear;
clc;
fc=20*(10^3);Rk=600;m=0.6;
```

```
L0=Rk/(4*%pi*fc);
C0=1/(4*%pi*fc*Rk);
C1=2*C0/m;
printf("-2C/m = %f microfarads\n",round(C1*(10^6)
    *1000)/1000);
L=L0/m;
printf("-L/m = %f mH\n",round(L*(10^3)*100)/100)
C2=C0*((4*m)/(1-(m*m)));
printf("-4mC/(1-m^2) = %f microfarads",round(C2
    *(10^6)*1000)/1000);
```

Scilab code Exa 14.9 Designing a low pass filter

```
clear;
clc;
fc=2000;fi=2050;Rk=500; //fi=frequency at which
    infinite attenuation occurs
L0=Rk/(%pi*fc);
C0=1/(%pi*fc*Rk);
printf("The elements of the constant-K L.P. are:\n")
    ;
printf("  L = %f mH\n",fix(L0*(10^3)*10)/10);
printf("  C = %f microfarads\n\n",fix(C0*(10^6)
    *1000)/1000);
m1=round(sqrt(1-((fc/fi)^2))*100)/100;
L1=m1*L0/2;
L2=(1-(m1*m1))*L0/(4*m1);
C1=m1*C0;
printf("The elements of the m-derived L.P.T. filter
    are:\n");
printf("  mL/2 = %f mH\n",fix(L1*(10^3)*100)/100);
printf("  mC = %f microfarads\n",fix(C1*(10^6)*100)
    /100);
printf("  (1-m^2)L/4m = %f mH\n\n",fix(L2*(10^3)
    *100)/100);
```

```
m2=0.6;
L3=m2*L0/2;
L4=(1-(m2*m2))*L0/(4*m2);
C2=m2*C0;
printf("The elements of the terminating half
   sections m-derived L.P.T. filter are:\n");
printf("  mL/2 = %f mH\n",fix(L3*(10^3)*10)/10);
printf("  mC = %f microfarads\n",fix(C2*(10^6)
   *10000)/10000);
printf("  (1-m^2)L/4m = %f mH\n\n",fix(L4*(10^3)*10)
   /10);
printf("The complete composite filter is constructed
    by using the  constant-K in cascade with the
   short-cut of m-derived section and terminating
   half section");
```

Scilab code Exa 14.10 Designing a high pass composite filter

```
clear;
clc;
fc=1.2*(10^3);fi=1.1*(10^3);Rk=600; //fi=frequency
   at which infinite attenuation occrus
L0=Rk/(4*%pi*fc);
C0=1/(4*%pi*fc*Rk);
printf("The elements of the constant-K H.P. are:\n")
   ;
printf("  L = %f mH\n",fix(L0*(10^3)*100)/100);
printf("  C = %f microfarads\n\n",fix(C0*(10^6)*100)
   /100);
m1=round(sqrt(1-((fi/fc)^2))*10)/10;
C1=2*C0/m1;
L1=L0/m1;
C2=4*m1*C0/(1-(m1*m1));
printf("The elements of the m-derived H.P.T. filter
   are:\n");
```

```
printf("   2C/m = %f microfarads\n",fix(C1*(10^6)
   *100)/100);
printf("   L/m = %f mH\n",round(L1*(10^3)*10)/10);
printf("   4mC/(1-m^2) = %f microfarads\n\n",fix(C2
   *(10^6)*100)/100);
m2=0.6;
C3=2*C0/m2;
L2=L0/m2;
C4=4*m2*C0/(1-(m2*m2));
printf("The elements of the terminating half section
    m-derived H.P.T. filter are:\n");
printf("   2C/m = %f microfarads\n",round(C3*(10^6)
   *100)/100);
printf("   L/m = %f mH\n",round(L2*(10^3)*100)/100);
printf("   4mC/(1-m^2) = %f microfarads\n\n",round(C4
   *(10^6)*100)/100);
printf("The complete composite filter is constructed
    by using the  constant-K in cascade with the
   sharp-cut off m-derived section and terminating
   half section");
```

Scilab code Exa 14.11 Designing a high pass filter and finding the frequency of peak attenuation

```
clear;
clc;
fc=1*(10^6);Rk=75;m=0.6;
L0=Rk/(4*%pi*fc);
C0=1/(4*%pi*fc*Rk);
printf("The elements of the prototype T-section H.P.
    are:\n");
printf("   L = %f mH\n",round(L0*(10^3)*1000)/1000);
printf("   C = %f picofarads\n\n",round(C0*(10^12)));
C1=2*C0/m;
L1=L0/m;
```

```
C2=4*m*C0/(1-(m*m));
printf(" The elements of the terminating half
   section m-derived H.P.T. filter are:\n");
printf("  2C/m = %f picofarads\n",fix(C1*(10^12)));
printf("  L/m = %f mH\n",round(L1*(10^6))/1000);
printf("  Cshunt = %f picofarads\n\n",round(C2
   *(10^12)));
fi=fc*sqrt(1-(m*m));
printf(" Frequency of peak attenuation = %d kHz",fi
   *(10^-3));
```

Scilab code Exa 14.12 Designing a low pass composite filter

```
clear;
clc;
Rk=500;fc=1000;fi1=1065;fi2=1250;
L0=Rk/(%pi*fc);
C0=1/(%pi*fc*Rk);
printf("The elements of the constant-K L.P. are:\n")
   ;
printf("  L = %f mH\n",round(L0*(10^3)));
printf("  C = %f microfarads\n\n",round(C0*(10^6)
   *100)/100);
m1=0.4
L1=m1*L0/2;
L2=(1-(m1*m1))*L0/(4*m1);
C1=m1*C0;
printf("The elements of the m-derived L.P.T. filter
   are:\n");
printf("  mL/2 = %f mH\n",round(L1*(10^3)*10)/10);
printf(" mC = %f microfarads\n",round(C1*(10^6)
   *1000)/1000);
printf("  (1-m^2)L/4m = %f mH\n\n",fix(L2*(10^3)*10)
   /10);
m2=sqrt(1-((fc/fi2)^2));
```

```
L3=m2*L0/2;
L4=(1-(m2*m2))*L0/(4*m2);
C2=m2*C0;
printf("The elements of the terminating half
   sections m-derived L.P.T. filter are:\n");
printf("  mL/2 = %f mH\n",fix(L3*(10^3)*10)/10);
printf("  C = %f microfarads\n",round(C2*(10^6)
   *1000)/1000);
printf("  Lshunt = %f mH\n\n",fix(L4*(10^3)*10)/10);
printf("The complete composite filter is constructed
    by using the  constant-K in cascade with the
   short-cut of m-derived section and terminating
   half section");
```

Scilab code Exa 14.13 Designing a prototype band pass filter

```
clear;
clc;
f1=1000;f2=4000;Rk=600;
C1=(f2-f1)/(4*%pi*Rk*f1*f2);
L1=Rk/(%pi*(f2-f1));
C2=1/(%pi*Rk*(f2-f1));
L2=Rk*(f2-f1)/(4*%pi*f1*f2);
printf("The elements of the prototype band pass
   filter are:\n");
printf("  L1 = %f mH\n",fix(L1*(10^5))/100);
printf("  C1 = %f microfarads\n",round(C1*(10^11))
   /10^5);
printf("  L2 = %f mH\n",round(L2*(10^4))/10);
printf("  C2 = %f microfarads",round(C2*(10^10))
   /10^4);
```

Scilab code Exa 14.14 Designing a prototype T section of a band pass filter

```
clear;
clc;
f1=12000;f2=16000;Rk=600;
C1=(f2-f1)/(4*%pi*Rk*f1*f2);
L1=Rk/(%pi*(f2-f1));
C2=1/(%pi*Rk*(f2-f1));
L2=Rk*(f2-f1)/(4*%pi*f1*f2);
printf("The elements of the prototype band pass
   filter are:\n");
printf("   L1 = %f mH\n",round(L1*(10^5))/100);
printf("   C1 = %f picofarads\n",round(C1*(10^12)));
printf("   L2 = %f mH\n",L2*(10^3));
printf("   C2 = %f microfarads",C2*(10^6));
//the difference in result of L2 and C2 is due to
   erroneous value in textbook.
disp("The difference in result of C2 and L2 is due
   to erroneous value in textbook")
```

Scilab code Exa 14.15 Designing a m derived low pass filter

```
clear;
clc;
fc=5000;fi=1.25*fc;Rk=600; //fi=frequency at which
   infinite attenuation occus
L0=Rk/(%pi*fc);
C0=1/(%pi*fc*Rk);
m=sqrt(1-((fc/fi)^2));
L1=m*L0/2;
L2=(1-(m*m))*L0/(4*m);
C1=m*C0;
printf("The elements of the m-derived L.P.T. filter
   are:\n");
```

```
printf("  mL/2 = %f mH\n",round(L1*(10^5))/100);
printf("  mC = %f microfarads\n",fix(C1*(10^10))
    /10^4);
printf("  (1-m^2)L/4m = %f mH",round(L2*(10^4))/10);
```

Scilab code Exa 14.16 Determining the values of shun arm of network

```
clear;
clc;
L=0.5*(10^-3);C=0.01*(10^-6);Rk=600;
L1=2*L;
C1=C/2;
L2=Rk*Rk*C1;
C2=L1/(Rk*Rk);
printf("The elements of the shunt arm will be an
    inductance of %f mH in parallel with a
    capacitance of %f microfarads",L2*(10^3),round(C2
    *(10^10))/10^4);
```

Scilab code Exa 14.17 Designing a low pass composite filter

```
clear;
clc;
fc=2400;fi=2500;Rk=600; //fi=frequency at which
    infinite attenuation occus
L0=Rk/(3.14*fc);
C0=1/(3.14*fc*Rk);
printf("The elements of the constant-K L.P. are:\n")
    ;
printf("  L = %f mH\n",round(L0*(10^5))/100);
printf("  C = %f microfarads\n\n",round(C0*(10^10))
    /10^4);
m1=sqrt(1-((fc/fi)^2));
```

```
L1=m1*L0/2;
L2=(1-(m1*m1))*L0/(4*m1);
C1=m1*C0;
printf("The elements of the m—derived L.P.T. filter
   are:\n");
printf("  mL/2 = %f mH\n",round(L1*(10^5))/100);
printf("  mC = %f microfarads\n",round(C1*(10^10))
   /10^4);
printf("  (1—m^2)L/4m = %f mH\n\n",round(L2*(10^5))
   /100);
m2=0.6;
L3=m2*L0/2;
L4=(1-(m2*m2))*L0/(2*m2);
C2=m2*C0/2;
printf("The elements of the terminating half
   sections m—derived L.P.T. filter are:\n");
printf("  mL/2 = %f mH\n",round(L3*(10^5))/100);
printf("  mC/2 = %f microfarads\n",round(C2*(10^10))
   /10^4);
printf("  (1—m^2)L/2m = %f mH\n\n",fix(L4*(10^5))
   /100);
printf("The complete composite filter is constructed
    by using the  constant—K in cascade with the
   short—cut of m—derived section and terminating
   half section");
```

Scilab code Exa 14.18 Determining the cut off frequency and characteristic impedance

```
clear;
clc;
C0=1*(10^-6);L0=10*(10^-3);
C=C0/2;
L=L0;
Rk1=sqrt(L/C);
```

```
fc1=1/(4*%pi*sqrt(L*C));
printf("The cut-off frequency(high pass) = %f Hz\n",
   round(fc1));
printf(" The characteristic impedance(high pass) =
   %f ohms\n\n",round(Rk1*10)/10);
C1=C0*2;
Rk2=sqrt(L/C1);
fc2=1/(%pi*sqrt(L*C1));
printf(" The cut-off frequency(low pass) = %f Hz\n",
   fix(fc2));
printf(" The characteristic impedance(low pass) = %f
    ohms",fix(Rk2*10)/10);
```

Scilab code Exa 14.19 Designing a low pass composite filter

```
clear;
clc;
Rk=500;fc=4000;fi=5000; //fi=frequency at which
   infinite attenuation occurs
L0=Rk/(%pi*fc);
C0=1/(%pi*fc*Rk);
printf("The elements of the constant-K L.P. are:\n")
   ;
printf("  L = %f mH\n",round(L0*(10^4))/10);
printf("  C = %f microfarads\n\n",round(C0*(10^9))
   /1000);
m1=sqrt(1-((fc/fi)^2));
L1=m1*L0/2;
L2=(1-(m1*m1))*L0/(4*m1);
C1=m1*C0;
printf("The elements of the m-derived L.P.T. filter
   are:\n");
printf("  mL/2 = %f mH\n",round(L1*(10^5))/100);
printf("  C = %f microfarads\n",round(C1*(10^10))
   /10^4);
```

```
printf("  Lshunt = %f mH\n\n",round(L2*(10^5))/100);
m2=0.6;
L3=m2*L0/2;
L4=(1-(m2*m2))*L0/(2*m2);
C2=m2*C0/2;
printf("The elements of the terminating half
   sections m-derived L.P.T. filter are:\n");
printf("  Lseris = %f mH\n",round(L3*(10^5))/100);
printf("  C = %f microfarads\n",round(C2*(10^10))
   /10^4);
printf("  Lshunt = %f mH\n\n",round(L4*(10^5))/100);
printf("The complete composite filter is constructed
    by using the  constant-K in cascade with the
   short-cut of m-derived section and terminating
   half section");
//the difference in result of the elements of the m
   derived L.P. T section is due to erroneous value
   in textbook.
disp("The difference in result of the elements of
   the m derived L.P. T section is due to erroneous
   value in textbook")
```

Scilab code Exa 14.20 Determining the second resonance frequency

```
clear;
clc;
fr=1*(10^6);C=0.04*(10^-12);C1=6*(10^-12);
fa=fr*(1+(C/(2*C1)));
printf("The second resonance frequency = %f MHz",
   round(fa*(10^-2))/10^4);
```

Scilab code Exa 14.21 Computing the values of elements of filter

```
clear;
clc;
Rk=600;f1=120*(10^3);f2=123*(10^3);
C1=(f2-f1)/(4*%pi*Rk*f1*f2);
L1=Rk/(%pi*(f2-f1));
C2=L1/(Rk*Rk);
L2=Rk*(f2-f1)/(4*%pi*f1*f2);
printf("The elements of the T-type constant k band
   pass filter are:\n");
printf(" L1 = %f mH\n",fix((L1)*(10^4))/10);
printf(" C1 = %f picofarads\n",fix((C1)*(10^14))
   /100);
printf(" L2 = %f microhenry\n",fix((L2)*(10^7))/10);
printf(" C2 = %f microfarads",round((C2)*(10^9))
   /1000);
```

Scilab code Exa 14.22 Calculating bandwidth

```
clear;
clc;
C=1/100;fo=1000;  //C=C1/C2
//value of fo as taken in solution
f1=fo*((sqrt(C+1))-sqrt(C));
f2=fo*((sqrt(C+1))+sqrt(C));
BW=f2-f1;
printf("Bandwidth = %d Hz",(BW));
```

Scilab code Exa 14.23 Finding the values of the elements of a prototype filter

```
clear;
clc;
Ro=600;f=120;fo=1500;  //f=f2-f1
```

```
F=fo*fo; //F=f1*f2
l1=Ro/(2*%pi*f);
c1=round(f*10^11/(2*Ro*%pi*F))/10^5;
c2=round(1*10^8/(%pi*Ro*f))/100;
l2=Ro*f/(4*%pi*F);
n=(1/2)*(sqrt(((2*c2/(c1/2))+9)-1));
C1=((2*n)-1)*c1*10^-6/(2*n);
C2=c1*10^-6/n;
L=n*n*l2;
C3=c2*10^-6/(n*n);
printf("The elements of the filter are:\n");
printf(" (2n-1)C1/n = %f microfarads\n",fix(C1
   *(10^9))/1000);
printf(" 2C1/n = %f microfarads\n",fix((C2)*(10^10))
   /10^4);
printf(" (n^2)L2  = %f H\n",round(L*10)/10);
printf(" C2/(n^2) = %f microfards",round((C3)*(10^9)
   )/1000);
```

Chapter 15

Equalizers

Scilab code Exa 15.4 Finding the elements of the lattice

```
clear;
clc;
R0=600;R1=400;L=40*(10^-3);
R2=R0*R0/R1;
printf("-Other arm of lattice equalizer will have a
   resistance of R2 = %f ohms\n",R2);
C=L/(R0*R0);
printf(" -R2 resistance will be in parallel with a
   capacitance of C = %f microfarads",round(C*(10^8)
   )/100);
```

Scilab code Exa 15.5 Calculating values of L and C

```
clear;
clc;
R=600;f=4000;
a=3;
w=2*%pi*f;
```

```
C=sqrt((exp(2*3*0.115)-1)/(4*w*w*R*R));
printf("-C = %f microfarads\n",round(C*(10^9))/1000)
    ;
L=2*C*R*R;
printf(" -L = %f mH",fix(L*(10^5))/100);
```

Scilab code Exa 15.6 Calculating characteristic impedance and the components of the shunt arm of equalizer

```
clear;
clc;
R1=1000;C1=0.0212*(10^-6);R2=250;
R0=sqrt(R1*R2);
printf("Characteristic impedance of line = %f ohms\n
    ",R0);
L2=C1*R0*R0;
printf("Components of the shunt arm are inductance
    of %f mH in parallel with a given resistance of
    %f ohms.",L2*(10^3),R0);
```

Scilab code Exa 15.10 Calculating value of R and L

```
clear;
clc;
R1=1;C=0.05;R0=1;
R2=R0*R0/R1;
printf("-Series arm will have a resistance R2 = %f
    ohms\n",R2);
L=C*R0*R0;
printf(" -Value of inductance in parallel with R2 =
    %f henry",L);
```

Scilab code Exa 15.11 Designing a constant resistance equalizer

```
clear;
clc;
f=500;Ro=400;f1=50;D1=17;
M1=10^(D1/10);
D2=4;f2=2500;
M2=10^(D2/10);
B=sqrt(((f1*f1*(M1-1))-(f2*f2*(M2-1)))/(M2-M1));
A=sqrt((B*B*M2)+(f2*f2*(M2-1)));
L11=Ro/(%pi*(A+B));
L12=Ro/(%pi*(A-B));
R11=L11*%pi*(A-B);
R12=L12*%pi*(A+B);
R21=Ro*Ro/R11;
R22=Ro*Ro/R12;
C21=L11/(Ro*Ro);
C22=L12/(Ro*Ro);
printf("The designed equalizer will have the
   configuration:\n");
printf(" I. R1 = %f ohms, C2 = %f microfarads, R2 =
   %f ohms, L1 = %f mH\n",round(R11),round(C21
   *(10^9))/1000,fix(R21),round(L11*(10^4))/10);
printf(" II. R1 = %f ohms, C2 = %f microfarads, R2 =
    %f ohms, L1 = %f mH\n\n",fix(R12),round(C22
   *(10^9))/1000,round(R22),round(L12*(10^3)));
M=((A*A)+(f*f))/((B*B)+(f*f));
F=10*log10(M);
printf(" Loss aat 500Hz = %f db",fix(F*10)/10);
```

Scilab code Exa 15.12 Designing a constant resistance equalizer

```
clear;
clc;
R0=600;D=10;b=10/6;fr=8.5*(10^3);
k=round((10^(D/20)*100))/100;
fb=fr/b;
Cb=1/(2*%pi*fb*R0);
Lb=R0/(2*%pi*fb);
printf("The desired bridged-T constant resistance
   equalizer will be as:\n");
printf("-Ro = %f ohms\n",R0);
L1=Lb*((k-1)/sqrt(k))*(((b*b)-1)/(b*b));
printf("-L1 = %f Henry\n",round(L1*10^4)/10^4);
C1=Cb*(sqrt(k)/(k-1))*(1/((b*b)-1));
printf("-C1 = %f microfarads\n",round(C1*(10^10))
   /10^4);
L2=Lb*((sqrt(k))/(k-1))*(1/((b*b)-1));
printf("-L2 = %f Henry\n",round(L2*10^5)/10^5);
C2=Cb*((k-1)/sqrt(k))*(((b*b)-1)/(b*b));
printf("-C2 = %f microfarads\n",round(C2*(10^10))
   /10^4);
R1=R0*(k-1);
printf("-R1 = %f ohms\n",round(R1));
R2=R0/(k-1);
printf("-R2 = %f ohms",round(R2));
```

Scilab code Exa 15.13 Designing a lattice equalizer

```
clear;
clc;
f1=200;f2=2400;Li=2.6;Lc1=0.494;Lc2=1.949;Ro=600;
//Li=total insertion loss
//value of f1 as taken in solution
//Lc=cable loss
Le1=Li-Lc1; //Le=equalizer loss
Le2=Li-Lc2;
```

```
M1=fix(exp(2*Le1));
M2=round(exp(2*Le2)*100)/100;
Q=((f2*f2*(M2-1))-(f1*f1*(M1-1)))/(M1-M2);
P=(f1*f1*(M1-1))+(M1*Q);
R11b=Ro*(sqrt(P)+sqrt(Q))/(sqrt(P)-sqrt(Q));
R21b=Ro*Ro/R11b;
L12b=Ro/(%pi*(sqrt(P)-sqrt(Q)));
C22b=L12b/(Ro*Ro);
L12a=Ro/(%pi*(sqrt(P)+sqrt(Q)));
R11a=Ro*(sqrt(P)-sqrt(Q))/(sqrt(P)+sqrt(Q));
R21a=Ro*Ro/R11a;
C22a=L12a/(Ro*Ro);
printf("The required equalizer will have the
    follloving configuration:\n");
printf("I. R11 = %f ohms\n",fix(R11a));
printf("-L12 = %f mH\n",round(L12a*(10^4))/10);
printf("-R21 = %f ohms\n",round(R21a));
printf("-C22 = %f microfarads\n\n",round(C22a*(10^9)
    )/10^3);
printf("II. R11 = %f ohms\n",round(R11b));
printf("-L12 = %f mH\n",fix(L12b*(10^4))/10);
printf("-R21 = %f ohms\n",round(R21b*100)/100);
printf("-C22 = %f microfarads",fix(C22b*(10^9))
    /10^3);
```

Chapter 16

Attenuators

Scilab code Exa 16.1 Designing a symmetrical bridge T network

```
clear;
clc;
D=40;Ro=600;
N=10^(D/20);
R3=Ro*(N-1);
R2=Ro/(N-1);
R1=Ro;
printf("R1 = %f ohms\n",R1);
printf(" R2 = %f ohms\n",round(R2*100)/100);
printf(" R3 = %f k ohms\n",R3*(10^-3));
```

Scilab code Exa 16.2 Designing a T pad attenuator

```
clear;
clc;
D=20;Ro=600;
N=10^(D/20);
R1=Ro*(N-1)/(N+1);
```

```
R2=Ro*2*N/((N*N)-1);
printf("R1 = %f ohms\n",round(R1*10)/10);
printf(" R2 = %f ohms\n",round(R2*10)/10);
```

Scilab code Exa 16.3 Designing a T type attenuator

```
clear;
clc;
D=20;Ro=75;
N=10^(D/20);
R1=Ro*(N-1)/(N+1);
R2=Ro*2*N/((N*N)-1);
printf("R1 = %f ohms\n",round(R1*100)/100);
printf(" R2 = %f ohms\n",round(R2*100)/100);
```

Scilab code Exa 16.4 Designing an attenuator and determining its equivalent T structure

```
clear;
clc;
D=20;Ro=500;
N=10^(D/20);
R1=Ro*(N-1)/(N+1);
R2=Ro*(N+1)/(N-1);
printf("The elements of the attenuator are:\n");
printf("R1 = %f ohms\n",round(R1*100)/100);
printf(" R2 = %f ohms\n\n",round(R2*100)/100);
r1=R1;
r2=(R2-R1)/2;
printf("The equivalent T structure of the designed
    lattice:\n");
printf(" R1 = %f ohms\n",round(r1*100)/100);
printf(" R2 = %f ohms\n",round(r2*100)/100);
```

Scilab code Exa 16.5 Finding the elements of the arm of a T section symmetrical resistive attenuator

```
clear;
clc;
D=15;Ro=75;
N=10^(D/20);
R1=Ro*(N-1)/(N+1);
R2=Ro*2*N/((N*N)-1);
printf("The arms of the T section will contain:\n");
printf(" R1 = %f ohms\n",round(R1*100)/100);
printf(" R2 = %f ohms\n",round(R2*100)/100);
//the difference in result of R1 is due to erroneous
    value in textbook.
disp("The difference in result of R1 is due to
   erroneous value in textbook")
```

Scilab code Exa 16.6 Calculating the characteristic and attenuation per section of an attenuator

```
clear;
clc;
R1=175;R2=350;
y=poly([1 -3 1],"N","coeff");
f=roots(y);
N=f(1,1);
D=20*log10(N);
Ro=R1*(N+1)/(N-1);
printf("The characteristic impedance of T attenuator
    = %f ohms and its attenuation per section is %f
   db",round(Ro*10)/10,round(D*100)/100);
```

Scilab code Exa 16.7 Determining the resistance value of a T type attenuator pad

```
clear;
clc;
D=40;Ri1=70;Ri2=600;
Ai=D*0.115;
R3=(sqrt(Ri1*Ri2))/sinh(Ai);
R2=(Ri2/tanh(Ai))-R3;
R1=(Ri1/tanh(Ai))-R3;
printf("The desired elements of T-pad are:\n");
printf(" R1 = %f ohms\n",round(R1*100)/100);
printf(" R2 = %f ohms\n",round(R2));
printf(" R3 = %f ohms\n",round(R3*100)/100);
```

Scilab code Exa 16.8 Finding the elements of a balanced T pad

```
clear;
clc;
D=10;Ri1=150;Ri2=75;
Ai=D*0.115;
R3=(sqrt(Ri1*Ri2))/sinh(Ai);
R2=(Ri2/tanh(Ai))-R3;
R1=(Ri1/tanh(Ai))-R3;
printf("The desired elements of T-pad are:\n");
printf(" R1 = %f ohms\n",round(R1*100)/100);
printf(" R2 = %f ohms\n",round(R2*10)/10);
printf(" R3= %f ohms\n\n",fix(R3*100)/100);
R2a=sqrt((Ri1*Ri2*Ri2)/(Ri1-Ri2));
R1a=sqrt(Ri1*(Ri1-Ri2));
printf("The minimum loss pad will be a L attenuator
    :\n");
```

```
printf(" R1 = %f ohms\n",round(R1a));
printf(" R2 = %f ohms\n",round(R2a));
```

Scilab code Exa 16.9 Designing L type attenuator

```
clear;
clc;
Ri1=500;D=15;
N=10^(D/20);N1=fix(N*1000)/1000;
R2=Ri1/(N1-1);
R1=Ri1*(1-(1/N));
printf("The series arm of the L attenuator is %f
   ohms,while its shunt arm is %f ohms",round(R1),
   round(R2));
//the difference in result of R2 is due to erroneous
    value in textbook.
disp("The difference in result of R2 is due to
   erroneous value in textbook")
```

Scilab code Exa 16.10 Designing a T pad

```
clear;
clc;
Ri1=50;Ri2=200;p=5;
d=p/100;
Ai=atanh(1/(1+d));A=fix(Ai*100)/100;
R3=(sqrt(Ri1*Ri2))/sinh(A);
R2=(Ri2/tanh(A))-R3;
R1=(Ri1/tanh(A))-R3;
printf("The desired elements of T-pad are:\n");
printf(" R1 = %f ohms\n",round(R1));
printf(" R2 = %f ohms\n",fix(R2*10)/10);
printf(" R3 = %f ohms\n",round(R3*10)/10);
```

Scilab code Exa 16.11 Designing an unbalances pi attenuator

```
clear;
clc;
D=20;Ri1=200;Ri2=500;
Ai=D*0.115;
Gi1=1/Ri1;Gi2=1/Ri2;
G3=(sqrt(Gi1*Gi2))/sinh(Ai);
G2=(Gi2/tanh(Ai))-G3;
G1=(Gi1/tanh(Ai))-G3;
R3=1/G3;R2=1/G2;R1=1/G1;
printf("The desired pi attenuator will be:\n");
printf(" R1 = %f ohms\n",round(R1*10)/10);
printf(" R2 = %f ohms\n",round(R2*10)/10);
printf(" R3 = %f ohms\n",round(R3));
//the difference in result of R3 is due to erroneous
    value in textbook.
disp("The difference in result of R3 is due to
   erroneous value in textbook")
```

Scilab code Exa 16.12 Designing a T attenuator

```
clear;
clc;
R1=500;R2=110;
Ro=sqrt((2*2*R1*R1/4)+(2*R1*R2));
a=acosh(1+(2*R1/(2*R2)));
Ri1=Ro;Ri2=150;Ai=round(a*10)/10;
R3=(sqrt(Ri1*Ri2))/(fix(sinh(Ai)*100)/100);
R2=(Ri2/(round(tanh(Ai)*10^4)/10^4))-(fix(R3*100)
   /100);
```

```
R1=(Ri1/(round(tanh(Ai)*10^4)/10^4))-R3;
printf("The desired elements of the attenuator are:\
   n");
printf("R1 = %f ohms\n",fix(R1*100)/100);
printf("R2 = %f ohms\n",fix(R2*100)/100);
printf("R3 = %f ohms\n",fix(R3*100)/100);
```

Scilab code Exa 16.13 Designing a ladder attenuator

```
clear;
clc;
Ro=600;D=5;
N=10^(D/20);
R1=Ro*(N+1)/(N-1);
R2=Ro*((N*N)-1)/(2*N);
printf("The desired ladder attenuator will be:\n");
printf("R1 = %f ohms\n",round(R1));
printf("R2 = %f ohms\n",round(R2));
//the difference in result of R1 is due to erroneous
    value in textbook.
disp("The difference in result of R1 is due to
   erroneous value in textbook")
```

Scilab code Exa 16.14 Designing an attenuator

```
clear;
clc;
d=12;Ro=500;
D=(d-3)/3;
N=round(10^(D/20)*1000)/1000;
R1=Ro*(N+1)/(N-1);
R2=Ro*(round((N*N))-1)/(2*N);
printf("The desired attenuator will be:\n");
printf("R1 = %f ohms\n",round(R1));
printf("R2 = %f ohms\n",round(R2));
//the difference in result of R1 is due to erroneous
    value in textbook.
disp("The difference in result of R1 is due to
   erroneous value in textbook")
```

Scilab code Exa 16.15 Designing a balanced attenuator

```
clear;
clc;
d1=0;d2=5;d3=10;d4=15;d5=20;Ro=600;
printf("For 0 db loss:\n");
printf("  R1 = 0\n  R2 = infinite\n\n");
n=4;
for i=1:n
    N=10^(5*i/20);
    R2=Ro*(N-1);
    R1=Ro/(N-1);
    printf("\n For %d db loss:\n",5*i);
    printf("  R1=%f\n",round(R1));
    printf("  R2=%f\n",round(R2));
end
```

Scilab code Exa 16.16 Finding the attenuation and the values of the shunt resistance of an H type attenuator

```
clear;
clc;
Ro=600;R1=240;
R2=((Ro*Ro)-(4*R1*R1))/(4*R1);
d=acosh(1+(2*R1/R2));
printf("Value of shunt resistance = %d ohm\n",R2);
printf(" Attenuation = %f db",round(d*8.686*10)/10);
```

Scilab code Exa 16.17 Designing a L type attenuator

```
clear;
clc;
Ri1=500;d=15;
P=d/8.686;
N=exp(P);
R1=Ri1*(1-(1/N));
R2=Ri1/(N-1);
printf("The desired attenuator will be:\n");
printf(" R1 = %f ohms\n",round(R1));
printf(" R2 = %f ohms\n",round(R2));
```

Scilab code Exa 16.18 Designing a minimum loss pad

```
clear;
clc;
Ri1=72;Ri2=52;
R1=Ri1*sqrt(1-(Ri2/Ri1));
R2=Ri2/sqrt(1-(Ri2/Ri1));
x=sqrt(Ri1/Ri2);
L=20*log10(x+(sqrt((x*x)-1)));
printf("The desired elements will be:\n");
printf(" R1 = %f ohms\n",round(R1));
printf(" R2 = %f ohms\n",fix(R2*10)/10);
printf(" Value of loss produced by the network = %f
    db",round(L));
```