

Scilab Textbook Companion for
Transmission Lines And Networks
by U. Sinha¹

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

July 31, 2019

¹Funded 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	4
1 Transmission Lines	5
2 Open and Short Circuited Lines	16
3 Line with any Termination	25
4 Power Lines	37
5 Telegraph and Telephone Lines	41
6 Ultrahigh Frequency Lines	52
7 Waveguides	60
8 Transmission Lines Measurements	64
9 Artificial Lines	66
11 Networks	70
12 Network Theorems	85
13 Network Parameters	100

14 Filters	112
15 Equalizers	128
16 Attenuators	133

List of Scilab Codes

Exa 1.1	Calculating the resistance and inductance and leakage and capacitance per km and velocity of propagation	5
Exa 1.2	Calculating the values of the line constants .	6
Exa 1.3	Determining Z_0 and a and b	6
Exa 1.4	Calculating terminating impedance and attenuation and phase constant	7
Exa 1.5	Calculating phase velocity and characteristic impedance and propagation constant and power delivered at receiving end	8
Exa 1.6	Calculating characteristic impedance and propagation constant	8
Exa 1.7	Calculating Z_0 and a and b and V_p	9
Exa 1.8	Calculating characteristic impedance	10
Exa 1.9	Calculating phase velocity and attenuation .	10
Exa 1.10	Calculating characteristic impedance	11
Exa 1.11	Calculating the voltage across load in terms of the sending end voltage	11
Exa 1.12	Calculating dc resistance and inductance and capacitance and ac resistance	12
Exa 1.13	Calculating per km high and low frequency inductance and capacitance and dc and ac resistance	13
Exa 1.14	Calculating line attenuation and velocity of propagation	14
Exa 1.15	Calculating sending and receiving end voltages and currents and power	14
Exa 2.1	Calculating Z_0	16

Exa 2.2	Calculating the line constants	16
Exa 2.3	Calculating the received current and voltage	17
Exa 2.4	Determining Z_0 and a and b and primary constants of line	18
Exa 2.5	Calculating the line constants	19
Exa 2.6	Calculating Z_0 and P and line constants . .	20
Exa 2.7	Calculating the series impedance and shunt admittance per km of line	21
Exa 2.8	Calculating Z_0 and a and b	21
Exa 2.9	Calculating output current when far end is short circuited	22
Exa 2.10	Calculating Z_0 and a and b	23
Exa 2.11	Calculating Z_0 and a and b	23
Exa 3.1	Calculating sending end voltage	25
Exa 3.2	Calculating rms value voltage and current .	25
Exa 3.3	Calculating the rms value of voltage and current	26
Exa 3.4	Calculating the input impedance of line . .	26
Exa 3.5	Calculating sending end impedance	27
Exa 3.6	Calculating input impedance	27
Exa 3.7	Calculating the value of voltage reflection coefficient	28
Exa 3.8	Calculating the ratio of transmitter voltage to receiver voltage	28
Exa 3.9	Calculating the value of voltage reflection coefficient	29
Exa 3.10	Calculating power delivered at receiving end	29
Exa 3.11	Calculating sending end current and receiving end current and voltage and efficiency of transmission	30
Exa 3.12	Calculating the resistance and inductance of the series and shunt elements	31
Exa 3.13	Determining the characteristic impedance and propagation constant	31
Exa 3.14	Calculating voltage and current	32
Exa 3.15	Calculating input impedance	33
Exa 3.16	Calculating power consumed in load	33

Exa 3.17	Calculating the exact and approximate distance to fault	34	
Exa 3.18	Calculating the reflection factor and reflection loss	35	
Exa 3.19	Calculating reflection coefficient and input impedance		35
Exa 3.20	Finding reflection loss	36	
Exa 4.2	Determining equivalent pi network	37	
Exa 4.3	Calculating inductance per km of line	37	
Exa 4.4	Calculating characteristic impedance and propagation constant	38	
Exa 4.5	Calculating inductive reactance	38	
Exa 4.6	Calculating capacitive susceptance	39	
Exa 4.7	Determining the capacitance per km and charging current	39	
Exa 5.1	Calculating decrease in attenuation and cut off frequency	41	
Exa 5.2	Calculating Z_0 and a and b and V_p	41	
Exa 5.3	Calculating modified Z_0 and a and V_p	42	
Exa 5.4	Calculating the phase velocity before and after velocity and decrease in attenuation and cut off frequency	43	
Exa 5.5	Calculating attenuation and cut off frequency	44	
Exa 5.6	Calculating attenuation and highest frequency of transmission	44	
Exa 5.7	Calculating the value of Z_0 and a and V_p after loading	45	
Exa 5.8	Calculating cut off frequency and attenuation	46	
Exa 5.9	Calculating Z_0 and a and b	46	
Exa 5.10	Calculating length of line for a given attenuation	47	
Exa 5.11	Calculating cutoff frequency and attenuation and phase velocity	47	
Exa 5.12	Finding the value of the loading coil inductance and the interval at which it needs to be placed	48	
Exa 5.13	Calculating rate of transmission of information	48	
Exa 5.14	Calculating a and b and cut off frequency	49	
Exa 5.15	Calculating wavelength of line	50	

Exa 5.16	Calculating the value of attenuation	50
Exa 5.17	Calculating potential at mid point	51
Exa 6.1	Calculating complex reflection coefficient and terminal impedance	52
Exa 6.2	Calculating value of load impedance	53
Exa 6.3	Calculating VSWR and position of voltage minimum nearest to load	53
Exa 6.4	Calculating standing wave ratio	54
Exa 6.5	Calculating value of load impedance	54
Exa 6.11	Calculating point of attachment and length of stub	55
Exa 6.14	Calculating point of attachment and length of stub	55
Exa 6.15	Calculating point of attachment and length of stub	56
Exa 6.17	Calculating the dimensions of a quarter wave line	56
Exa 6.18	Designing a quarter wave transformer	57
Exa 6.19	Designing a single stub matching given system to eliminate standing wave ratio	57
Exa 6.20	calculating frequency and terminated impedance	58
Exa 6.21	Calculating load and input impedance	58
Exa 7.1	Calculating critical and guide wavelengths	60
Exa 7.2	Finding the value of the dominant mode	60
Exa 7.3	Calculating the cut off wavelength and guide wavelength and group and phase velocities and the characteristic wave impedance	61
Exa 7.4	Calculating voltage attenuation	61
Exa 7.5	Calculating group and phase velocities and phase constant	62
Exa 7.6	Calculating cut off frequencies	62
Exa 7.7	Calculating cut off wavelength and guide wavelength and characteristic wave impedance	63
Exa 8.1	Calculating the value of load impedance	64
Exa 8.2	Calculating insertion loss	64
Exa 8.3	Calculating insertion loss	65
Exa 9.1	Designing an artificial line	66
Exa 9.2	Designing an artificial line	67

Exa 9.3	Designing a delay line using T section	67
Exa 9.4	Designing an artificial line	68
Exa 11.1	Calculating image and iterative impedance .	70
Exa 11.2	Calculating iterative impedance	71
Exa 11.3	Calculating iterative and image impedance .	71
Exa 11.4	Calculating image and iterative impedances and transfer constants	72
Exa 11.5	Computing the insertion ratio and loss . . .	73
Exa 11.6	Determining the components of a symmetrical T section network	74
Exa 11.7	Calculating current	74
Exa 11.8	Calculating characteristic impedance	74
Exa 11.9	Designing a L section	75
Exa 11.10	Designing a reactive T network	75
Exa 11.11	Determining the equivalent T network . . .	76
Exa 11.12	Finding the elements of the arms of a loss less T network	77
Exa 11.13	Determining the image and iterative impedance and equivalent T network	77
Exa 11.14	Calculating the characteristic impedance and propagation function of given network . . .	78
Exa 11.15	Calculating characteristic impedance and prop- agation function	79
Exa 11.16	Designing a L matching loss less network . .	79
Exa 11.17	Calculating different losses and total insertion loss	80
Exa 11.18	Calculating the elements of a L section net- work	81
Exa 11.19	Calculating image impedance and transfer con- stant	82
Exa 11.20	Calculating reduction in power	82
Exa 11.21	Calculating characteristic impedance and at- tenuation constant and phase shift constant	83
Exa 11.22	Calculating image impedance and propaga- tion constant and elements of T network . .	83
Exa 12.1	Finding the rms value of current	85
Exa 12.2	Calculating voltage	85
Exa 12.3	Calculating current	86

Exa 12.5	Determining Thevenins equivalent generator	86
Exa 12.6	Determining Thevenins equivalent circuit . .	87
Exa 12.7	Deriving the equivalent voltage generator circuit	87
Exa 12.8	Determining the Nortons equivalent circuit .	88
Exa 12.9	Obtaining Nortons equivalent circuit	89
Exa 12.10	Verifying reciprocity theorem	89
Exa 12.11	Calculating value of compensation source . .	89
Exa 12.12	Calculating power in load and turns ratio for maximum power transfer	90
Exa 12.13	Determining value of maximum power . . .	90
Exa 12.14	Calculating value of R and X which result in maximum power transfer and value of maximum power transferred	91
Exa 12.17	Calculating effective resistance of coil and inductance of the effective resistance	91
Exa 12.19	Determining the elements of a series type Foster network	92
Exa 12.20	Determining the elements of series type Foster network	93
Exa 12.21	Finding current	93
Exa 12.22	Using Thevenins theorem to calculate power	94
Exa 12.23	Finding Thevenins equivalent generator . . .	94
Exa 12.25	Determining the value of load impedance for which maximum power is consumed in load and the corresponding power	95
Exa 12.26	Determining Thevenins equivalent circuit . .	95
Exa 12.27	Calculating current	96
Exa 12.28	Using Millmans theorem to find current and voltage	96
Exa 12.29	Verifying Tellegens theorem	97
Exa 12.30	Verifying Tellegens theorem	97
Exa 12.31	Finding current	98
Exa 12.32	Calculating current and voltage using Thevenins theorem	98
Exa 13.1	Determining z and y parameters	100
Exa 13.2	Obtaining the open circuit impedance parameters	101

Exa 13.3	Finding driving point and transfer impedances and loop equations and voltage	101
Exa 13.4	Computing y parameters	102
Exa 13.5	Finding the z parameters	102
Exa 13.6	Computing h parameters	103
Exa 13.7	Calculating transmission parameters and loop equations	104
Exa 13.8	Obtaining the transmission parameters . . .	104
Exa 13.9	Calculating input impedance	105
Exa 13.10	Finding ABCD parameters and the equivalent T network	106
Exa 13.11	Obtaining the parameters Ya and Yb and Yc	106
Exa 13.12	Finding the equivalent pi network	107
Exa 13.13	Obtaining the z and y and ABCD parameters and the equivalent pi network	107
Exa 13.14	Finding y parameters	108
Exa 13.15	Determining the ABCD parameters and image impedance	109
Exa 13.17	Determining the open and short circuit impedance	110
Exa 13.18	Calculating y parameters	110
Exa 13.19	Determining the image impedances	111
Exa 14.1	Calculating values of inductor and capacitor	112
Exa 14.2	Calculating cut off frequency and attenuation and phase shift	112
Exa 14.3	Calculating value of L and C	113
Exa 14.4	Determining iterative impedance and cut off frequency and ratio of output voltage to input voltage	113
Exa 14.5	Determining the elements of a prototype HP T section filter	114
Exa 14.6	Calculating frequency	114
Exa 14.7	Designing a m derived T section low pass filter	115
Exa 14.8	Designing an m derived T section high pass filter	115
Exa 14.9	Designing a low pass filter	116
Exa 14.10	Designing a high pass composite filter . . .	117
Exa 14.11	Designing a high pass filter and finding the frequency of peak attenuation	118

Exa 14.12	Designing a low pass composite filter	119
Exa 14.13	Designing a prototype band pass filter	120
Exa 14.14	Designing a prototype T section of a band pass filter	120
Exa 14.15	Designing a m derived low pass filter	121
Exa 14.16	Determining the values of shun arm of network	122
Exa 14.17	Designing a low pass composite filter	122
Exa 14.18	Determining the cut off frequency and char- acteristic impedance	123
Exa 14.19	Designing a low pass composite filter	124
Exa 14.20	Determining the second resonance frequency	125
Exa 14.21	Computing the values of elements of filter . .	125
Exa 14.22	Calculating bandwidth	126
Exa 14.23	Finding the values of the elements of a proto- type filter	126
Exa 15.4	Finding the elements of the lattice	128
Exa 15.5	Calculating values of L and C	128
Exa 15.6	Calculating characteristic impedance and the components of the shunt arm of equalizer . .	129
Exa 15.10	Calculating value of R and L	129
Exa 15.11	Designing a constant resistance equalizer . .	130
Exa 15.12	Designing a constant resistance equalizer . .	130
Exa 15.13	Designing a lattice equalizer	131
Exa 16.1	Designing a symmetrical bridge T network . .	133
Exa 16.2	Designing a T pad attenuator	133
Exa 16.3	Designing a T type attenuator	134
Exa 16.4	Designing an attenuator and determining its equivalent T structure	134
Exa 16.5	Finding the elements of the arm of a T section symmetrical resistive attenuator	135
Exa 16.6	Calculating the characteristic and attenuation per section of an attenuator	135
Exa 16.7	Determining the resistance value of a T type attenuator pad	136
Exa 16.8	Finding the elements of a balanced T pad . .	136
Exa 16.9	Designing L type attenuator	137
Exa 16.10	Designing a T pad	137
Exa 16.11	Designing an unbalances pi attenuator	138

Exa 16.12	Designing a T attenuator	138
Exa 16.13	Designing a ladder attenuator	139
Exa 16.14	Designing an attenuator	139
Exa 16.15	Designing a balanced attenuator	140
Exa 16.16	Finding the attenuation and the values of the shunt resistance of an H type attenuator . . .	140
Exa 16.17	Designing a L type attenuator	141
Exa 16.18	Designing a minimum loss pad	141

Chapter 1

Transmission Lines

Scilab code Exa 1.1 Calculating the resistance and inductance and leakage and capa

```
1 clear;
2 clc;
3 Zo=710* exp(%i*(-%pi/(180/16)));f=1000;
4 w=2*%pi*f;
5 a=.01;b=.035;
6 P=a+%i*b;
7 Z=Zo*P;
8 R=real(Z);
9 r=round(R*100)/100;
10 printf('—Resistance R = %f ohms/km\n',r);
11 L=((imag(Z))/w)*10^3;
12 l=round(L*100)/100;
13 printf('—Inductance L = %f mH/km\n',l);
14 Y=P/Zo;
15 G=real(Y);
16 printf('—Conductance G = %f mhos/km\n',-G);
17 C=((imag(Y))/w)*10^6;
18 c=round(C*1000)/1000;
19 printf('—Capacitance C = %f microfarads/km\n',c);
20 Vp=round(w*1000/(b*10^5))/1000;
21 printf('—Velocity of propagation Vp = %f^* 10^5 km/
```

```
sec',Vp);
```

Scilab code Exa 1.2 Calculating the values of the line constants

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

Scilab code Exa 1.3 Determining Zo and a and b

```
1 clear;
2 clc;
3 R=10;L=.0037;f=1000;G=.4*(10^-6);C=.0083*(10^-6);
4 w=2*%pi*f;
5 Z=R+(%i*w*L);
6 Y=G+(%i*w*C);
7 Zo=sqrt(Z/Y);
8 C=round(real(Zo));
9 D=round(imag(Zo));
10 printf('-Zo = %f + j(%f) ohms\n',C,D);
```



```

11 P=sqrt(Z*Y);
12 a=real(P);
13 a1=round(a*10000)/10000;
14 printf('-Attenuation constant a = %f neper/km\n',a1)
    ;
15 b=imag(P);
16 b1=round(b*10000)/10000;
17 printf('-Phase constant b = %f radians/km',b1);

```

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

```

1 clear;
2 clc;
3 f=1000;R=6;L=.0022;G=.25*(10^-6);C=.005*(10^-6);l
    =100;
4 //value of C as taken in solution
5 w=2*pi*f;
6 Z=R+(%i*w*L);
7 Y=G+(%i*w*C);
8 Zo=sqrt(Z/Y);
9 C=real(Zo);
10 D=imag(Zo);
11 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);
12 P=sqrt(Z*Y);
13 a=real(P);
14 b=imag(P);
15 a1=a*1*8.66;
16 printf('(ii)(a)Attenuation suffered while travelling
    = %f db\n',a1);
17 Vp=(w/b)*(10^-5);v=round(Vp*100)/100;
18 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 pr

```
1 clear;
2 clc;
3 f=1000;R=10.4;L=.0037;G=.8*(10^-6);C=.00835*(10^-6);
   l=1000;Vs=1;
4 w=2*pi*f;
5 Z=R+round((%i*w*L));
6 Y=G+(%i*w*C);
7 Zo=sqrt(Z/Y);
8 printf('Characteristic impedance is Zo = %f /_%f
   ohms\n',fix(abs(Zo)),round(((atan(imag(Zo),real(
   Zo))*180/%pi))));
9 P=sqrt(Z*Y);
10 printf('Propagation constant P = %f + j(%f)\n',fix(
   real(P)*10^4)/10^4,round(imag(P)*10^4)/10^4);
11 b=imag(P);
12
13 Is=Vs/Zo;Vp=(w/b)*(10^-5);v=round(Vp*100)/100;
14 printf('Phase velocity Vp = %f * 10^5 km/sec\n',v);
15 Ir=Is*exp(-P*l);
16 P=((abs(Ir))^2)*real(Zo);
17 printf("Power delivered at receiving end = %f micro
   -watt",P*(10^6));
```

Scilab code Exa 1.6 Calculating characteristic impedance and propagation constant

```
1 clear;
2 clc;
3 f=5000/(2*pi);R=196;C=.09*(10^-6);L=.71*(10^-3);G
   =0;
```

```

4 //value of C as taken in solution
5 w=2*%pi*f;
6 Z=R+(%i*w*L);
7 Y=G+(%i*w*C);
8 Zo=sqrt(Z/Y);
9 printf('-Characteristic impedance is Zo = %f /_%f
      ohms\n',fix(abs(Zo)),round(((atan(imag(Zo),real(
      Zo))*180/%pi))*10)/10);
10 P=sqrt(Z*Y);F=fix(abs(P)*100)/100;
11 printf("-Propagation constant P = %f /_%f ohms\n',F
      ,((atan(imag(P),real(P))*180/%pi));
12 //the difference in result is due to erroneous
      value in textbook.
13 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

```

1 clc;
2 R=10.4;L=3.666*(10^-3);G=.08*(10^-6);C
      =.00835*(10^-6);w=5000;
3 //value of L and C as taken in solution
4 Z=R+(%i*w*L);
5 Y=G+%i*w*C;
6 Zo=sqrt(Z/Y);
7 printf('-Zo = %f /_%f ohms\n',fix(abs(Zo)),round(((
      atan(imag(Zo),real(Zo))*180/%pi))*10)/10);
8 P=sqrt(Z*Y);
9 a=real(P);
10 b=imag(P);
11 printf('-Attenuation constant a = %f neper/km\n',a);
12 printf("-Phase constant b = %f radians/km\n",b);
13 Vp=(w/b)*(10^-5);v=round(Vp*1000)/1000;
14 printf('(ii)(b)Phase velocity Vp = %f * 10^5 km/sec\
      n',v);

```

```

15 //the difference in result of Zo is due to erroneous
    value in textbook.
16 disp("The difference in result of Zo is due to
    erroneous value in textbook")

```

Scilab code Exa 1.8 Calculating characteristic impedance

```

1 clear;
2 clc;
3 R=65;L=1.6*(10^-3);C=.1*(10^-6);G=2.25*(10^-6);f
    =800;
4 w=5000;
5 Z=R+round(%i*w*L);
6 Y=G+%i*w*C;
7 Zo=sqrt(Z/Y);
8 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

```

1 clear;
2 clc;
3 R=6;L=2*(10^-3);G=.5*(10^-6);C=.005*(10^-6);f=1000;l
    =100;
4 w=2*%pi*f;
5 Z=R+(%i*w*L);
6 Y=G+%i*w*C;
7 Zo=sqrt(Z/Y);
8 P=sqrt(Z*Y);
9 a=real(P);
10 b=imag(P);
11 a1=fix(a*8.66*1*10^3)/10^3;

```

```

12 printf("-Attenuation suffered while travelling = %f
    db\n",a1);
13 Vp=fix(w/b)/10^3;
14 printf("-Phase velocity Vp = %f km/sec",Vp );

```

Scilab code Exa 1.10 Calculating characteristic impedance

```

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

```

1 clear;
2 clc;
3 x=1;l=12;V=0.1;
4 V1=1-V;
5 //V1=ratio of voltage at 1km from the sending end to
    the voltage at sending end
6 P=-log(V1);

```

```

7 V2=exp(-P*1);
8 x=V2*100;x1=round(x*100)/100;
9 //x=ratio of voltage across the load impedance to
  the voltage at sending end
10 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

```

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

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

Scilab code Exa 1.14 Calculating line attenuation and velocity of propagation

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

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

```
1 clear;
2 clc;
3 l=200;Vg=10;Zg=500;Zs=683-(%i*138);P=0.0074+(%i
   *0.0356);Zo=Zs;
4 Is=Vg/(Zg+Zs);
5 modIs=abs(Is);
6 printf("-Sending end current = %f mA\n",round(modIs
   *(10^3)*100)/100);
7 Vs=Is*Zs;
8 modVs=abs(Vs);
9 printf("-Sending end voltage = %f V r.m.s.\n",round(
   modVs*100)/100);
```



```
10 Rs=real(Zs);
11 Ps=((modIs)^2)*Rs;
12 printf("–Sending end power = %f mW\n",round(Ps
    *(10^3)*100)/100);
13 Vr=modVs*exp(-P*1);
14 A=imag(-P*1);
15 printf("–Receiving end voltage = %f /_ %f V\n",round
    (abs(Vr)*100)/100,A);
16 Zr=Zs;
17 Ir=Vr/Zr;
18 modIr=abs(Ir);
19 printf("–Receiving end current = %f mA\n",round(
    modIr*(10^3)*100)/100);
20 Rr=Rs;
21 Pr=((modIr)^2)*Rr;
22 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 Z_o

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

Scilab code Exa 2.2 Calculating the line constants

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

```

7  printf("-Zo = %f / - %f ohms\n", fix(abs(Zo)), atan(D,C
      )*180/%pi);
8  w=2*%pi*f;
9  Z1=sqrt(Zsc/Zoc);
10 A=real(Z1);
11 B=imag(Z1);
12 D=(1+A+(%i*B))/(1-(A+(%i*B)));
13 r=abs(D);
14 theta=atan((imag(D))/(real(D)))-%pi;
15 n=1;
16 P=(1/(2*1))*((log(r))+(%i*(theta+(2*n*%pi))));
17 E=real(P);
18 F=imag(P);
19 printf("-P = %f / - %f\n", round(abs(P)*10000)/10000,
      round(atan(F,E)*180*100/%pi)/100);
20 Z=P*Zo;
21 R=real(Z);
22 L=(imag(Z))/w;
23 printf("-R = %f ohms/km\n", round(R*100)/100);
24 printf("-L = %f mH/km\n", round(L*(10^3)*100)/100);
25 Y=P/Zo;
26 G=real(Y);
27 C=(imag(Y))/w;
28 printf("-G = %f micro-mhos/km\n", round(G*(10^6)));
29 printf("-C = %f microfarads/km", round(C*(10^6)
      *10000)/10000);

```

Scilab code Exa 2.3 Calculating the received current and voltage

```

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

```

```

6 P=a+(%i*b);
7 Ir=Is/(cosh(10*P));
8 A=real(Ir);
9 B=imag(Ir);
10 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 Z_0 and a and b and primary constants of line

```

1 clear;
2 clc;
3 Zoc=1930*exp(%i*(%pi/(180/68.9)));Zsc=1308*exp(%i*(-
        %pi/(180/76.2)));l=8;w=5000;
4 Zo=sqrt(Zoc*Zsc);
5 C=real(Zo);
6 D=imag(Zo);
7 printf("-Zo = %f /_ %f ohms\n",round(abs(Zo)),round(
        atan(D,C)*180*100/%pi)/100);
8 Z1=sqrt(Zsc/Zoc);
9 A=round(real(Z1)*1000)/1000;
10 B=round(imag(Z1)*1000)/1000;
11 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;
12 phi1=atan(imag(K),real(K))+(%pi/4);
13 phi2=atan(imag(J),real(J));
14 phi3=phi1-phi2;
15 P=(1/(2*8))*(log(round(abs(K/J)*10)/10)+(%i*(round(
        phi3*10)/10+(6.28)))));
16 printf(" -a = %f neper/km\n",fix(real(P)*100)/100);
17 printf(" -b = %f radians/km\n",fix(imag(P)*1000)
        /1000);
18 Z=P*Zo;
19 R=real(Z);
20 L=(imag(Z))/w;

```

```

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

```

Scilab code Exa 2.5 Calculating the line constants

```

1 clear;
2 clc;
3 w=5000; Zoc=1300*exp(%i*(%pi/(180/80))); Zsc=3200*exp(
        %i*(-%pi/(180/80))); l=40;
4 Zo=sqrt(Zoc*Zsc);
5 Z1=sqrt(Zsc/Zoc);
6 A=real(Z1);
7 B=imag(Z1);
8 D=(1+A+(%i*B))/(1-(A+(%i*B))); K=(1+A+(%i*B)); J=(1-(A
        +( %i*B)));
9 r=round(abs(D)*1000)/1000;
10 theta=round(atan((imag(D))/(real(D)))*10)/10;
11 P=(1/(2*l))*((log(r))+(%i*(theta+6.28)));
12 b=imag(P);
13 Z=P*Zo;
14 R=real(Z);
15 L=(imag(Z))/w;
16 printf("R = %f ohms\n", fix(R*100)/100);
17 printf(" L = %f mH/km\n", round(L*(10^6))/1000);

```

```

18 Y=P/Zo;
19 G=real(Y);
20 C=(imag(Y))/w;
21 printf(" G = %f micro-mhos/km\n",round(G*(10^8))
        /100);
22 printf(" C = %f micro-farads/km",round(C*(10^9))
        /1000);
23 //the difference in result of L and C is due to
    erroneous value in textbook.
24 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

```

1 clear;
2 clc;
3 Zoc=286*exp(%i*(-%pi/(180/40)));Zsc=1520*exp(%i*(%pi
    /(180/16)));l=50;f=700;
4 w=2*%pi*f;
5 Zo=sqrt(Zoc*Zsc);
6 S=real(Zo);
7 T=imag(Zo);
8 printf("-Characteristic impedance of transmission
    line is Zo = %f /- %f ohms\n",round(abs(Zo)*100)
    /100,atan(T,S)*180/%pi);
9 A=atanh(sqrt(Zsc/Zoc));
10 P=A/l;
11 U=real(P);
12 V=imag(P);
13 printf("-Propagation constant = %f /- %f\n",fix(abs(
    P)*1000)/1000,round(atan(V,U)*180*100/%pi)/100)
14 Z=P*Zo;
15 R=real(Z);
16 L=(imag(Z))/w;
17 printf("-R = %f ohms/km\n",round(R*100)/100);

```

```

18 printf("-L = %f mH/km\n", fix(L*(10^3)*100)/100);
19 Y=P/Zo;
20 G=real(Y);
21 C=(imag(Y))/w;
22 printf("-G = %f micromhos/km\n", fix(G*(10^6)*100)
        /100);
23 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 o

```

1 clear;
2 clc;
3 Zoc=2500*exp(%i*(-%pi/(180/70)));Zsc=49*exp(%i*(%pi
        /(180/25)));l=1;
4 f=(8000)/(2*%pi);
5 w=2*%pi*f;
6 Zo=sqrt(Zoc*Zsc);
7 A=atanh(sqrt(Zsc/Zoc));
8 P=A/l;
9 P1=P*1/1000;
10 Z=P1*Zo;
11 printf("Series impedance of line is Z = %f /- %f
        ohms/km\n", round(abs(Z)*1000)/1000, atan(imag(Z),
        real(Z))*180/%pi);
12 Y=P1/Zo;
13 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

```

1 clear;

```

```

2  clc;
3  Zoc=2000*exp(%i*(-%pi/(180/80)));Zsc=20*exp(%i*(%pi
    /(180/20)));l=0.5;w=10000;
4  //value of length of cable as taken in solution
5  Zo=sqrt(Zoc*Zsc);
6  C=real(Zo);
7  D=imag(Zo);
8  printf("-Zo = %f / - %f ohms\n",abs(Zo),atan(D,C)
    *180/%pi);
9  A=atanh(sqrt(Zsc/Zoc));
10 P=A/l;
11 a=real(P);
12 printf("-a = %f neper/km\n",fix(a*10000)/10000);
13 b=imag(P);
14 printf("-b = %f henry/km",round(b*10000)/10000);

```

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

```

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

```

Scilab code Exa 2.10 Calculating Zo and a and b

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

Scilab code Exa 2.11 Calculating Zo and a and b

```
1 clear;
2 clc;
3 f=20*(10^6);Zoc=1390;Zsc=4.61;l=5;
4 lo=(3*(10^8))/f; //lo=wavelength
5 Zo=sqrt(Zoc*Zsc);
6 printf("-Characteristic impedance = %f ohms\n",round
    (Zo));
7 er=(lo/l)^2; //er=relative permittivity of
```

```

    dielectric
8  printf("-Relative permittivity of dielectric = %f\n"
    ,er);
9  P=fix(((atanh(sqrt(Zsc/Zoc)))/1)*10000)/10000;
10 a=P*8.686;
11 printf("-a = %f db/m\n",fix(a*10000)/10000);
12 Vp=(3*(10^8))/(sqrt(er)*10^8);
13 printf("-Velocity of propagation = %f * 10^8 m/sec",
    Vp);
14 //the difference in result of attenuation constant
    is due to erroneous value in textbook.
15 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

```
1 clear;
2 clc;
3 l=10;a=0.03;b=0.03;Vr=4;f=1000;
4 P=a+(%i*b);
5 Vs=Vr*(exp(P*l));
6 C=real(Vs);
7 D=imag(Vs);
8 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

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

```

7 C=real(I);
8 D=imag(I);
9 printf("R.m.s. voltage at the required distance is V
    = %f volts\n", round((V)*100)/100);
10 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

```

1 clc;
2 Zo=500*exp(%i*(-%pi/(180/43)));l=10;P=0.07+(%i*0.08)
    ;Vs=5;
3 V=Vs*(exp(-P*l));
4 I=V/Zo;
5 C=real(V);
6 D=imag(V);
7 E=real(I)*(10^3);
8 F=imag(I)*(10^3);
9 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);
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

```

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

```

```

4 K=round(sinh(P*1)*100)/100; J=round(cosh(P*1)*100)
    /100;
5 Zin=Zo*((Zr*J)+(Zo*K))/((Zo*J)+(Zr*K));
6 C=real(Zin);
7 D=imag(Zin);
8 printf("Input impedance of line is Zin = %f /- %f
    ohms", abs(Zin), round(atan(D,C)*180*10/%pi)/10);
9 A=((Zr*J)+(Zo*K));
10 B=((Zo*J)+(Zr*K));
11 //the difference in result is due to erroneous
    value in textbook.
12 disp("The difference in result is due to erroneous
    value in textbook")

```

Scilab code Exa 3.5 Calculating sending end impedance

```

1 clear;
2 clc;
3 Zo=55; Zr=115+(%i*75); n=1.183;
4 //value of Zr as taken in solution
5 m=(2*%pi*n)-(2*%pi);
6 Zin=Zo*(Zr+(%i*Zo*tan(m)))/(Zo+(%i*Zr*tan(m)));
7 C=real(Zin);
8 D=imag(Zin);
9 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

```

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

```

```

4 l=100;x=4; //Vr=voltage reflection coefficient
5 modVr=abs(Vr);
6 A=modVr*exp(-2*a*(1-x));
7 Zin=Zo*(1+A)/(1-A);
8 C=real(Zin);
9 D=imag(Zin);
10 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

```

1 clear;
2 clc;
3 Ka=0.2*exp(%i*(-%pi/(180/30)));d=1/12;
4 Kb=Ka/exp(%i*%pi*4*d);
5 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

```

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

```

```

9 printf("–Ratio of the transmitter voltage and the
    receiver voltage in nepers = %f\n",round((V2)
    *100)/100);
10 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

```

1 clear;
2 clc;
3 Zo=692*exp(%i*(-%pi/(180/12 )));Zr=200;l=100;V=1;f
    =1000;
4 K=(Zr-Zo)/(Zr+Zo);
5 C=real(K);
6 D=imag(K);
7 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

```

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

```

```

11 Ir=Is*exp(-P*1);
12 P=((abs(Ir))^2)*real(Zo);
13 printf("Power delivered at receiving end = %f micro
    -watt",P*(10^6));
14 //the difference in result is due to erroneous
    value in textbook.
15 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

```

1 clear;
2 clc;
3 l=50;Zo=692*exp(%i*(-%pi/(180/12)));a=0.00755;b
    =0.0355;Zr=300;f=1000;Vs=10;
4 P=a+(%i*b);
5 Zin=Zo*((Zr*cosh(P*1))+(Zo*sinh(P*1)))/((Zo*cosh(P*1)
    )+(Zr*sinh(P*1)));
6 Is=Vs/Zin*(10)^-1;
7 A=real(Is);
8 B=imag(Is);
9 printf("Sending end current = %f /_ %f Amp\n",round
    (abs(Is)*10^5)/10^5,round(atan(B,A)*180*10/%pi)
    /10);
10 Vr=Vs*((cosh(P*1))-((Zo*sinh(P*1))/Zin));
11 C=real(Vr);
12 D=imag(Vr);
13 printf("Receiving end voltage = %f /_ %f volts\n",
    round(abs(Vr)*10)/10,round(atan(D,C)*180/%pi));
14 Ir=Vr/Zr;
15 E=real(Ir);
16 F=imag(Ir);
17 printf("Receiving end current = %f /_ %f Amp\n",
    round(abs(Ir)*1000)/1000,round(atan(F,E)*180/%pi)
    );

```



```

18 modIr=abs(Ir);
19 Rr=Zr;
20 Pr=((modIr)^2)*Rr;
21 theta=atan((imag(Zin))/(real(Zin)));
22 modIs=abs(Is);
23 Ps=Vs*modIs*cos(theta);
24 n=(Pr/Ps)*10;
25 printf("–Efficiency of transmission(in percentage)=
    %f",round(n));

```

Scilab code Exa 3.12 Calculating the resistance and inductance of the series and s

```

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

```

Scilab code Exa 3.13 Determining the characteristic impedance and propagation cons

```

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

```

Scilab code Exa 3.14 Calculating voltage and current

```

1 clear;
2 clc;
3 a=0;Zo=50;Vr=100;l=50;y=10;Ir=0;f=20*(10^6);
4 lo=300/f; //lo=lambda
5 b=2*pi/lo;
6 P=a+(i*b);
7 V=(Vr*cosh(P*y))+(Ir*Zo*sinh(P*y));
8 printf("-Voltage at the required point = %f /- %f
    volts\n",abs(V),atan(imag(V),real(V))*180/pi);

```

```

9 I=((Vr/Zo)*sinh(P*y))+(Ir*cosh(P*y));
10 A=real(I);
11 B=imag(I);
12 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

```

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

```

Scilab code Exa 3.16 Calculating power consumed in load

```

1 clear;
2 clc;
3 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;
4 w=2*%pi*f;
5 Z=R+(%i*w*L);
6 Y=G+(%i*w*C);
7 Zo=sqrt(Z/Y);
8 P=sqrt(Z*Y);
9 Is=Vs/Zo;
10 Ir=Is*(exp(-P*l));
11 Ir1=abs(Ir);
12 Rr=real(Zo);

```

```

13 Pr=Ir1*Ir1*Rr;
14 printf("Power consumed in load is Pr = %f mW",round(
    Pr*(10^3)*10)/10);
15 //the difference in result is due to erroneous
    value in textbook.
16 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

```

1 clear;
2 clc;
3 R=10.4;L=0.0036;C=0.0083*(10^-6);G=0;f1=1420;f2
    =1860;
4 Favg=(f1+f2)/2;
5 w=2*%pi*Favg;
6 Z=R+(%i*w*L);
7 Y=G+(%i*w*C);
8 P=sqrt(Z*Y);
9 a=real(P);
10 b=imag(P);
11 v=w/b;
12 d=v/(2*(f2-f1));
13 printf("-Approximate distance = %f km\n",round(d));
14 w1=2*%pi*f1;
15 Z1=R+(%i*w1*L);
16 Y1=G+(%i*w1*C);
17 P1=sqrt(Z1*Y1);
18 a1=real(P1);
19 b1=imag(P1);
20 v1=w1/b1;
21 w2=2*%pi*f2;
22 Z2=R+(%i*w2*L);
23 Y2=G+(%i*w2*C);
24 P2=sqrt(Z2*Y2);

```

```

25 a2=real(P2);
26 b2=imag(P2);
27 v2=w2/b2;
28 D=(v1*v2)/(2*((v1*f2)-(v2*f1)));
29 printf("Exact distance = %f km",round(D*10)/10);

```

Scilab code Exa 3.18 Calculating the reflection factor and reflection loss

```

1 clear;
2 clc;
3 Z1=300;Zo=650*exp(%i*(-%pi/(180/12)));Z2=Zo;
4 Z=Z1+Z2;
5 K=(2*sqrt(Z1*abs(Z2)))/abs(Z);
6 printf("Reflection factor = %f\n",round(K*1000)
    /1000);
7 L=20*(log10(1/K));
8 printf("Reflection loss = %f db",round(L*100)/100);
9 //the difference in result of reflection loss is due
    to erroneous value in textbook.
10 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

```

1 clear;
2 clc;
3 Zo=600;a=0.01;b=0.03;l=100;Zr=200;
4 P=a+(%i*b);
5 K=(Zr-Zo)/(Zr+Zo);
6 printf("Reflection coefficient = %f /_- %f\n",abs(K)
    ,atan(imag(K),real(K))*180/%pi);
7 k=((abs(K))*exp(-2*P*l));
8 b=(1+k)/(1-k);

```

```
9 Zin=Zo*b;
10 C=real(Zin);
11 D=imag(Zin);
12 printf("Input impedance Zin = %f + j(%f) ohms",round
    (C*1000)/1000,round(D*100)/100);
13 //Zin has been wrongly calculated in the book
```

Scilab code Exa 3.20 Finding reflection loss

```
1 clc;
2 Zo=632.8;Zr=80;
3 F=20*log10((2*Zr*Zo)/(Zo+Zr));
4 printf("Reflection loss = %f db",round(F*100)/100);
5 //the difference in result is due to erroneous
    value in textbook.
6 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

```
1 clear;
2 clc
3 l=225; Zo=401-(%i*29); P=(0.148+(%i*2.06))*(10^-3);
4 Zs=Zo*sinh(P*l);
5 A=real(Zs);
6 B=imag(Zs);
7 printf("-Series branch of the equivalent network
      will have the impedance = %f + j(%f) ohms\n",
      round(A), round(B));
8 Zsh=Zo*coth(P*l/2);
9 C=real(Zsh);
10 D=imag(Zsh);
11 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

```
1 clear;
```

```

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

```

Scilab code Exa 4.4 Calculating characteristic impedance and propagation constant

```

1  clear;
2  clc;
3  l=440;L=2.2*(10^-3);C=0.0136*(10^-6);R=0.120;G=0;f
    =60;
4  w=2*%pi*f;
5  Z=R+(%i*w*L);
6  Y=G+(%i*w*C);
7  Zo=sqrt(Z/Y);
8  A=real(Zo);
9  B=imag(Zo);
10 printf("-Characteristic impedance = %f + j(%f) ohms\
    n",round(A),round(B));
11 P=sqrt(Z*Y);
12 E=real(P)*10^4;
13 F=imag(P)*10^3;
14 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

```

1  clear;
2  clc;

```



```

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

```

Scilab code Exa 4.6 Calculating capacitive susceptance

```

1 clear;
2 clc;
3 dia=2*(10^-2);f=50;d=3.5;
4 r=dia/2;
5 Cab=0.01207/(log10(d/r))*(10^-6);
6 Xc=1/(2*pi*f*Cab);
7 Bc=(1/Xc)*10^6;
8 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

```

1 clear;
2 clc;

```

```

3 dia=2*(10^-2);f=50;d1=3.5;V=66000;
4 r=dia/2;
5 Cn1=0.02414/(log10(d1/r))*(10^-6);
6 printf("(a) Capacitance per km = %f microfarads/km, to
   neutral\n",fix(Cn1*(10^6)*10^5)/10^5);
7 Vn=V/sqrt(3);
8 Ic1=Vn*2*pi*f*Cn1;
9 printf("\tCharging current = %f mA/km\n",fix(Ic1
   *(10^3)*10)/10);
10 d1=3.5;d2=4.5;d3=5.5;
11 Deq=nthroot(d1*d2*d3,3);
12 Cn2=0.02414/(log10(Deq/r))*(10^-6);
13 printf("(b) Capacitance per km =%f microfarads/km, to
   neutral\n",fix(Cn2*(10^6)*10^5)/10^5);
14 Ic2=Vn*2*pi*f*Cn2;
15 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

```
1 clear;
2 clc;
3 R=28;C=0.04*(10^-6);G=0;L=0;f=1600;s=2000;r=3.7;l
   =88*(10^-3);
4 Rc=R+(r/(s*(10^-3)));
5 Lc=L+(l/(s*(10^-3)));
6 Fc=1/(%pi*(sqrt(Lc*C)));
7 printf("Cutoff frequency = %f Hz\n",round(Fc));
8 w=2*%pi*f;
9 a=sqrt(w*C*R/2);
10 a1=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
11 A=(a-a1)*8.686;
12 printf("Decrease in attenuation = %f db/km",round(A
   *100)/100);
```

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

```
1 clear;
```

```

2  clc;
3  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);
4  s=7.88;
5  Rc=R+(r/s);
6  Lc=L+(l/s);
7  al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
8  printf("-a = %f neper/km\n",round(al*10000)/10000);
9  b=w*(sqrt(Lc*C));
10 printf("-b = %f radians/km\n",round(b*10^4)/10^4);
11 lo=2*pi/b; //lo=lambda
12 printf("-lo = %f km\n",round(lo*100)/100);
13 Vp=(w/b)*10^-4;
14 printf("-Vp = %f * 10^4 km/sec\n",round(Vp*100)/100)
    ;
15 Zo=(sqrt(Lc/C))*10^-3;
16 printf("-Zo = %f * 10^3 ohms",fix(Zo*100)/100);

```

Scilab code Exa 5.3 Calculating modified Zo and a and Vp

```

1  clear;
2  clc;
3  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);
4  s=7.88;
5  Rc=R+(r/s);
6  Lc=L+(l/s);
7  Z=fix((Rc+(%i*w*Lc))*100)/100;
8  Y=G+(%i*w*C);
9  Zo=sqrt(Z/Y);
10 A=real(Zo);
11 B=imag(Zo);
12 P=sqrt(Z*Y);
13 a=real(P);
14 b=imag(P);

```



```

12 Vp1=(1/(sqrt(Lc*C)))*10^-4;
13 printf("-Phase velocity after loading = %f * 10^4 km
    /sec\n",Vp1);
14 A=a-al;
15 printf("-Decrease in attenuation = %f neper/km\n",
    round(A*10000)/10000);
16 Fc=1/(%pi*(sqrt(Lc*C*d)));
17 printf("-Cutoff frequency = %f kHz",round(Fc*(10^-3)
    *1000)/1000);

```

Scilab code Exa 5.5 Calculating attenuation and cut off frequency

```

1 clear;
2 clc;
3 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;
4 Rc=R+(r/d);
5 Lc=L+(l/d)
6 w=2*%pi*f;
7 Z=Rc+(%i*w*Lc);
8 Y=G+(%i*w*C);
9 P=sqrt(Z*Y);
10 a=real(P);
11 printf("(a) Attenuation constant a = %f neper/km\n",
    round(a*1000)/1000);
12 Fc=1/(%pi*(sqrt(l*C*d)));
13 printf("(b) Cutoff frequency = %f kHz",round(Fc
    *(10^-3)*1000)/1000);

```

Scilab code Exa 5.6 Calculating attenuation and highest frequency of transmission

```

1 clear;
2 clc;

```

```

3 R=25;C=0.04*(10^-6);L=0.6*(10^-3);G=0;l=45*(10^-3);r
    =20;d=1;f=1000;
4 Rc=R+(r/d);
5 Lc=L+(l/d)
6 w=2*pi*f;
7 Z=Rc+(i*w*Lc);
8 Y=G+(i*w*C);
9 P=sqrt(Z*Y);
10 a=real(P);
11 printf("-Attenuation per km line = %f neper/km\n",
    round(a*10^5)/10^5);
12 Fc=1/(pi*(sqrt(Lc*C*d)));
13 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 Z_o and a and V_p after loading

```

1 clear;
2 clc;
3 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);
4 d=1.135;
5 Rc=R+(r/d);
6 Lc=round((L+(l/d))*1000)/1000;
7 Zo=sqrt(Lc/C);
8 printf("-Zo after loading = %f ohms\n",round(Zo));
9 al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
10 printf("-a after loading = %f neper/km\n",round(al
    *10^4)/10^4);
11 Vp=1/(sqrt(Lc*C));
12 printf("-Approximate value of Vp = %f km/sec",fix(Vp
    ));

```

Scilab code Exa 5.8 Calculating cut off frequency and attenuation

```
1 clear;
2 clc;
3 R=30;L=0;G=0;C=7000*(10^-12);f=3/(2*%pi)*(10^3);r
   =35;l=70*(10^-3);
4 d=1;
5 Rc=R+(r/d);
6 Lc=L+(l/d);
7 Fc=1/(%pi*(sqrt(Lc*C*d)));
8 printf("Approximate value of highest frequency = %f
   kHz\n",round(Fc*(10^-3)*100)/100);
9 a1=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));
10 printf("Approximate value of attenuation per km
   after = %f neper/km",round(a1*10^4)/10^4);
```

Scilab code Exa 5.9 Calculating Zo and a and b

```
1 clear;
2 clc;
3 R=44;L=0.001;G=1*(10^-6);C=0.065*(10^-6);f=1600;r
   =3.7;l=88*(10^-3);
4 d=1.136;
5 w=2*%pi*f;
6 Rc=R+(r/d);
7 Lc=L+(l/d);
8 Zo=sqrt(Lc/C);
9 printf("Approximate value of Zo = %f ohms\n",round(
   Zo*100)/100);
10 a1=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C))*8.686;
11 printf("Approximate value of a = %f db/km\n",fix(a1
   *1000)/1000);
```



```

12 b=w*(sqrt(Lc*C));
13 printf("—Approximate value of b = %f radians/km",fix
    (b*1000)/1000);
14 //the difference in result of Zo is due to erroneous
    value in textbook.
15 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

```

1 clear;
2 clc;
3 a=0.005;f=1000;a1=0.05;att=10 //att=given
    attenuation in db
4 lo=att/(a*8.686);
5 printf("—For the open wire line the required length
    is lo = %f km\n",round(lo*10)/10);
6 lc=att/(a1*8.686);
7 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

```

1 clear;
2 clc;
3 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);
4 Rc=R+(r/d);
5 Lc=L+(l/d);
6 Fc=1/(%pi*(sqrt(Lc*C*d)));
7 printf("(i) Cutoff frequency = %f kHz\n",round(Fc
    *(10^-3)*100)/100);
8 al=((Rc/2)*sqrt(C/Lc))+((G/2)*sqrt(Lc/C));

```

```

9  printf("(ii) Attenuation per km = %f neper/km\n", fix(
    al*10000)/10000);
10 Vp1=(1/(sqrt(Lc*C)))*10^-4;
11 printf("(iii) Phase velocity = %f * 10^4 km/sec", fix(
    Vp1*1000)/1000);
12 //the difference in result of attenuation per km and
    phase velocity is due to erroneous value in
    textbook.
13 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 inte

```

1  clear;
2  clc;
3  R=80;C=0.06*(10^-6);L=0;G=0;f=2500;al=0.04;ratio=40;
4  fc=2*f;
5  //ratio=ratio off resistance to inductance of
    loading coil
6  A=2/(3.14*3.14*C*4*f*f); //A=Lc*d
7  B=al/(%pi*f*1.414*C/2);
8  C=%pi*f*1.414*C/2;
9  y=poly([ratio*C*100*A -al*100 80*100*C],"x","coeff")
    ;
10 a=roots(y);
11 b=round(a(1,1)*100)/100;
12 Lo=round((A/b)*1000)/1000;
13 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

```

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

```

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

```

1 clear;
2 clc;
3 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
4 w=2*pi*f;
5 Z=round(R+(%i*w*L));
6 Y=G+(%i*w*C);
7 Zo=sqrt(Z/Y);
8 P=round(sqrt(Z*Y)*10^4)/10^4;
9 Zc=r+(%i*w*l);
10 A=fix(((cosh(P*d))+(Zc*(sinh(P*d))/(2*Zo)))*10^3)
    /10^3;
11 P1=(acosh(A))/d;
12 a1=real(P1);
13 b1=imag(P1);
14 printf("New value of attenuation constant = %f neper
    /km\n",round(a1*100)/100);
15 printf(" New value of phase constant = %f rad/km\n",
    round(b1*1000)/1000);
16 Lc=L+(1/(d*10^3));
17 Fc=1/(3.14*(sqrt(Lc*C*d)));

```

```

18 printf(" Cutoff frequency = %f * 10^4 Hz",round(Fc
    *10^-2)/10^2);
19 //the difference in result is due to erroneous
    value in textbook.
20 disp("The difference in result is due to erroneous
    value in textbook")

```

Scilab code Exa 5.15 Calculating wavelength of line

```

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

```

Scilab code Exa 5.16 Calculating the value of attenuation

```

1 clear;
2 clc;
3 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;
4 pf=0.005//pf=power factor //value of pf as taken in
    solution
5 r=pf*w*L;
6 Rc=R+(r/s);
7 Lc=L+(l/s);

```

```

8 P=sqrt((Rc+(%i*w*Lc))*(G+(%i*w*C)));
9 theta=round(atan(imag(P),real(P))*180/%pi);
10 a=abs(P)*cos(theta*%pi/180);
11 printf("Attenuation constant of line = %f neper/km",
        fix(a*10^4)/10^4);

```

Scilab code Exa 5.17 Calculating potential at mid point

```

1 clear;
2 clc;
3 l=100;R=10;G=10^-5;Vs=40;Zr=0;
4 Zo=sqrt(R/G);
5 P=sqrt(R*G);
6 Zin=Zo*(Zr+(Zo*tanh(P*l)))/(Zo+(Zr*tanh(P*l)));
7 Is=Vs/Zin;
8 V=(Vs*(cosh(P*l)))-(Is*Zo*(sinh(P*l)));
9 Vm=2*V;
10 printf("Potential at mid point = %f volts",Vm);
11 //the difference in result is due to erroneous
    value in textbook.
12 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

```
1 clear;
2 clc;
3 f=200*(10^6);s=4.48;ymin=6;Zo=300; //s=standing wave
    ratio
4 lo=300/(f*(10^-6)); //where f is in megahertz ,lo=
    wavelength of wave in air
5 b=2*%pi/lo;
6 phi=(2*b*ymin*(10^-2))-%pi;
7 ampK=(s-1)/(s+1); //ampK=amplitude of the reflection
    coefficient
8 K=ampK*(exp(%i*phi));
9 A=real(K);
10 B=imag(K);
11 printf("Complex reflection coefficient= %f /_ %f\n"
    ,round(abs(K)*1000)/1000,round(atan(B,A)*180/%pi)
    );
12 ZR=(Zo*(1+K))/(1-K);
13 C=real(ZR);
14 D=imag(ZR);
15 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

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

```
1 clear;
2 clc;
3 Zo=50;f=300*(10^6);ZR=50+(%i*50);
4 lo=300/(f*(10^-6)); //where f is in megahertz ,lo=
           wavelength of wave in air
```

```

5 K=(ZR-Zo)/(ZR+Zo);
6 ampK=sqrt((real(K)^2)+(imag(K)^2));
7 S=(1+ampK)/(1-ampK);
8 printf("-VSWR = %f\n",round(S*100)/100);
9 phi=atan(imag(K)/real(K));
10 ymax=phi*lo/(2*2*%pi);
11 ymin=ymax+(lo/4);
12 printf("-Position of voltage minimum nearest load =
    %f metres",round(ymin*10000)/10000);

```

Scilab code Exa 6.4 Calculating standing wave ratio

```

1 clear;
2 clc;
3 Zo=400; ZRa=70, ZRb=800; ZRc=650-(%i*475);
4 Ka=(ZRa-Zo)/(ZRa+Zo);
5 ampKa=sqrt((real(Ka)^2)+(imag(Ka)^2));
6 Sa=(1+ampKa)/(1-ampKa);
7 printf("(a) Standing wave ratio = %f\n",round(Sa*100)
    /100);
8 Kb=(ZRb-Zo)/(ZRb+Zo);
9 ampKb=sqrt((real(Kb)^2)+(imag(Kb)^2));
10 Sb=(1+ampKb)/(1-ampKb);
11 printf("(b) Standing wave ratio = %f\n",Sb);
12 Kc=(ZRc-Zo)/(ZRc+Zo);
13 ampKc=sqrt((real(Kc)^2)+(imag(Kc)^2));
14 Sc=(1+ampKc)/(1-ampKc);
15 printf("(c) Standing wave ratio = %f",round(Sc*1000)
    /1000);

```

Scilab code Exa 6.5 Calculating value of load impedance

```

1 clear;

```



```

2  clc;
3  s=2;f=300*(10^6);lo=1;ymin=0.8; //lo=wavelength
4  ampK=(s-1)/(s+1);
5  b=2*%pi/lo;
6  phi=(2*b*ymin)-%pi;
7  K=ampK*(exp(%i*phi));
8  Zr=(1+K)/(1-K);
9  A=real(Zr);
10 B=imag(Zr);
11 printf("Value of load impedance = %f /_ %f ohms",
        round(abs(Zr)*10)/10,fix(atan(B,A)*180*100/%pi)
        /100);
12 //the difference in result is due to erroneous
    value in textbook.
13 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

```

1  clear;
2  clc;
3  ZR=100;Zo=600;f=100*(10^6);
4  lo=300/(f*(10^-6)); //lo=wavelength
5  Ls=(lo/(2*%pi))*(atan(sqrt(ZR/Zo)));
6  printf("-Point of attachment = %f cms\n",round(Ls
        *(10^2)*10)/10)
7  Lt=(lo/(2*%pi))*(%pi+(atan((sqrt(ZR*Zo))/(ZR-Zo))));
8  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

```

1  clear;

```

```

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

```

1  clear;
2  clc;
3  Zo=400;ZR=200-(%i*100);lo=3; //lo=wavelength
4  //value os Zo as taken in solution
5  K=(ZR-Zo)/(ZR+Zo);
6  ampK=abs(K);
7  phi=%pi + atan(imag(K)/real(K));
8  Ls=(lo/(4*%pi))*(phi+%pi-acos(ampK));
9  printf("Shortest distance from the lead to the stub
    location = %f metres\n",round(Ls*100)/100);
10 Lt=(lo/(2*%pi))*(atan(sqrt(1-(ampK*ampK)))/(2*ampK))
    ;
11 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

```

1  clear;
2  clc;

```

```

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

```

Scilab code Exa 6.18 Designing a quarter wave transformer

```

1 clear;
2 clc;
3 Zin=36;Zt=500;f=40;x=0.97
4 Zo=sqrt(Zin*Zt);
5 A=10^(Zo/276);
6 lo=300/f;
7 l1=lo*x/4;
8 printf("-The characteristic impedance of the
transmission line = %f ohms\n",round(Zo));
9 printf("-The spacing between the conductors shud be
%f times the radius of the conductor\n",round(A))
;
10 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 st

```

1 clear;
2 clc;
3 f=150;S=4.48;Ymin=6*(10^-2);
4 lo=300/(f); //lo=wavelength
5 b=(2*pi)/lo;

```

```

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

```

Scilab code Exa 6.20 calculating frequency and terminated impedance

```

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

```

Scilab code Exa 6.21 Calculating load and input impedance

```

1 clear;
2 clc;
3 Ymin=18*(10^-2);S=2.5;dmin=20*(10^-2);l=52*(10^-2);
      Zo=300;

```

```

4 //dmin=distance between adjacent voltage minimas
5 ampK=round(((S-1)/(S+1))*100)/100;
6 ZR=fix(Zo*(1+ampK)/(1-ampK));
7 printf("Input impedance = %f ohms\n",ZR);
8 lo=2*dmin; //lo=wavelength
9 b=(2*pi)/lo;
10 phi=(2*b*Ymin)-pi;
11 theta=-fix((phi-(2*b*1)));
12 Zm=Zo*(round((1+(ampK*exp(i*theta)))*100)/100)/(
    round((1-(ampK*exp(i*theta)))*100)/100);
13 printf("Load impedance = %f + j(%f) ohms",round(real
    (Zm*100))/100,round(imag(Zm*100))/100);
14 //the difference in result is due to erroneous
    value in textbook.
15 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

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

Scilab code Exa 7.2 Finding the value of the dominant mode

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

```

5 Zs=500;n=377;c=3*(10^8);
6 lo=sqrt(1-((n/Zs)^2))*lc;
7 f=c/lo;
8 f1=f/(10^7);
9 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 gr

```

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

```

Scilab code Exa 7.4 Calculating voltage attenuation

```

1 clear;
2 clc;

```

```

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

```

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

```

Scilab code Exa 7.6 Calculating cut off frequencies

```

1 clear;
2 clc

```



```

3 d=5;c=3*(10^8);
4 lo=1.706*d;
5 f=c/lo;
6 ff=f/(10^7);
7 printf("(i) Cutoff frequency for TE11 = %f GHz\n",
    round(ff*100)/100);
8 l=1.306*d;
9 fc=c/l;
10 ffc=fc/(10^7);
11 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 charac

```

1 clear;
2 clc;
3 c=3*(10^8);f=8*(10^9);r=2.5;h=1.84;n=377;
4 l=c/f;
5 lo=1*(10^2);
6 lc=2*pi*r/h;
7 printf("-Cutoff wavelength = %f cm\n",round(lc*100)
    /100);
8 lp=lo/(sqrt(1-((lo/lc)^2)));
9 printf("-Guide wavelength = %f cm\n",round(lp*100)
    /100);
10 Zo=n/(sqrt(1-((lo/lc)^2)));
11 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

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

Scilab code Exa 8.2 Calculating insertion loss

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

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

Scilab code Exa 8.3 Calculating insertion loss

```
1 clear;
2 clc;
3 S=2;
4 Lr=10*log10(((S+1)^2)/(4*S));
5 Ld=10*log10((S+1)/(S-1));
6 L=Ld+Lr;
7 printf("Insertion loss = %f db",round(L*100)/100);
8 //the difference in result is due to erroneous
   value in textbook.
9 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

```
1 clear;
2 clc;
3 R=10.4;L=3.67*(10^-3);G=0.8*(10^-6);C
   =0.00835*(10^-6);bmax=0.1;
4 f=5.5*(10^3); //artificial line will be designed fr
   highest frequency of operation
5 w=2*%pi*f;
6 Z=R+(%i*w*L);
7 Y=G+(%i*w*C);
8 P=sqrt(Z*Y);
9 b=imag(P);
10 l=bmax/b;
11 Zs=Z*l/2;
12 Zsh=1/(Y*l);
13 Zr=Y*l;
14 R1=real(Zs);
15 printf("-R1/2 = %f ohms\n",round(R1*10)/10);
16 L1=imag(Zs)/w;
17 printf("-L1/2 = %f mH\n",fix(L1*(10^3)*100)/100);
18 C2=imag(Zr)/w;
19 printf("-C2 = %f microfarads\n",fix(C2*(10^6)*10^4)
```

```

    /10^4);
20 G2=real(Zr);
21 R2=1/(round(G2*10^6*10)/10);
22 printf("-R2 = %f m ohms",R2);

```

Scilab code Exa 9.2 Designing an artificial line

```

1 clear;
2 clc;
3 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;
4 w=2*pi*f2;
5 Z=R+(%i*w*L);
6 Y=G+(%i*w*C);
7 P=fix(sqrt(Z*Y)*10^4)/10^4;
8 a=-f1*((-3*real(P)*real(P)*imag(P))-((imag(P))^3))
    /24;
9 a1=round(a*1000)/1000;
10 v=sqrt(bmax/a1);
11 l=f1/v;
12 R1=R*f1/(2*l);
13 L1=L*f1*10^3/(2*l);
14 C1=C*f1*10^6/l;
15 G1=G*f1/l;
16 Rg=1/G1;
17 printf("The elements of the artificial line are:\n")
    ;
18 printf(" R/2 = %f ohms\n",fix(R1*100)/100);
19 printf(" L/2 = %f mH\n",fix(L1*100)/100);
20 printf(" C = %f microfarads\n",fix(C1*1000)/1000);
21 printf(" Rg = %f k ohms\n",round((Rg*0.1)/100));

```

Scilab code Exa 9.3 Designing a delay line using T section

```

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

```

Scilab code Exa 9.4 Designing an artificial line

```

1 clear;
2 clc;
3 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
    ;
4 Z=R+(%i*w*L);
5 Y=G+(%i*w*C);
6 P=sqrt(Z*Y);
7 b=imag(P);
8 lmax=bmax/b;
9 l=s/20; //l<lmax
10 Zs=Z*l/2;
11 Zsh=1/(Y*l);
12 Zr=Y*l;
13 R1=real(Zs);
14 printf("R1 = %f ohms\n",round(R1*100)/100);
15 L1=imag(Zs)/w;
16 printf("L1 = %f mH\n",round(L1*(10^3)*100)/100);

```

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

Chapter 11

Networks

Scilab code Exa 11.1 Calculating image and iterative impedance

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

Scilab code Exa 11.2 Calculating iterative impedance

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

Scilab code Exa 11.3 Calculating iterative and image impedance

```
1 clear;
2 clc;
3 Z1=30+(%i*7.5);Z2=50+(%i*10);Z3=-%i*3229;
4 Za=Z1;Zb=Z2;Zc=Z3;
5 a=Za+Zc;
6 b=Zb+Zc;
7 s=(Za*Zb)+(Zb*Zc)+(Zc*Za);
8 Zi1=sqrt(a*s/b);
9 printf("Image impedances Zi1 = %f /_ %f ohms\n",
      round(abs(Zi1)*10)/10,round(atan(imag(Zi1),real(
      Zi1))*180*100/%pi)/100);
10 Zi2=sqrt(b*s/a);
11 printf(" Image impedances Zi1 = %f /_ %f ohms\n",
      round(abs(Zi2)*10)/10,round(atan(imag(Zi2),real(
```

```

    Zi2))*180*100/%pi)/100);
12 Zt1=(1/2)*((Za-Zb)+sqrt(((Za-Zb)^2)+(4*s)));
13 printf(" Iterative impedances Zt1 = %f + j(%f) ohms\
    n",round(real(Zt1)*100)/100,round(imag(Zt1)*100)
    /100);
14 Zt2=(1/2)*((Zb-Za)+sqrt(((Za-Zb)^2)+(4*s)));
15 printf(" Iterative impedances Zt1 = %f + j(%f) ohms\
    n",round(real(Zt2)*100)/100,round(imag(Zt2)*100)
    /100);
16 //the difference in result is due to erroneous
    value in textbook.
17 disp("The difference in result is due to erroneous
    value in textbook")

```

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

```

1 clear;
2 clc;
3 Za=%i*300;Zc=-%i*700;
4 Zoc1=Za+Zc;
5 Zsc1=Za;
6 Zoc2=Zc;
7 Zsc2=(Za*Zc)/(Za+Zc);
8 Zi1=sqrt(Zoc1*Zsc1);
9 printf("-Image impedance Zi1 = %f ohms\n",round(Zi1)
    );
10 Zi2=sqrt(Zoc2*Zsc2);
11 printf("-Image impedance Zi2(in ohms)= %f ohms\n",
    round(Zi2));
12 Zt1=(Za/2)+(sqrt((Za*Za/4)+(Za*Zc)));
13 A=real(Zt1)
14 B=imag(Zt1);
15 printf("-Iterative impedance Zt1 = %f + j(%f) ohms\n
    ",round(A),B);
16 Zt2=(-Za/2)+(sqrt((Za*Za/4)+(Za*Zc)));

```

```

17 C=real(Zt2);
18 D=imag(Zt2);
19 printf("-Iterative impedance Zt2 = %f + j(%f) ohms\n",
    ",round(C),D);
20 I=(1+(sqrt(Zsc1/Zoc1)))/(1-(sqrt(Zsc1/Zoc1)));
21 Qi=(log(I))/2;
22 E=real(Qi);
23 F=imag(Qi);
24 printf("-Image transfer constant = %f + j(%f)\n",E,
    round(F*1000)/1000);
25 I1=(Zt1+Zc);
26 I2=Zc;
27 Q2=log(I1/I2);
28 G=real(Q2);
29 H=imag(Q2);
30 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

```

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

```

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

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

Scilab code Exa 11.7 Calculating current

```
1 clear;
2 clc;
3 Z1=20;Z2=10;Vrms=10;
4 Zot=sqrt(((Z1*Z1)/4)+(Z1*Z2));
5 I=Vrms/Zot;
6 a=[30 27.32;1 -3.732];
7 b=[0.577;0];
8 b=inv(a)*b;
9 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

```
1 clear;
2 clc;
3 Z1=50+(%i*125);Z2=200-(%i*100);
4 Zot=sqrt((Z1/4)*(Z1+(4*Z2)));
5 A=real(Zot);
6 B=imag(Zot);
```

```
7 printf("Characteristic impedance = %f + j(%f) ohms",
    round(A), round(B*100)/100);
```

Scilab code Exa 11.9 Designing a L section

```
1 clear;
2 clc;
3 Z1=20+(-%i*5);w=5*(10^6);Rg=600;
4 R1=real(Z1);
5 Xc=-%i*imag(Z1);
6 A=imag(Xc);
7 printf("Compensating reactance = j(%d) ohms\n ",A);
8 X21=-sqrt(R1*(Rg-R1));
9 X22=-X21;
10 X31=-Rg*sqrt(R1/(Rg-R1));
11 X32=-X31;
12 X2a=X22+(Xc/%i);
13 L2=X2a/w;
14 C3=-1/(w*X31);
15 printf("(a)In the first case X2 is inductive , X2=L2
    = %f micro-henry\n",round(L2*(10^6)*100)/100);
16 printf("\t\t\t\t\tX3=C3 = %f pf\n",round(C3*(10^12))
    );
17 X2b=X21+(Xc/%i);
18 C2=-1/(w*X2b);
19 L3=X32/w;
20 printf("(b)In the second case X2 is capacitive ,X2=C2
    = %f pf\n",round(C2*(10^12)));
21 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

```

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

```

Scilab code Exa 11.11 Determining the equivalent T network

```

1 clear;
2 clc;
3 Zin=400+(%i*4000);Zout1=100+(%i*1000);Zout2=38+(%i
    *380);
4 Zoc1=Zin;Zoc2=Zout1;Zsc2=Zout2;
5 Z3=sqrt(Zoc1*(Zoc2-Zsc2));
6 Z1=Zoc1-Z3;
7 Z2=Zoc2-Z3;
8 A=real(Z1);
9 B=imag(Z1);
10 C=real(Z2);
11 D=imag(Z2);
12 E=real(Z3);
13 F=imag(Z3);
14 printf("-Z1 = %f + j(%f) ohms\n",round(A*10)/10,

```

```

    round(B*10)/10);
15 printf("-Z2 = %f + j(%f) ohms\n", fix(C*10)/10, fix(D)
    );
16 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

```

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

```

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

```

1 clear;
2 clc;
3 Ya=40*(10^-3); Yb=50*(10^-3); Yc=20*(10^-3);
4 Za=1/Ya; Zb=1/Yb; Zc=1/Yc;
5 Z1=Zb*Zc/(Za+Zb+Zc);
6 Z2=Za*Zc/(Za+Zb+Zc);
7 Z3=Zb*Za/(Za+Zb+Zc);

```

```

8 Zi1=sqrt(((Z3+Z1)/(Z3+Z2))*((Z1*Z2)+(Z2*Z3)+(Z1*Z3))
);
9 printf("(a) Zi1 = %f ohms\n",round(Zi1));
10 Zi2=sqrt(((Z3+Z2)/(Z3+Z1))*((Z1*Z2)+(Z2*Z3)+(Z1*Z3))
);
11 printf("    Zi2 = %f ohms\n",round(Zi2*100)/100);
12 Zt1=(1/2)*((Z1-Z2)+(sqrt(((Z1-Z2)^2)+(4*((Z1*Z2)+(Z2
*Z3)+(Z1*Z3))))));
13 printf("    Zt1 = %f ohms\n",fix(Zt1*100)/100);
14 Zt2=(1/2)*((Z2-Z1)+(sqrt(((Z1-Z2)^2)+(4*((Z1*Z2)+(Z2
*Z3)+(Z1*Z3))))));
15 printf("    Zt2 = %f ohms\n\n",fix(Zt2*100)/100);
16 Zb1=Za*Zb/(Za+Zb);
17 Z11=Zb*Zc/(Zb+Zc+Zb1);
18 Z21=Zb1*Zc/(Zb+Zc+Zb1);
19 Z31=Zb1*Zb/(Zb+Zc+Zb1);
20 Zr=Zc+Z21;
21 Zs=Z21+Zb;
22 Z12=Z31*Zs/(Z31+Zr+Za);
23 Z22=Zr*Za/(Z31+Zr+Za);
24 Z32=Z31*Za/(Z31+Zr+Za);
25 Z121=Z12+Z11;
26 printf("    The desired T network will be as:\n");
27 printf("    Z1 = %f ohms\n",round(Z121*100)/100);
28 printf("    Z2 = %f ohms\n",fix(Z22*10)/10);
29 printf("    Z3 = %f ohms\n",round(Z32*10^4)/10^4);

```

Scilab code Exa 11.14 Calculating the characteristic impedance and propagation fun

```

1 clear;
2 clc;
3 L=20*(10^-3);C=0.064*(10^-6);f=400;
4 w=2*pi*f;
5 Z1=round(2*i*w*L*10)/10;
6 Z2=1/(i*w*C);

```



```

7 Zo=sqrt((Z1*Z1/4)+(Z1*Z2));
8 printf("Characteristic impedance = %f ohms\n",round(
    Zo*100)/100);
9 Pf=(1+(Z1/(2*Z2)));
10 printf(" Propagation constant = %f",round(Pf*100)
    /100);
11 //the difference in result is due to erroneous
    value in textbook.
12 disp("The difference in result is due to erroneous
    value in textbook")

```

Scilab code Exa 11.15 Calculating characteristic impedance and propagation function

```

1 clear;
2 clc;
3 Za=300;Zc=600;R=1000;
4 Zi1=sqrt((Za*Za)+(Za*Zc));
5 Zi2=Za*Zc/sqrt((Za*Za)+(Za*Zc));
6 Zt1=(Za/2)+sqrt((Za*Za/4)+(Za*Zc));
7 Zt2=(-Za/2)+sqrt((Za*Za/4)+(Za*Zc));
8 printf("The image impedances are:\n ");
9 printf(" Zi1 = %f ohms\n",round(Zi1*10)/10);
10 printf(" Zi2 = %f ohms\n",round(Zi2*10)/10);
11 printf(" Zt1 = %f ohms\n",Zt1);
12 printf(" Zt2 = %f ohms\n\n",Zt2);
13 I=((R+Zt1+Zt2)*(R+Zt1)/(Zt1))-Zt1*(1/(R+R));
14 Ir=20*log10(round((I*1000))/1000);
15 printf(" Insertion loss = %f db",Ir);

```

Scilab code Exa 11.16 Designing a L matching loss less network

```

1 clear;
2 clc;

```

```

3 Rg=100;Rl=50;f=5*(10^6);
4 w=2*%pi*f;
5 X21=sqrt(Rl*(Rg-Rl));
6 X22=-X21;
7 X31=-Rg*sqrt(Rl/(Rg-Rl));
8 X32=-X31;
9 L2=X21/w;
10 printf("(i)X2 is inductive and X3 is capacitive
      where\n X2=L2 = %f microhenry\n",round(L2*(10^6)
      *1000)/1000);
11 C2=-1/(w*X31);
12 printf(" X3=C3 = %f pf\n",round(C2*(10^12)*10)/10);
13 L31=X32/w;
14 printf("(ii)X3 is inductive and X2 is capacitive
      where\n X3=L3 = %f microhenry\n",round(L31
      *(10^6)*1000)/1000);
15 C21=-1/(w*X22);
16 printf(" X2=C2 = %f pf",round(C21*(10^12)));

```

Scilab code Exa 11.17 Calculating different losses and total insertion loss

```

1 clear;
2 clc;
3 Zl=100;Zsh=500;Zg=300;
4 Zoc=Zl+Zsh;
5 Zsc=Zl+(1/((1/Zl)+(1/Zsh)));
6 Zi1=sqrt(Zoc*Zsc);
7 Zi2=Zi1;
8 theta=atanh(sqrt(Zsc/Zoc));
9 att=theta*8.686;
10 printf("-Attenuation loss = %f db\n",round(att*10)
      /10);
11 inp=20*log10(round((Zi1+Zg))/(2*sqrt(Zi1*Zg)));
12 printf("-Loss due to mismatch at the input = %f db\n
      ",round(inp*100)/100);

```

```

13 out=20*log10((Zi2+Zl)/(2*(sqrt(Zi2*Zl))));
14 printf("-Loss due to mismatch at output = %f db\n",
    round(out*100)/100);
15 Ki1=(Zi1-Zg)/(Zi1+Zg);
16 Ki2=(Zi2-Zl)/(Zi2+Zl);
17 inte=-20*log10(1-(Ki1*Ki2*exp(-2*theta)));
18 printf("-Loss due to interaction = %f db\n",fix(inte
    *100)/100);
19 ext=20*log10((Zg+Zl)/(2*sqrt(Zg*Zl)));
20 printf("-External reflection loss = %f db\n",round(
    ext*100)/100);
21 tot=att+inp+out-inte-ext;
22 printf("-Total insertion loss = %f db\n",fix(tot
    *100)/100);

```

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

```

1 clear;
2 clc;
3 Rg=8000;Zl=500+(%i*500);f=5*(10^6);
4 //value of f as taken in solution
5 w=2*pi*f;
6 Xc=-%i*imag(Zl);
7 Rl=real(Zl);
8 X21=sqrt(Rl*(Rg-Rl));
9 X22=-X21;
10 X31=-Rg*sqrt(Rg/(Rg-Rl));
11 X32=-X31;
12 X2a=X21+(Xc/%i);
13 L2a=X2a/w;
14 C3a=-1/(w*X31);
15 printf("(a)X2 is inductive and X3 is capacitive
    where\n    X2=L2 = %f mH\n",round(L2a*(10^3)*1000)
    /1000);
16 printf("    X3=C3 = %f pf\n",round(C3a*(10^12)*1000)

```

```

    /1000);
17 X2b=X22+(Xc/%i);
18 C2b=-1/(w*X2b);
19 L3b=X32/w;
20 printf("(b)X2 is capacitive and X3 is inductive
    where\n    X2=C2 = %f pf\n",round(C2b*(10^12)*100)
    /100);
21 printf("    X3=L3 = %f mH",round(L3b*(10^3)*1000)
    /1000);

```

Scilab code Exa 11.19 Calculating image impedance and transfer constant

```

1 clear;
2 clc;
3 Zoc1=-%i*400;Zoc2=-%i*600;Zsc1=%i*267;Zsc2=%i*400;
4 A=Zoc1/Zsc1;
5 B=Zoc2/Zsc2;
6 printf("(a)Since Zoc1/Zsc1 = Zoc2/Zsc2 = %f the
    results are consistant\n",round(A*10)/10);
7 Zi1=sqrt(Zoc1*Zsc1);
8 Zi2=sqrt(Zoc2*Zsc2);
9 printf("The image impedances are:\n Zi1 = %f ohms\n
    Zi2 = %f ohms\n",round(Zi1*10)/10,round(Zi2*10)
    /10);
10 C=(1+sqrt(B))/(1-sqrt(B));
11 phi=round(atan(imag(C),real(C))*180*10/%pi)/10;
12 theta=round(%i*phi*%pi*1000/(2*180))/1000;
13 printf("Image transfer constant = j %f",theta/%i);

```

Scilab code Exa 11.20 Calculating reduction in power

```

1 clear;
2 clc;

```

```

3 Zg=300;Zi=400;Zi1=600;theta=10;
4 thetai=theta/8.686;
5 Zi2=Zi1;
6 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))))));
7 Ir1=(round(Ir*10)/10)*8.686;
8 printf("The reduction in power will be = %f db",
    round(Ir1*100)/100);

```

Scilab code Exa 11.21 Calculating characteristic impedance and attenuation constant

```

1 clear;
2 clc;
3 R1=200;L1=100*(10^-3);R2=200;L2=100*(10^-3);C2
    =2.5*(10^-6);w=2000;
4 Z1=R1+(%i*w*L1);
5 Z2=1/(%i*w*C2);
6 Zoc=Z1+Z2;
7 Zsc=Z1+(1/((1/Z1)+(1/Z2)));
8 Zo=sqrt(Zoc*Zsc);
9 printf("-Characteristic impedance = %f ohms\n",round
    (Zo));
10 P=atanh(sqrt(Zsc/Zoc));
11 a=real(P);
12 printf("-Attenuation constant = %f nepers\n",round(a
    *100)/100);
13 b=(imag(P))*180/%pi;
14 printf("-Phase shift constant = %d degrees",b);

```

Scilab code Exa 11.22 Calculating image impedance and propagation constant and ele

```

1 clear;
2 clc;
3 Z1o=1260*(exp(%i*30/(180/(%pi))));Z2o=2430*(exp(-%i
    *34/(180/(%pi))));
4 Z1s=318*(exp(%i*72/(180/(%pi))));Z2s=613*(exp(%i
    *8/(180/(%pi))));
5 Zi1=sqrt(Z1o*Z1s);
6 A=real(Zi1);
7 B=imag(Zi1);
8 printf("(i)Image impedance Zi1 = %f /_ %f ohms\n",
    round(abs(Zi1)),atan(B,A)*180/%pi);
9 Zi2=sqrt(Z2o*Z2s);
10 C=real(Zi2);
11 D=imag(Zi2);
12 printf("  Image impedance Zi2 = %f /_ %f ohms\n",
    round(abs(Zi2)),atan(D,C)*180/%pi);
13 Z3=sqrt(Z2o*(round(Z1o)-round(Z1s)));
14 Z1=Z1o-Z3;
15 Z2=Z2o-Z3;
16
17 P=atanh(sqrt(Z1s/Z1o));
18 printf(" (ii)Propagation constant = %f /_ %f\n",
    round(abs(P*100))/100,round(atan(imag(P),real(P))
    *100)/100);
19 printf(" (iii)The elements of the T network are:\n")
    ;
20 printf("      Z1 = %f + j(%f) ohms\n",round(real(Z1))
    ,round(imag(Z1)));
21 printf("      Z2 = %f + j(%f) ohms\n",round(real(Z2))
    ,round(imag(Z2)));
22 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

```
1 clear;
2 clc;
3 w=2*pi*(10^6);C=100*(10^-12);V=10;L=100*(10^-6);
4 Zc=1/(w*C);
5 Ic=(V/sqrt(2))/Zc;
6 printf("R.m.s. value of current flowing through
   capacitor C = %f mA\n",round(Ic*(10^3)*100)/100);
7 Zl=w*L;
8 Il=(V/sqrt(2))/Zl;
9 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

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

```

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

```

Scilab code Exa 12.3 Calculating current

```

1 clear;
2 clc;
3 R1=5;R2=%i*5;R3=3+(%i*4);V1=50*exp(%i*90/(180/%pi));
  V2=50;
4 Z1=R1+(1/((1/R2)+(1/R3)));
5 I1=V1/Z1;
6 Iz1=I1*R2/(R2+R3);
7 Z2=R2+(1/((1/R1)+(1/R3)));
8 I2=V2/Z2;
9 Iz2=-I2*R1/(R1+R3);
10 Iz=Iz1+Iz2;
11 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

```

1 clear;
2 clc;
3 Isc=10;Voc=120;R1=8;I1=6;
4 Zeq=Voc/Isc; //Zeq=impedance of the equivalent
  thevenin's generator Z=R+jX
5 t=Voc/Isc; //(R^2)+(X^2)=(t^2)
6 R=((Voc/I1)^2)-(t^2)-(R1^2)/(2*R1);
7 X=sqrt((t^2)-(R^2));

```



```

8 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

```

1 clear;
2 clc;
3 V=10;R1=5;R2=%i*5;R3=3+(%i*4);
4 Zab=R2+(1/((1/R1)+(1/R3)));
5 A=real(Zab);
6 B=imag(Zab);
7 I=V/(R1+R3);
8 Voc=I*R3;
9 C=real(Voc);
10 D=imag(Voc);
11 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

```

1 clear;
2 clc;
3 Voc=150;R1=10;I1=2.65;Isc=3;
4 t=Voc/Isc; //(R^2)+(X^2)=(t^2)
5 R=(((Voc/I1)^2)-(t^2)-(R1^2))/(2*R1);
6 X=sqrt((t^2)-(R^2));
7 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);

```

```

8 if X>0 then
9     printf("–Since X has a positive value the
           circuit will be inductive");
10 else
11     disp("–Since X has a negative value the circuit
           will be capacitive");
12 end

```

Scilab code Exa 12.8 Determining the Nortons equivalent circuit

```

1 clear;
2 clc;
3 Za=-%i*5;Zb=5+(%i*5);V=50;Z1=5-(%i*5);Z2=10;
4 Isc=V/Za;
5 A=real(Isc);
6 B=imag(Isc);
7 printf("–Short circuit current = %d /- %d Amps\n",
       abs(Isc),atan(B,A)*180/%pi);
8 Zab=1/((1/Za)+(1/Zb));
9 C=real(Zab);
10 D=imag(Zab);
11 printf("–Equivalent impedance = %d + j(%d) ohms\n",C
       ,D);
12 I1=Isc*(Zab/(Zab+Z1));
13 E=real(I1);
14 F=imag(I1);
15 printf("–Current through impedance Z1(=5-j5) = %d /-
       %d Amps\n",abs(I1),atan(F,E)*180/%pi);
16 I2=Isc*(Zab/(Zab+Z2));
17 G=real(I2);
18 H=imag(I2);
19 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

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

Scilab code Exa 12.10 Verifying reciprocity theorem

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

Scilab code Exa 12.11 Calculating value of compensation source

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

```

4 Zeq=1/((1/Z1)+(1/Z2));
5 Zi=Z+Zeq;
6 I=V/Zi;
7 Vc=I*Zeq;
8 A=real(Vc);
9 B=imag(Vc);
10 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

```

1 clear;
2 clc;
3 Zg=10000;Zl=24+(%i*7);Vrms=100;
4 Rl=real(Zl);
5 Xl=imag(Zl);
6 Zeq=sqrt(((Rl+Zg)^2)+((Xl)^2));
7 Irms=Vrms/Zeq;
8 P=Irms*Irms*Rl;
9 printf("(i)Power in the load if connected directly
        to the generator = %f mW\n",round(P*(10^3)*10)
        /10);
10 ampZl=sqrt((Rl*Rl)+(Xl*Xl));
11 n=1/sqrt(ampZl/Zg);
12 printf("(ii)The desired turn ratio of transformer is
        1:%d\n",n);
13 P1=Vrms*Vrms/(4*Zg);
14 printf("(iii)Power transferred under ideal load
        conditions =%f mW",P1*(10^3));

```

Scilab code Exa 12.13 Determining value of maximum power

```

1 clear;
2 clc;
3 Zg=10+(%i*20);V=50;
4 Rg=real(Zg);Xg=imag(Zg);
5 ampZg=sqrt((Rg*Rg)+(Xg*Xg));
6 Rl=ampZg;
7 I=V/(Zg+Rl);
8 ampI=round(sqrt((real(I))^2+(imag(I))^2)*100)/100;
9 P=ampI*ampI*round(Rl*10)/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 t

```

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

```

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

```

1 clear;
2 clc;
3 f=5*(10^6);C=400*(10^-12);R=10*(10^3);
4 w=2*pi*f;
5 L=2/(w*w*C);
6 r=1/(w*w*C*C*R);
7 printf("-Effective resistance of the coil = %f ohms\n",round(r*100)/100);
8 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

```

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

```

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

```
1 clear;
2 clc;
3 w1=1000;w2=1500;w3=2000;w4=3000;w5=5000;w=100;Z=%i
  *100;
4 F=((w*w)-(w2*w2))*((w*w)-(w4*w4))/(((w*w)-(w1*w1))
  *((w*w)-(w3*w3))*((w*w)-(w5*w5)));
5 H=Z/(%i*w*F);
6 A=((w1*w1)-(w2*w2))*((w1*w1)-(w4*w4))/(((w1*w1)-(w3*
  w3))*((w1*w1)-(w5*w5)));
7 B=((w3*w3)-(w2*w2))*((w3*w3)-(w4*w4))/(((w3*w3)-(w5*
  w5))*((w3*w3)-(w1*w1)));
8 C=((w5*w5)-(w2*w2))*((w5*w5)-(w4*w4))/(((w5*w5)-(w1*
  w1))*((w5*w5)-(w3*w3)));
9 C2=-1/(H*A);
10 printf("-C2 = %f microfarads\n",round(C2*(10^6)*100)
  /100)
11 L2=1/(w1*w1*C2);
12 printf("-L2 = %f henry\n",round(L2*100)/100);
13 C4=-1/(H*B);
14 printf("-C4 = %f microfarads\n",round(C4*(10^6)*100)
  /100);
15 L4=1/(w3*w3*C4);
16 printf("-L4 = %f henry\n",round(L4*100)/100);
17 C6=-1/(H*C);
18 L6=1/(w5*w5*C6);
19 printf("-C6 = %f microfarads\n",round(L6*100)/100);
20 printf("-L6 = %f henry\n",round(C6*(10^6)*100)/100)
```

Scilab code Exa 12.21 Finding current

```

1 clear;
2 clc;
3 V=10;R1=10;Z1=5;Z2=5;
4 Zab=1/((1/Z1)+(1/Z2));
5 I1=V/(Z1+Z2);
6 Voc=I1*Z1;
7 I=Voc/(Zab+R1);
8 printf("Current in the 10 ohm resistor = %d mA",I
        *(10^3));

```

Scilab code Exa 12.22 Using Thevenins theorem to calculate power

```

1 clear;
2 clc;
3 R1=10;V=100;Z1=-%i*30;Z2=-%i*30;Z3=20+(%i*10);
4 I=V/(Z1+Z2);
5 Voc=I*Z2;
6 Zab=1/((1/Z1)+(1/Z2))+Z3;
7 Z=Zab+R1;
8 I1=Voc/Z;
9 ampI1=sqrt(real(I1)^2+imag(I1)^2);
10 P1=ampI1*ampI1*R1;
11 printf("Power in the load = %f Watts",round(P1*100)
        /100);

```

Scilab code Exa 12.23 Finding Thevenins equivalent generator

```

1 clear;
2 clc;
3 V=100;R1=10;R2=10;R3=10;R4=10;
4 I=V/(R1+R2);
5 Vab=I*R2;
6 Zab=1/((1/R1)+(1/R2));

```



```

7 V1=Vab*R4/(Zab+R3+R4);
8 Z1=1/((1/(Zab+R3))+(1/R4));
9 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 po

```

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

```

Scilab code Exa 12.26 Determining Thevenins equivalent circuit

```

1 clear;
2 clc;
3 V=100; R1=50; R2=60; R3=40; R4=60; R5=40;
4 Rac=1/((1/(R2+R3))+(1/(R4+R5)));
5 Rt=Rac+R1;
6 I=V/Rt;
7 V1=I*R1;
8 Vac=V-V1;
9 Vab=R2*Vac/(R2+R3);

```

```

10 Vad=R5*Vac/(R4+R5);
11 Vbd=Vab-Vad;
12 Voc=Vbd;
13 R1y=R5*R1/(R1+R4+R5);
14 R2y=R1*R4/(R1+R4+R5);
15 R3y=R4*R5/(R1+R4+R5);
16 Rbd=R3y+(1/((1/(R1y+R2))+(1/(R2y+R3))));
17 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

```

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

```

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

```

1 clear;
2 clc;
3 V1=5;V2=4;R1=5;R2=4;I3=1;R3=10;R4=5;
4 V3=I3*R3;
5 Y1=1/R1;Y2=1/R2;Y3=1/R3;
6 V4=((V1*Y1)+(V2*Y2)+(V3*Y3))/(Y1+Y2+Y3);
7 I4=V4/R4;
8 printf("Voltage across resistor R4 = %f volts\n",V4)
    ;

```

```

9 printf("Current in resistor R4 = %f Amp",I4);
10 //the difference in result is due to erroneous
    value in textbook.
11 disp("The difference in result is due to erroneous
    value in textbook")

```

Scilab code Exa 12.29 Verifying Tellegens theorem

```

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

```

Scilab code Exa 12.30 Verifying Tellegens theorem

```

1 clear;
2 clc;
3 V1=4;V2=2;V3=2;V4=3;V5=-1;V6=-5;i1=2;i2=2;i3=4;i4
    =-2;i5=-6;i6=4;
4 Va=V2+V3-V1;
5 printf("V2+V3-V1= %d volts\n",Va);
6 Vb=V2+V4+V6;
7 printf("V2+V4+V6= %d volts\n",Vb);

```

```

8 Vc=V4+V5-V3;
9 printf("V4+V5-V6= %d volts\n",Vc);
10 printf("-Hence KVL is satisfied for all three loops
    PQTU,PRYX and QRST respectively.\n\n");
11 Ia=i1+i2-i6;
12 printf("i1+i2-i6= %d Amp\n",Ia);
13 Ib=i3+i4-i2;
14 printf("i3+i4-i2= %d Amp\n",Ib);
15 Ic=i5+i6-i4;
16 printf("i5+i6-i2= %d Amp\n",Ic);
17 printf("-Hence KCL is satisfied at all three nodes P
    ,Q and R respectively.\n\n");
18 VI=(V1*i1)+(V2*i2)+(V3*i3)+(V4*i4)+(V5*i5)+(V6*i6);
19 printf("-Summation V*I= %d\n\n",VI);
20 printf("-This proves Tellegens theorem.");

```

Scilab code Exa 12.31 Finding current

```

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

```

Scilab code Exa 12.32 Calculating current and voltage using Thevenins theorem

```

1 clear;
2 clc;
3 I1=6;I2=8;R1=4;R2=4;R3=6;R4=4;Rl=3;
4 V1=I1*R1;V2=I2*R4;
5 Voc1=V1*(R3+R4)/(R1+R2+R3+R4);

```

```
6 Voc2=V2*(R1+R2)/(R1+R2+R3+R4);
7 Voc=Voc1+Voc2;
8 Zab=1/((1/(R1+R2))+(1/(R3+R4)));
9 I1=Voc/(Zab+R1);
10 printf("-i1 = %f Amp\n",round(I1*10)/10);
11 e1=I1*R1;
12 printf("-e1 = %f volts",round(e1*10)/10);
```

Chapter 13

Network Parameters

Scilab code Exa 13.1 Determining z and y parameters

```
1 clear;
2 clc;
3 Z1=%i*20; Z2=%i*25; Z3=30;
4 Z11=Z1+Z3; //Z11=V1/I1 when I2=0
5 Z12=Z3; //Z12=V1/I2 when I1=0
6 Z21=Z3; //Z21=V2/I1 when I2=0
7 Z22=Z2+Z3; //Z22=V2/I2 when I1=0
8 printf("(a)The z-parameters are\n");
9 printf("  Z11 = %f + j(%f) ohms\n", real(Z11), imag(
    Z11));
10 printf("  Z12 = %f ohms\n", Z12);
11 printf("  Z21 = %f ohms\n", Z21);
12 printf("  Z22 = %f + j(%f) ohms\n", real(Z22), imag(
    Z22));
13 deltax=(Z11*Z22)-(Z12*Z21);
14 y11=Z22/deltax;
15 y12=-Z12/deltax;
16 y21=-Z21/deltax;
17 y22=Z11/deltax;
18
19 printf("(b)The y-parameters are\n");
```

```

20 printf("  y11 = %f + j(%f) mhos\n",real(y11),imag(
    y11));
21 printf("  y12 = %f + j(%f) mhos\n",real(y12),imag(
    y12));
22 printf("  y21 = %f + j(%f) mhos\n",real(y21),imag(
    y21));
23 printf("  y22 = %f + j(%f) mhos\n",real(y22),imag(
    y22));

```

Scilab code Exa 13.2 Obtaining the open circuit impedance parameters

```

1 clear;
2 clc;
3 L=1;C=1;R=1;
4 printf(" z11 = s + %d ohms\n",C);
5 printf(" z21 = %d ohms\n",C);
6 printf(" z22 = 1/s + %d ohms\n",C);
7 printf(" z12 = %d ohms\n",C);

```

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

```

1 clear;
2 clc;
3 Voc2=100;I1=10;V2=25;
4 Voc1=100;I2=20;V1=50;
5 z11=Voc1/I1;
6 z21=V2/I1;
7 z22=Voc2/I2;
8 z12=V1/I2;
9 printf("(a)The driving point impedances are:\n");
10 printf("  z11 = %f ohms\n",z11);
11 printf("  z22 = %f ohms\n\n",z22);
12 printf("  The transfer impedances are:\n");

```

```

13 printf(" z12 = %f ohms\n", z12);
14 printf(" z21 = %f ohms\n\n", z21);
15 printf(" The loop equations are:\n");
16 printf(" V1= %f*I1 + %f*I2\n", z11, z12);
17 printf(" V2= %f*I1+ %f*I2\n\n", z21, z22);
18 R1=10;
19 a=[z11 z12 ; z21 (z22+R1)];
20 b=[100 ; 0];
21 b=inv(a)*b;
22 I2=b(2,1);
23 V1=-I2*R1;
24 printf("(b) Voltage across resistor = %f volts", round
      (V1*10)/10);

```

Scilab code Exa 13.4 Computing y parameters

```

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

```

Scilab code Exa 13.5 Finding the z parameters

```

1 clear;

```



```

2  clc;
3  Y1=1;Y2=1;Y3=2;V3=3;
4  Z1=1/Y1;Z2=1/Y2;Z3=1/Y3;
5  V1=1;I1=-1;
6  z11=V1/I1;
7  V2=1;I2=3;
8  z22=V2/I2;
9  z21=V2/I1;
10 printf(" z11 = %f ohms\n",z11);
11 printf("  z22 = %f ohms\n",z22);
12 printf("  z21 = %f ohms\n",z21);
13 printf("  z11 = %f ohms\n",0);

```

Scilab code Exa 13.6 Computing h parameters

```

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

```

```

15 //the difference in result of h22 is due to
    erroneous value in textbook.
16 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

```

1 clear;
2 clc;
3 z11=40; z22=30; z12=20; z21=20;
4 deltaz=(z11*z22)-(z12*z21);
5 A=z11/z12;
6 printf("The transmission parameters are:\n");
7 printf("  A = %f\n",A);
8 B=deltaz/z21;
9 printf("  B = %f ohms\n",B);
10 C=1/z21;
11 printf("  C = %f mho\n",C);
12 D=z22/z21;
13 printf("  D = %f\n",D);
14 printf(" The network equations using z-parameter are
    \n");
15 printf("  V1 = %fI1 + %fI2\n",z11,z12);
16 printf("  V2 = %fI1 + %fI2\n",z21,z22);
17 printf(" The network equations using ABCD parameter
    are\n");
18 printf("  V1 = %fV2 - %fI2\n",A,B);
19 printf("  I1 = %fV2 - %fI2\n",C,D);

```

Scilab code Exa 13.8 Obtaining the transmission parameters

```

1 clear;
2 clc;

```

```

3 V1=1;R1=1;R2=2;R3=1;R4=2;I2=0;
4 //I2=0 because port 2-2' has been open circuited
5 a=[R1+R2 -R2;-2 5];
6 b=[1;0];
7 b=inv(a)*b;
8 I1=b(1,1);
9 I3=b(2,1);
10 V2=I3*R4;
11 A=V1/V2;
12 C=I1/V2;
13 V21=0; //because port 2-2' has been short circuited
14 c=[3 -2;-2 3];
15 d=[1;0];
16 d=inv(c)*d;
17 I11=d(1,1);
18 I21=d(2,1);
19 B=V1/I21;
20 D=I11/I21;
21 R=(A*D)-(B*C);
22 printf("The transmission parameters are:\n");
23 printf("  A = %f\n",A);
24 printf("  B = %f ohms\n",B);
25 printf("  C = %f mhos\n",C);
26 printf("  D = %f\n\n",D);
27 printf(" AD-BC = %f . Hence the circuit is
      reciprocal." ,R);

```

Scilab code Exa 13.9 Calculating input impedance

```

1 clear;
2 clc;
3 R1=3;z11=5;z12=2;z21=2;z22=1;
4 Zi=z11-(z12/z21);
5 printf("Input impedance = %f ohms",Zi);

```

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

```
1 clear;
2 clc;
3 z11=10; z22=20; z12=5; z21=5;
4 deltaz=(z11*z22)-(z12*z21);
5 A=z11/z12;
6 printf("(a)The ABCD parameters are:\n");
7 printf("  A = %f\n",A);
8 B=deltaz/z21;
9 printf("  B = %f ohms\n",B);
10 C=1/z12;
11 printf("  C = %f mho\n",C);
12 D=z22/z21;
13 printf("  D = %f\n\n",D);
14 Z3=z21;
15 Z1=z11-Z3;
16 Z2=z22-Z3;
17 printf("(b)The equivalent T-network:\n");
18 printf("  Z1 = %f ohms\n",Z1);
19 printf("  Z2 = %f ohms\n",Z2);
20 printf("  Z13 = %f ohms\n",Z3);
```

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

```
1 clear;
2 clc;
3 V1=10; I1=2.5; I2a=-0.5; //values with terminals 3-4
   short-circuited
4 V2=V1; I2b=1.5; //values with terminals 1-2 short
   circuited
5 Yb=-I2a/V1;
```

```

6 Ya=(I1/V1)-Yb;
7 Yc=(I2b/V2)-Yb;
8 printf("-Ya = %f mho\n",Ya);
9 printf("-Yb = %f mhp\n",Yb);
10 printf("-Yc = %f mho",Yc);

```

Scilab code Exa 13.12 Finding the equivalent pi network

```

1 clear;
2 clc;
3 Za=2;Zb=2.5;Zc=5;
4 Ya=1/Za;Yb=1/Zb;Yc=1/Zc;
5 Y1=(Ya*Yc)/(Ya+Yb+Yc);
6 Z1=1/Y1;
7 Y2=(Yb*Yc)/(Ya+Yb+Yc);
8 Z2=1/Y2;
9 Y3=(Ya*Yb)/(Ya+Yb+Yc);
10 Z3=1/Y3;
11 printf("The equivalent pi network is: \n");
12 printf(" Z1 = %f ohms\n",Z1);
13 printf(" Z2 = %f ohms\n",Z2);
14 printf(" Z3 = %f ohms\n",Z3);

```

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

```

1 clear;
2 clc;
3 y11=0.5;y12=-0.2;y21=-0.2;y22=1;
4 Y2=-y12;
5 Y1=y11-Y2;
6 Y3=y22-Y2;
7 deltay=(y11*y22)-(y12*y21);
8 z11=y22/deltay;

```

```

 9 z22=y11/deltay;
10 z12=y12/deltay;
11 z21=z12;
12 A=-y22/y12;
13 B=-1/y12;
14 C=-deltay/y12;
15 D=-y11/y12;
16 printf("The y parameters are\n");
17 printf(" y11 = %f mho\n",y11);
18 printf(" y12 = %f mho\n",y12);
19 printf(" y21 = %f mho\n",y21);
20 printf(" y22 = %f mho\n\n",y22);
21 printf(" The z parameters are\n");
22 printf(" z11 = %f ohm\n",round(z11*1000)/1000);
23 printf(" z12 = %f ohm\n",round(z12*10^4)/10^4);
24 printf(" z21 = %f ohm\n",round(z21*10^4)/10^4);
25 printf(" z22 = %f ohm\n\n",round(z22*1000)/1000);
26 printf(" The ABCD parametrs are\n");
27 printf(" A = %f\n",A);
28 printf(" B = %f ohm\n",B);
29 printf(" C = %f mho\n",C);
30 printf(" D = %f\n\n",D);
31 printf(" The equivalent pi network is\n");
32 printf(" Y1 = %f mho\n",Y1);
33 printf(" Y2 = %f mho\n",Y2);
34 printf(" Y3 = %f mho",Y3);

```

Scilab code Exa 13.14 Finding y parameters

```

1 clear;
2 clc;
3 Y1=1;Y2=2;Y3=1;
4 y11=Y1+Y2;
5 y12=-Y2;
6 y21=y12;

```

```

7 y22=Y2+Y3;
8 Y11=y11+y11;
9 Y12=y12+y12;
10 Y21=Y12;
11 Y22=y22+y22;
12 printf("The y-parameters are:\n");
13 printf(" Y11 = %f mho\n",Y11);
14 printf(" Y12 = %f mho\n",Y12);
15 printf(" Y21 = %f mho\n",Y21);
16 printf(" Y22 = %f mho\n",Y22);

```

Scilab code Exa 13.15 Determining the ABCD parameters and image impedance

```

1 clear;
2 clc;
3 Za=10;Zb=10;Zc=5;
4 Ya=1/Za;Yb=1/Zb;Yc=1/Zc;
5 A=1+(Za*Yb);
6 printf("The ABCD parameters are:\n");
7 printf(" A = %f\n",A);
8 B=Za+Zb+(Za*Zb*Yc);
9 printf(" B = %f ohm\n",B);
10 C=Yc;
11 printf(" C = %f mho\n",C);
12 D=1+(Zb*Yc);
13 printf(" D = %f \n\n",D);
14 Zi1=sqrt((A*B)/(C*D));
15 printf(" Image impedances of the two port network
are:\n");
16 printf(" Zi1 = %f ohms\n",round(Zi1*10)/10);
17 Zi2=sqrt((D*B)/(A*C));
18 printf(" Zi2 = %f ohms",round(Zi2*100)/100);

```

Scilab code Exa 13.17 Determining the open and short circuit impedance

```
1 clear;
2 clc;
3 R1=2;R2=1;R3=1;R4=2;
4 A=[R1+R2 R2 -R4;R2 R2+R4 R4;-R1 R1 R4+R3+R1];
5 B=det(A);
6 C=[1 R2 -R1;0 R2+R1 R1;0 R4 R4+R3+R1];
7 D=det(C);
8 E=[1 -R1;0 R4+R3+R1];
9 F=[R2+R1 -R1;-R1 R4+R3+R1];
10 G=[R4+R2 0 -R4;R1 1 R4;-R4 0 R4+R2];
11 H=[1 R1;0 R4+R3+R1];
12 I=[R2+R1 R1;R1 R4+R3+R1];
13 printf(" Z1s = %d/%d ohms\n",B,D);
14 printf(" Z1o = %d/%d ohms\n",det(F),det(E));
15 printf(" Z2s = %d/%d ohms\n",det(A),det(C));
16 printf(" Z2o = %d/%d ohms\n",det(I),det(H));
```

Scilab code Exa 13.18 Calculating y parameters

```
1 clear;
2 clc;
3 R1=2;R2=1;R3=2;R4=1;R5=2;
4 R3=R3/2;
5 z11=R2+R1;
6 z22=R3+R2;
7 z12=R2;
8 z21=R2;
9 Z11=z11+z11;
10 Z12=z12+z12;
11 Z21=z21+z21;
12 Z22=z22+z22;
13 deltaZ=(Z11*Z22)-(Z12*Z21);
14 y11=Z22/deltaZ;
```



```

15 y12=-Z12/deltaZ;
16 y21=y12;
17 y22=Z11/deltaZ;
18 printf("The y-parameters are:\n");
19 printf("-y11 = %f mho\n",y11);
20 printf("-y12 = %f mho\n",y12);
21 printf("-y21 = %f mho\n",y21);
22 printf("-y22 = %f mho\n",y22);

```

Scilab code Exa 13.19 Determining the image impedances

```

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

```

Chapter 14

Filters

Scilab code Exa 14.1 Calculating values of inductor and capacitor

```
1 clear;
2 clc;
3 Rk=600;fc=3000;
4 L=Rk/(%pi*fc);
5 printf("–Desired value of inductor L = %f mH\n",
        round(L*(10^3)*10)/10);
6 C=1/(%pi*Rk*fc);
7 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

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

```

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

```

Scilab code Exa 14.3 Calculating value of L and C

```

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

```

1 clear;
2 clc;
3 L=60*(10^-3);C=0.2*(10^-6);
4 fc=1/(pi*(sqrt(L*C)));
5 printf("(i)Cut off frequency fc for the low pass
    filter = %f kHz\n\n",round(fc*(10^-1))/100);
6 fa=1000;
7 Za=(sqrt(L/C))*(sqrt(1-((fa/fc)^2)));
8 printf(" (ii)Iterative impedance at f=1 KHz = %f
    ohms\n",fix(Za));

```

```

9 fb=5000;
10 Zb=(sqrt(L/C))*(sqrt(1-((fb/fc)^2)));
11 printf("    Iterative impedance at f=5 KHz = j(%f)
        ohms\n\n",fix(Zb/%i));
12 aa=0; //attenuation at frequency fa
13 ba=2*(asind(fa/fc)); //phase shift at frequency fa
14 ab=2*acosh(fb/fc); //attenuation at frequency fb
15 bb=180; //attenuation at frequency fb
16 V1=exp(aa); //V1=(Vin/Vout) at frequency of 1kHz
17 V2=exp(ab); //V2=(Vin/Vout) at frequency of 5kHz
18 printf(" (iii)At 5kHz,the voltage ratio = %f and
        phase difference = %f degrees\n",round(V2*10)/10,
        bb);
19 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

```

1 clear;
2 clc;
3 fc=1000;Rk=600;
4 L=Rk/(4*%pi*fc);
5 C=1/(4*%pi*fc*Rk);
6 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

```

1 clear;
2 clc;

```

```

3 a=1.15;
4 F=cosh(a/2);
5 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

```

1 clear;
2 clc;
3 Rk=600;fc=1000;fi=1050;
4 L0=Rk/(%pi*fc);
5 C0=1/(%pi*fc*Rk);
6 m=round(sqrt(1-((fc/fi)^2))*10)/10;
7 L1=m*L0/2;
8 printf("-mL/2 = %f mH\n",round(L1*(10^3)*100)/100);
9 C=m*C0;
10 printf("-mC = %f microfarads\n",round(C*(10^6)*1000)
   /1000);
11 L2=L0*((1-(m*m))/(4*m));
12 printf("-(1-m^2)L/(4m) = %f mH",L2*(10^3));
13 //the difference in result is due to erroneous value
   in textbook.
14 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

```

1 clear;
2 clc;
3 fc=20*(10^3);Rk=600;m=0.6;
4 L0=Rk/(4*%pi*fc);
5 C0=1/(4*%pi*fc*Rk);

```

```

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

```

Scilab code Exa 14.9 Designing a low pass filter

```

1 clear;
2 clc;
3 fc=2000;fi=2050;Rk=500; //fi=frequency at which
      infinite attenuation occurs
4 L0=Rk/(%pi*fc);
5 C0=1/(%pi*fc*Rk);
6 printf("The elements of the constant-K L.P. are:\n")
      ;
7 printf("  L = %f mH\n",fix(L0*(10^3)*10)/10);
8 printf("  C = %f microfarads\n\n",fix(C0*(10^6)
      *1000)/1000);
9 m1=round(sqrt(1-((fc/fi)^2))*100)/100;
10 L1=m1*L0/2;
11 L2=(1-(m1*m1))*L0/(4*m1);
12 C1=m1*C0;
13 printf("The elements of the m-derived L.P.T. filter
      are:\n");
14 printf("  mL/2 = %f mH\n",fix(L1*(10^3)*100)/100);
15 printf("  mC = %f microfarads\n",fix(C1*(10^6)*100)
      /100);
16 printf("  (1-m^2)L/4m = %f mH\n\n",fix(L2*(10^3)
      *100)/100);
17 m2=0.6;
18 L3=m2*L0/2;

```

```

19 L4=(1-(m2*m2))*L0/(4*m2);
20 C2=m2*C0;
21 printf("The elements of the terminating half
    sections m-derived L.P.T. filter are:\n");
22 printf("  mL/2 = %f mH\n",fix(L3*(10^3)*10)/10);
23 printf("  mC = %f microfarads\n",fix(C2*(10^6)
    *10000)/10000);
24 printf("  (1-m^2)L/4m = %f mH\n\n",fix(L4*(10^3)*10)
    /10);
25 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

```

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

```

```

15 printf("  L/m = %f mH\n",round(L1*(10^3)*10)/10);
16 printf("  4mC/(1-m^2) = %f microfarads\n\n",fix(C2
    *(10^6)*100)/100);
17 m2=0.6;
18 C3=2*C0/m2;
19 L2=L0/m2;
20 C4=4*m2*C0/(1-(m2*m2));
21 printf("The elements of the terminating half section
    m-derived H.P.T. filter are:\n");
22 printf("  2C/m = %f microfarads\n",round(C3*(10^6)
    *100)/100);
23 printf("  L/m = %f mH\n",round(L2*(10^3)*100)/100);
24 printf("  4mC/(1-m^2) = %f microfarads\n\n",round(C4
    *(10^6)*100)/100);
25 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 pe

```

1 clear;
2 clc;
3 fc=1*(10^6);Rk=75;m=0.6;
4 L0=Rk/(4*pi*fc);
5 C0=1/(4*pi*fc*Rk);
6 printf("The elements of the prototype T-section H.P.
    are:\n");
7 printf("  L = %f mH\n",round(L0*(10^3)*1000)/1000);
8 printf("  C = %f picofarads\n\n",round(C0*(10^12)));
9 C1=2*C0/m;
10 L1=L0/m;
11 C2=4*m*C0/(1-(m*m));
12 printf(" The elements of the terminating half
    section m-derived H.P.T. filter are:\n");

```



```

13 printf(" 2C/m = %f picofarads\n",fix(C1*(10^12)));
14 printf(" L/m = %f mH\n",round(L1*(10^6))/1000);
15 printf(" Cshunt = %f picofarads\n\n",round(C2
    *(10^12)));
16 fi=fc*sqrt(1-(m*m));
17 printf(" Frequency of peak attenuation = %d kHz",fi
    *(10^-3));

```

Scilab code Exa 14.12 Designing a low pass composite filter

```

1 clear;
2 clc;
3 Rk=500;fc=1000;fi1=1065;fi2=1250;
4 L0=Rk/(%pi*fc);
5 C0=1/(%pi*fc*Rk);
6 printf("The elements of the constant-K L.P. are:\n")
    ;
7 printf(" L = %f mH\n",round(L0*(10^3)));
8 printf(" C = %f microfarads\n\n",round(C0*(10^6)
    *100)/100);
9 m1=0.4
10 L1=m1*L0/2;
11 L2=(1-(m1*m1))*L0/(4*m1);
12 C1=m1*C0;
13 printf("The elements of the m-derived L.P.T. filter
    are:\n");
14 printf(" mL/2 = %f mH\n",round(L1*(10^3)*10)/10);
15 printf(" mC = %f microfarads\n",round(C1*(10^6)
    *1000)/1000);
16 printf(" (1-m^2)L/4m = %f mH\n\n",fix(L2*(10^3)*10)
    /10);
17 m2=sqrt(1-((fc/fi2)^2));
18 L3=m2*L0/2;
19 L4=(1-(m2*m2))*L0/(4*m2);
20 C2=m2*C0;

```

```

21 printf("The elements of the terminating half
    sections m-derived L.P.T. filter are:\n");
22 printf("  mL/2 = %f mH\n",fix(L3*(10^3)*10)/10);
23 printf("  C = %f microfarads\n",round(C2*(10^6)
    *1000)/1000);
24 printf("  Lshunt = %f mH\n\n",fix(L4*(10^3)*10)/10);
25 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

```

1 clear;
2 clc;
3 f1=1000;f2=4000;Rk=600;
4 C1=(f2-f1)/(4*pi*Rk*f1*f2);
5 L1=Rk/(pi*(f2-f1));
6 C2=1/(pi*Rk*(f2-f1));
7 L2=Rk*(f2-f1)/(4*pi*f1*f2);
8 printf("The elements of the prototype band pass
    filter are:\n");
9 printf("  L1 = %f mH\n",fix(L1*(10^5))/100);
10 printf("  C1 = %f microfarads\n",round(C1*(10^11))
    /10^5);
11 printf("  L2 = %f mH\n",round(L2*(10^4))/10);
12 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

```

1 clear;
2 clc;

```

```

3 f1=12000;f2=16000;Rk=600;
4 C1=(f2-f1)/(4*pi*Rk*f1*f2);
5 L1=Rk/(pi*(f2-f1));
6 C2=1/(pi*Rk*(f2-f1));
7 L2=Rk*(f2-f1)/(4*pi*f1*f2);
8 printf("The elements of the prototype band pass
   filter are:\n");
9 printf("  L1 = %f mH\n",round(L1*(10^5))/100);
10 printf("  C1 = %f picofarads\n",round(C1*(10^12)));
11 printf("  L2 = %f mH\n",L2*(10^3));
12 printf("  C2 = %f microfarads",C2*(10^6));
13 //the difference in result of L2 and C2 is due to
   erroneous value in textbook.
14 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

```

1 clear;
2 clc;
3 fc=5000;fi=1.25*fc;Rk=600; //fi=frequency at which
   infinite attenuation occurs
4 L0=Rk/(pi*fc);
5 C0=1/(pi*fc*Rk);
6 m=sqrt(1-((fc/fi)^2));
7 L1=m*L0/2;
8 L2=(1-(m*m))*L0/(4*m);
9 C1=m*C0;
10 printf("The elements of the m-derived L.P.T. filter
   are:\n");
11 printf("  mL/2 = %f mH\n",round(L1*(10^5))/100);
12 printf("  mC = %f microfarads\n",fix(C1*(10^10))
   /10^4);
13 printf("  (1-m^2)L/4m = %f mH",round(L2*(10^4))/10);

```

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

```
1 clear;
2 clc;
3 L=0.5*(10^-3);C=0.01*(10^-6);Rk=600;
4 L1=2*L;
5 C1=C/2;
6 L2=Rk*Rk*C1;
7 C2=L1/(Rk*Rk);
8 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

```
1 clear;
2 clc;
3 fc=2400;fi=2500;Rk=600; //fi=frequency at which
    infinite attenuation occurs
4 L0=Rk/(3.14*fc);
5 C0=1/(3.14*fc*Rk);
6 printf("The elements of the constant-K L.P. are:\n")
    ;
7 printf("  L = %f mH\n",round(L0*(10^5))/100);
8 printf("  C = %f microfarads\n\n",round(C0*(10^10))
    /10^4);
9 m1=sqrt(1-((fc/fi)^2));
10 L1=m1*L0/2;
11 L2=(1-(m1*m1))*L0/(4*m1);
12 C1=m1*C0;
```

```

13 printf("The elements of the m-derived L.P.T. filter
    are:\n");
14 printf("  mL/2 = %f mH\n",round(L1*(10^5))/100);
15 printf("  mC = %f microfarads\n",round(C1*(10^10))
    /10^4);
16 printf("  (1-m^2)L/4m = %f mH\n\n",round(L2*(10^5))
    /100);
17 m2=0.6;
18 L3=m2*L0/2;
19 L4=(1-(m2*m2))*L0/(2*m2);
20 C2=m2*C0/2;
21 printf("The elements of the terminating half
    sections m-derived L.P.T. filter are:\n");
22 printf("  mL/2 = %f mH\n",round(L3*(10^5))/100);
23 printf("  mC/2 = %f microfarads\n",round(C2*(10^10))
    /10^4);
24 printf("  (1-m^2)L/2m = %f mH\n\n",fix(L4*(10^5))
    /100);
25 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

```

1 clear;
2 clc;
3 C0=1*(10^-6);L0=10*(10^-3);
4 C=C0/2;
5 L=L0;
6 Rk1=sqrt(L/C);
7 fc1=1/(4*pi*sqrt(L*C));
8 printf("The cut-off frequency (high pass) = %f Hz\n",
    round(fc1));
9 printf(" The characteristic impedance (high pass) =

```

```

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

```

Scilab code Exa 14.19 Designing a low pass composite filter

```

1 clear;
2 clc;
3 Rk=500;fc=4000;fi=5000; //fi=frequency at which
    infinite attenuation occurs
4 L0=Rk/(%pi*fc);
5 C0=1/(%pi*fc*Rk);
6 printf("The elements of the constant-K L.P. are:\n")
    ;
7 printf("  L = %f mH\n", round(L0*(10^4))/10);
8 printf("  C = %f microfarads\n\n", round(C0*(10^9))
    /1000);
9 m1=sqrt(1-((fc/fi)^2));
10 L1=m1*L0/2;
11 L2=(1-(m1*m1))*L0/(4*m1);
12 C1=m1*C0;
13 printf("The elements of the m-derived L.P.T. filter
    are:\n");
14 printf("  mL/2 = %f mH\n", round(L1*(10^5))/100);
15 printf("  C = %f microfarads\n", round(C1*(10^10))
    /10^4);
16 printf("  Lshunt = %f mH\n\n", round(L2*(10^5))/100);
17 m2=0.6;
18 L3=m2*L0/2;
19 L4=(1-(m2*m2))*L0/(2*m2);

```

```

20 C2=m2*C0/2;
21 printf("The elements of the terminating half
    sections m-derived L.P.T. filter are:\n");
22 printf("  Lseris = %f mH\n",round(L3*(10^5))/100);
23 printf("  C = %f microfarads\n",round(C2*(10^10))
    /10^4);
24 printf("  Lshunt = %f mH\n\n",round(L4*(10^5))/100);
25 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");
26 //the difference in result of the elements of the m
    derived L.P. T section is due to erroneous value
    in textbook.
27 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

```

1 clear;
2 clc;
3 fr=1*(10^6);C=0.04*(10^-12);C1=6*(10^-12);
4 fa=fr*(1+(C/(2*C1)));
5 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

```

1 clear;
2 clc;
3 Rk=600;f1=120*(10^3);f2=123*(10^3);
4 C1=(f2-f1)/(4*pi*Rk*f1*f2);

```

```

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

```

Scilab code Exa 14.22 Calculating bandwidth

```

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

```

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

```

1 clear;
2 clc;
3 Ro=600;f=120;fo=1500; //f=f2-f1
4 F=fo*fo; //F=f1*f2
5 l1=Ro/(2*%pi*f);
6 c1=round(f*10^11/(2*Ro*%pi*F))/10^5;
7 c2=round(1*10^8/(%pi*Ro*f))/100;
8 l2=Ro*f/(4*%pi*F);

```



```

9  n=(1/2)*(sqrt(((2*c2/(c1/2))+9)-1));
10 C1=((2*n)-1)*c1*10^-6/(2*n);
11 C2=c1*10^-6/n;
12 L=n*n*12;
13 C3=c2*10^-6/(n*n);
14 printf("The elements of the filter are:\n");
15 printf(" (2n-1)C1/n = %f microfarads\n",fix(C1
    *(10^9))/1000);
16 printf(" 2C1/n = %f microfarads\n",fix((C2)*(10^10))
    /10^4);
17 printf(" (n^2)L2 = %f H\n",round(L*10)/10);
18 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

```
1 clear;
2 clc;
3 R0=600;R1=400;L=40*(10^-3);
4 R2=R0*R0/R1;
5 printf("—Other arm of lattice equalizer will have a
   resistance of R2 = %f ohms\n",R2);
6 C=L/(R0*R0);
7 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

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

```

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

```

Scilab code Exa 15.6 Calculating characteristic impedance and the components of the

```

1 clear;
2 clc;
3 R1=1000;C1=0.0212*(10^-6);R2=250;
4 R0=sqrt(R1*R2);
5 printf(" Characteristic impedance of line = %f ohms\n
",R0);
6 L2=C1*R0*R0;
7 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

```

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

```

Scilab code Exa 15.11 Designing a constant resistance equalizer

```
1 clear;
2 clc;
3 f=500;Ro=400;f1=50;D1=17;
4 M1=10^(D1/10);
5 D2=4;f2=2500;
6 M2=10^(D2/10);
7 B=sqrt(((f1*f1*(M1-1))-(f2*f2*(M2-1)))/(M2-M1));
8 A=sqrt((B*B*M2)+(f2*f2*(M2-1)));
9 L11=Ro/(%pi*(A+B));
10 L12=Ro/(%pi*(A-B));
11 R11=L11*%pi*(A-B);
12 R12=L12*%pi*(A+B);
13 R21=Ro*Ro/R11;
14 R22=Ro*Ro/R12;
15 C21=L11/(Ro*Ro);
16 C22=L12/(Ro*Ro);
17 printf("The designed equalizer will have the
    configuration:\n");
18 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);
19 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)));
20 M=((A*A)+(f*f))/((B*B)+(f*f));
21 F=10*log10(M);
22 printf(" Loss aat 500Hz = %f db",fix(F*10)/10);
```

Scilab code Exa 15.12 Designing a constant resistance equalizer

```
1 clear;
2 clc;
3 R0=600;D=10;b=10/6;fr=8.5*(10^3);
```

```

4 k=round((10^(D/20)*100))/100;
5 fb=fr/b;
6 Cb=1/(2*pi*fb*R0);
7 Lb=R0/(2*pi*fb);
8 printf("The desired bridged-T constant resistance
    equalizer will be as:\n");
9 printf("-Ro = %f ohms\n",R0);
10 L1=Lb*((k-1)/sqrt(k))*((b*b)-1)/(b*b);
11 printf("-L1 = %f Henry\n",round(L1*10^4)/10^4);
12 C1=Cb*(sqrt(k)/(k-1))*(1/((b*b)-1));
13 printf("-C1 = %f microfarads\n",round(C1*(10^10))
    /10^4);
14 L2=Lb*(sqrt(k)/(k-1))*(1/((b*b)-1));
15 printf("-L2 = %f Henry\n",round(L2*10^5)/10^5);
16 C2=Cb*((k-1)/sqrt(k))*((b*b)-1)/(b*b);
17 printf("-C2 = %f microfarads\n",round(C2*(10^10))
    /10^4);
18 R1=R0*(k-1);
19 printf("-R1 = %f ohms\n",round(R1));
20 R2=R0/(k-1);
21 printf("-R2 = %f ohms",round(R2));

```

Scilab code Exa 15.13 Designing a lattice equalizer

```

1 clear;
2 clc;
3 f1=200;f2=2400;Li=2.6;Lc1=0.494;Lc2=1.949;Ro=600;
4 //Li=total insertion loss
5 //value of f1 as taken in solution
6 //Lc=cable loss
7 Le1=Li-Lc1; //Le=equalizer loss
8 Le2=Li-Lc2;
9 M1=fix(exp(2*Le1));
10 M2=round(exp(2*Le2)*100)/100;
11 Q=((f2*f2*(M2-1))-(f1*f1*(M1-1)))/(M1-M2);

```

```

12 P=(f1*f1*(M1-1))+(M1*Q);
13 R11b=Ro*(sqrt(P)+sqrt(Q))/(sqrt(P)-sqrt(Q));
14 R21b=Ro*Ro/R11b;
15 L12b=Ro/(%pi*(sqrt(P)-sqrt(Q)));
16 C22b=L12b/(Ro*Ro);
17 L12a=Ro/(%pi*(sqrt(P)+sqrt(Q)));
18 R11a=Ro*(sqrt(P)-sqrt(Q))/(sqrt(P)+sqrt(Q));
19 R21a=Ro*Ro/R11a;
20 C22a=L12a/(Ro*Ro);
21 printf("The required equalizer will have the
    following configuration:\n");
22 printf("I. R11 = %f ohms\n",fix(R11a));
23 printf("-L12 = %f mH\n",round(L12a*(10^4))/10);
24 printf("-R21 = %f ohms\n",round(R21a));
25 printf("-C22 = %f microfarads\n\n",round(C22a*(10^9)
    )/10^3);
26 printf("II. R11 = %f ohms\n",round(R11b));
27 printf("-L12 = %f mH\n",fix(L12b*(10^4))/10);
28 printf("-R21 = %f ohms\n",round(R21b*100)/100);
29 printf("-C22 = %f microfarads",fix(C22b*(10^9))
    /10^3);

```

Chapter 16

Attenuators

Scilab code Exa 16.1 Designing a symmetrical bridge T network

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

Scilab code Exa 16.2 Designing a T pad attenuator

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

```

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

```

Scilab code Exa 16.3 Designing a T type attenuator

```

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

```

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

```

1 clear;
2 clc;
3 D=20;Ro=500;
4 N=10^(D/20);
5 R1=Ro*(N-1)/(N+1);
6 R2=Ro*(N+1)/(N-1);
7 printf("The elements of the attenuator are:\n");
8 printf("R1 = %f ohms\n",round(R1*100)/100);
9 printf(" R2 = %f ohms\n\n",round(R2*100)/100);
10 r1=R1;
11 r2=(R2-R1)/2;
12 printf("The equivalent T structure of the designed
    lattice:\n");
13 printf(" R1 = %f ohms\n",round(r1*100)/100);
14 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 re

```
1 clear;
2 clc;
3 D=15;Ro=75;
4 N=10^(D/20);
5 R1=Ro*(N-1)/(N+1);
6 R2=Ro*2*N/((N*N)-1);
7 printf("The arms of the T section will contain:\n");
8 printf(" R1 = %f ohms\n",round(R1*100)/100);
9 printf(" R2 = %f ohms\n",round(R2*100)/100);
10 //the difference in result of R1 is due to erroneous
    value in textbook.
11 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

```
1 clear;
2 clc;
3 R1=175;R2=350;
4 y=poly([1 -3 1],"N","coeff");
5 f=roots(y);
6 N=f(1,1);
7 D=20*log10(N);
8 Ro=R1*(N+1)/(N-1);
9 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

```
1 clear;
2 clc;
3 D=40; Ri1=70; Ri2=600;
4 Ai=D*0.115;
5 R3=(sqrt(Ri1*Ri2))/sinh(Ai);
6 R2=(Ri2/tanh(Ai))-R3;
7 R1=(Ri1/tanh(Ai))-R3;
8 printf("The desired elements of T-pad are:\n");
9 printf(" R1 = %f ohms\n",round(R1*100)/100);
10 printf(" R2 = %f ohms\n",round(R2));
11 printf(" R3 = %f ohms\n",round(R3*100)/100);
```

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

```
1 clear;
2 clc;
3 D=10; Ri1=150; Ri2=75;
4 Ai=D*0.115;
5 R3=(sqrt(Ri1*Ri2))/sinh(Ai);
6 R2=(Ri2/tanh(Ai))-R3;
7 R1=(Ri1/tanh(Ai))-R3;
8 printf("The desired elements of T-pad are:\n");
9 printf(" R1 = %f ohms\n",round(R1*100)/100);
10 printf(" R2 = %f ohms\n",round(R2*10)/10);
11 printf(" R3= %f ohms\n\n",fix(R3*100)/100);
12 R2a=sqrt((Ri1*Ri2*Ri2)/(Ri1-Ri2));
13 R1a=sqrt(Ri1*(Ri1-Ri2));
14 printf("The minimum loss pad will be a L attenuator
: \n");
15 printf(" R1 = %f ohms\n",round(R1a));
16 printf(" R2 = %f ohms\n",round(R2a));
```

Scilab code Exa 16.9 Designing L type attenuator

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

Scilab code Exa 16.10 Designing a T pad

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

Scilab code Exa 16.11 Designing an unbalances pi attenuator

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

Scilab code Exa 16.12 Designing a T attenuator

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

```

10 printf("The desired elements of the attenuator are:\n");
11 printf("R1 = %f ohms\n",fix(R1*100)/100);
12 printf("R2 = %f ohms\n",fix(R2*100)/100);
13 printf("R3 = %f ohms\n",fix(R3*100)/100);

```

Scilab code Exa 16.13 Designing a ladder attenuator

```

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

```

Scilab code Exa 16.14 Designing an attenuator

```

1 clear;
2 clc;
3 d=12;Ro=500;
4 D=(d-3)/3;
5 N=round(10^(D/20)*1000)/1000;
6 R1=Ro*(N+1)/(N-1);
7 R2=Ro*(round((N*N))-1)/(2*N);
8 printf("The desired attenuator will be:\n");
9 printf("R1 = %f ohms\n",round(R1));

```

```

10 printf("R2 = %f ohms\n",round(R2));
11 //the difference in result of R1 is due to erroneous
    value in textbook.
12 disp("The difference in result of R1 is due to
    erroneous value in textbook")

```

Scilab code Exa 16.15 Designing a balanced attenuator

```

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

```

Scilab code Exa 16.16 Finding the attenuation and the values of the shunt resistances

```

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

```

Scilab code Exa 16.17 Designing a L type attenuator

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

Scilab code Exa 16.18 Designing a minimum loss pad

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