Scilab Textbook Companion for Antennas And Wave Propagation by J. D. Kraus, R. J. Marhefka And A. S. Khan[1]

Created by
Aditya Rutwik
Bachelor Of Technology
Electronics Engineering
CMR Institute of Technology
College Teacher
None
Cross-Checked by
None

July 31, 2019

---

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

Book Description

Title: Antennas And Wave Propagation

Author: J. D. Kraus, R. J. Marhefka And A. S. Khan

Publisher: TataMcGrawHill Education Private Limited New Delhi

Edition: 4

Year: 2012

ISBN: 0070671559

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes

Chapter 2

Antenna Basics

Scilab code Exa 2.1 1

```
//Chapter 2: Antenna Basics
//Example 2-3.1
clc;

//Variable Initialization
e_half_power = 1/sqrt(2)    //E(theta) at half power (relative quantity)

//Calculation
theta = acos(sqrt(e_half_power)) // theta (radians)
hpbw = 2*theta*180/%pi      // Half power beamwidth (degrees)

//Result
mprintf(”The Half Power Beamwidth is %.0f degrees”, hpbw)
```

Scilab code Exa 2.2 2

```
//Chapter 2: Antenna Basics
//Example 2-3.2
clc;

//Variable Initialization
e_half_power = 1/sqrt(2)   //E(theta) at half power(unitless)
e_null = 0       //E(theta) = 0 at null points (unitless)
theta_1 = 0      //theta' (degrees)
theta = 1        //theta (degrees)

//Calculation
for x=0:2       //loop untill theta = i
theta = 0.5*acos(e_half_power/cos(theta_1*%pi/180))            //theta(radian)
theta_1 = theta*180/%pi        //theta(degrees)
end

hpbw = 2*(theta*180/%pi)     //Half-power beamwidth (Degrees)
theta = 0.5*acos(e_null)   //theta (radians)
fnbw = 2*(theta*180/%pi)     //Beamwidth between first null (degrees)

//Result
mprintf("The half power beamwidth is %.2f degrees",hpbw)
mprintf("\nThe beamwidth between first nulls is %d degrees", fnbw)
```

Scilab code Exa 2.3 3

```
//Chapter 2: Antenna Basics
//Example 2-4.1
```

```
clc;

//Calculation
deff(’y=f(x)’,’y=sin(x)’)        //sin(theta)
omega=intg(20*%pi/180,40*%pi/180,f)
omega1=omega*(180/%pi)
deff(’y=f1(x)’,’y=1’)
omega2=intg(30,70,f1)
omega_f=omega2*omega1            //omega (square degrees)

//Result
mprintf(”The solid angle, omega is %.0f square degrees”,omega_f)
```

Scilab code Exa 2.4 4

```
//Chapter 2: Antenna Basics
//Example 2-4.2
clc;
clear;

//Calculation
deff(’z=f(x,y)’,’z=(cos(x)**4)*sin(x)*1’)//Integration Function
X=[0 0;%pi/2 %pi/2;%pi/2 0];
Y=[0 0;2*%pi 2*%pi;0 2*%pi];
[I,err]=int2d(X,Y,f)//Beam area (steradians)

//Result
mprintf(’The Beam Area of the given pattern is %.2f sr’,I)
```

Scilab code Exa 2.5 5

```
//Chapter 2: Antenna Basics
//Example 2-7.1
clc;
clear;

//Variable declaration
n = 10      //Number of isotropic point sources
dr = %pi/2     //Distance(radians)
hpbw = 40     //Half power beamwidth (degrees)

//Calculation
deff('z=f(x,y)','z=(sin(%pi/20)*(sin((%pi/2)*(5*cos(y)-6))/sin((%pi/20)*(5*cos(y)-6))))**2')
X=[0 0;%pi/2 %pi/2;%pi/2 0];
Y=[0 0;2*%pi 2*%pi;0 2*%pi];
[g1,err]=int2d(X,Y,f)
gain = (4*%pi)/g1     //Gain (unitless)
gain_db = 10*log10(gain)//Gain (dB)
gain_hpbw = 40000/(hpbw**2)     //Gain from approx. equation (unitless)
gain_hpbw_db = 10*log10(gain_hpbw)     //Gain from approx. equation (dB)
gain_diff = gain_hpbw_db - gain_db     //Difference in gain (dB)

//Result
mprintf("The Gain G is %.2f dB",gain_db)
mprintf("\nThe Gain from approx. equation is %.0f dB",gain_hpbw_db)
mprintf("\nThe Difference is %.2f dB",gain_diff)

//An error arises due to incorrect integration of the normalized power pattern
//Subsequently, the difference in gain is varying
```

Scilab code Exa 2.6 6

```
//Chapter 2: Antenna Basics
//Example 2-7.2
clc;
clear;

//Variable Initialization
theta_hp = 90
phi_hp = 90

//Calculation
X=[0 0;%pi %pi;%pi 0];
Y=[0 0;0 %pi;%pi %pi];
function z = f(x,y), z=sin(x)*sin(x)*sin(x)*sin(y)*sin(y),endfunction
[I,err]=int2d(X,Y,f)                    //Exact Directivity(No unit)
direct_e=4*%pi/I          //Exact Directivity(Unitless)
direct_apprx=41253.0/(theta_hp * phi_hp)   //Approximate Directivity(Unitless)
db_diff=10*log10(direct_e/direct_apprx)   //Difference(dB)

//Result
mprintf("The exact directivity is %.1f",direct_e)
mprintf("\nThe approximate directivity is %.1f",direct_apprx)
mprintf("\nThe decibel difference is %.1f dB",db_diff)
```

Scilab code Exa 2.7 7

```
//Chapter 2: Antenna Basics
//Example 2-10.1
clc;

//Variable Initialization
Z = 120*%pi                        //Intrinsic impedance of free space (ohm)

//Calculation
max_aper = Z/(320*%pi**2)    //Max. effective aperture (lambda squared)
direct = 4*%pi*max_aper      //Directivity (unitless)

//Result
mprintf("The Maximum effective aperture is %.3f lambda square",max_aper)
mprintf("\nThe Directivity is %.1f", direct)
```

Scilab code Exa 2.8 8

```
//Chapter 2: Antenna Basics
//Example 2-10.2
clc;

//Variable Initialization
R_r = 73                                  //Radiation resistance (ohm)

//Calculation
eff_aper = 30/(R_r*%pi)          //Effective aperture (lambda squared)
directivity = 4*%pi*eff_aper     //Directivity (unitless)
```

```

//Result
mprintf("The effective aperture is %.2f lambda square",eff_aper)
mprintf("\nThe directivity is %.2f",directivity)
```

Scilab code Exa 2.9 9

```
//Chapter 2: Antenna Basics
//Example 2-11.1
clc;

//Variable Initialization
P_t = 15          //Transmitter power (W)
A_et = 2.5        //Effective aperture of transmitter (meter^2)
A_er = 0.5        //Effective aperture of receiver (meter^2)
r = 15e3          //Distance between the antennas (Line of sight) (m)
frequency  = 5e9      //Frequency (Hz)
c = 3e8           //Speed of light (m/s)

//Calculation
wave_len = c/frequency                                    //Wavelength (m)
P_r = (P_t*A_et*A_er)/((r**2)*(wave_len**2))     //Received power (W)

//Result
mprintf("The power delivered to the receiver is %.2e watts",P_r)
```

Scilab code Exa 2.10 10

```
//Chapter 2: Antenna Basics
//Example 2-16.1
clc;

//Variable Initialization
E1 = 3      //Magnitude of electric field in x direction (V/m)
E2 = 6      //Magnitude of electric field in y direction (V/m)
Z = 377     //Intrinsic impedance of free space (ohm)

//Calculation
avg_power = 0.5*(E1**2 + E2**2)/Z       //Average power per unit area (W/m^2)

//Result
disp(avg_power,”The average power per unit area in watts/meter square”)
```

Scilab code Exa 2.11 11

```
//Chapter 2: Antenna Basics
//Example 2-17.1
clc;

//Variable Initialization
AR_w = 4          //Axial Ratio for left elliptically polarized wave (unitless)
tau_w = 15        //Tilt angle for left elliptically polarized wave (degrees)
AR_a = -2         //Axial Ratio for right elliptically polarized wave (unitless)
```

```
tau_a = 45         //Tilt angle for right
   elliptically polarized wave (degrees)
tau_w2 = 20.7      //2*Tilt angle for left
   elliptically polarized wave (degrees)
tau_a2 = 39.3      //2*Tilt angle for right
   elliptically polarized wave (degrees)

//Calculation
eps_a2 = 2*atan(1,AR_a)*180/%pi  //Polarisation
   latitude (degrees)
eps_w2 = 2*atan(1,AR_w)*180/%pi   //Antenna latitude
    (degrees)
gamma_w2 =acos(cos(eps_w2*%pi/180)*cos(tau_w2*%pi
   /180))       //great-circle angle - antenna (
   radians)
gamma_a2 =acos(cos(eps_a2*%pi/180)*cos(tau_a2*%pi
   /180))  //great-circle angle - wave (radians)
M_Ma = (gamma_w2*180/%pi) + (gamma_a2*180/%pi)
              //total great-circle angle (degrees)
F = cos((M_Ma/2)*%pi/180)**2        //Polarisation
   matching factor (relative quantity)

//Result
mprintf("The polarization matching factor is %.2f",F
   )
```

Chapter 3

The Antenna Family

Scilab code Exa 3.1 1

```
//Chapter 3: The Antenna Family
//Example 3-3.2
clc;

//Variable Initialization
Z_0 = 377   //Intrinsic impedance of free space(ohm)
Z_d = 710 +%i   //Terminal impedance of dipole
    cylinder (ohm)

//Calculation
Z_s = (Z_0**2)/(4*Z_d)  //Terminal impedance of the
    slot (ohm)

//Result
mprintf("The terminal impedance of the slot is %d
    ohms",Z_s)
```

Scilab code Exa 3.2 2

```
//Chapter 3: The Antenna Family
//Example 3-6.1
clc;

//Variable Initialization
L = 10              //Horn length (lambda)
delta = 0.25        //Path length difference (lambda)

//Calculation
theta = 2*acos(L/(L+delta))     //Horn flare angle (radians)
theta = theta*180/%pi            //Horn flare angle (degrees)

//Result
mprintf("The largest flare angle for given delta is %.1f degrees",theta)
```

Scilab code Exa 3.3 3

```
//Chapter 3: The Antenna Family
//Example 3-7.1
clc;

//Variable Initialization
f = 599e6      //Frequency of TV Station (Hz)
E = 1e-6       //Field strength (V/m)
D = 20         //Diameter of antenna (m)
c = 3e8        //Speed of light (m/s)
Z_0 = 377      //Intrinsic impedence of free space (ohm)

//Calculation
wave_lt = c/f                       //Wavelength (m)
```

```
A_e = (D*(wave_lt**2))/(4*%pi)   //Effective aperture (m^2)
P_r = (E**2)*A_e/Z_0                       //Received power (W)

//Result
mprintf("The received power is %.2e W", P_r)
```

Scilab code Exa 3.4 4

```
//Chapter 3: The Antenna Family
//Example 3-11.1
clc;

//Variable Initialization
n = 4                    //Number of patch antennas (lambda)
diameter = 0.5           //Diameter of patch antennas (lambda)

//Calculation
A_e = n*diameter                 //Effective aperture (lambda^2)
D = (4*%pi*A_e)            //Directivity (unitless)
D_dbi = 10*log10(D)      //Directivity (dBi)
ohm_a = (4*%pi)/D          //Beam area (steradians)

//Result
mprintf("The directivity is %d or %d dBi",D,D_dbi)
mprintf("\nThe beam area is %.1f sr", ohm_a)
```

Chapter 4

Radiation

Scilab code Exa 4.1 1

```
//Chapter 4: Radiation
//Example 4-4.1
clc;

//Variable Initialization
theta = 30                  //Angle of radiation (degrees)
epsilon_0 = 8.854e-12       //Permittivity of free space (F/m)
I_dl = 10                   //Current in length dl (A-m)
r = 100e3                   //Distance of point from origin (m)

//Calculation
E_mag = (I_dl*sin(theta*%pi/180))/(4*%pi*epsilon_0)                              //Magnitude of Electric field vector (V/m)
H_mag = (I_dl*sin(theta*%pi/180))/(4)                              //Magnitude of Magnetic field vector (T)

```

```
//Result
disp(E_mag,"The magnitude of E vector in V/m ")
mprintf("\nThe magnitude of H vector is %.3f /pi T",
    H_mag)
```

Scilab code Exa 4.2 2

```
//Chapter 4: Radiation
//Example 4-4.2
clc;

//Variable Initialization
v = 3e8           //Speed of light(m/s)
f = 10e6          //Frequency (Hz)

//Calculation
w = 2*%pi*f       //Angular frequency(rad/s)
r = v/w               //Distance (m)

//Result
mprintf("The distance for the specified condition is
    %.2f m",r)
```

Scilab code Exa 4.3 3

```
//Chapter 4: Radiation
//Example 4-4.3
clc;

//Variable Initialization
c = 3e8               //Speed of light (m/s)
f = 3e9               //Frequency (Hz)

```

```
//Calculation
v = 0.6*c           //60% of velocity of light (m/s)
w = 2*%pi*f     //Angular frequency (rad/s)
r = v/w             //Distance (m)

//Result
mprintf(”The distance for the specified condition is %.6f m”, r)
```

Scilab code Exa 4.4 4

```
//Chapter 4: Radiation
//Example 4-5.1
clc;

//Variable Initialization
dl = 1e-2       //Length of radiating element (m)
I_eff = 0.5     //Effective current (A)
f = 3e9         //Frequency (Hz)
c = 3e8         //Velocity of light (m/s)

//Calculation
w = 2*%pi*f     //Angular Frequency (rad/s)
P = 20*(w**2)*(I_eff**2)*(dl**2)/(c**2)     //Radiated power (W)

//Result
mprintf(”The radiated power is %.2f W”, P)

//The answer obtained is varying compared with the textbook answer because of a calculation error
```

Scilab code Exa 4.5 5

```
//Chapter 4: Radiation
//Example 4-5.2
clc;

//Variable Initialization
L = 5           //Length of radiating element (m)
f1 = 30e3       //Frequency (Hz)
f2 = 30e6       //Frequency (Hz)
f3 = 15e6       //Frequency (Hz)
c = 3e8         //Velocity of light (m/s)

//Calculation
wave_lt1 = c/f1                 //Wavelength (m)
wave_lt1 = wave_lt1 /10
R_r1 = 800*(L/wave_lt1)**2      //Radiation resistance (ohm)

wave_lt2 = c/f2                 //Wavelength (m)
L = wave_lt2/2                  //Effective length (m)
R_r2 = 200*(L/wave_lt2)**2      //Radiation resistance (ohm)

wave_lt3 = c/f3                 //Wavelength (m)
L = wave_lt3/4                  //Effective length (m)
R_r3 = 400*(L/wave_lt3)**2      //Radiation resistance (ohm)

//Result
mprintf("The radiation resistance for f1 is %.2f ohms", R_r1)
mprintf("\nThe radiation resistance for f2 is %d ohms",R_r2)
mprintf("\nThe radiation resistance for f3 is %d ohms",R_r3)
```

Scilab code Exa 4.6 6

```
//Chapter 4: Radiation
//Example 4-6.1
clc;

//Variable Initialization
Im = 5                //Maximum current (A)
r = 1e3               //Distance (km)
eta = 120*%pi    //Intrinsic impedance (ohm)
theta = 60*%pi/180          //Angle of radiation (radians)

//Calculation
sin2 = sin(theta)**2        //Sine squared theta (unitless)
P_av = (eta*(Im**2))/(8*(%pi**2)*(r**2))
P_av = P_av*(cos(%pi/2*cos(theta))**2)/(sin2)
                         //Average power (W)

//Result
mprintf("The average power available at 1km distance is %e W",P_av)
```

Chapter 5

Point Sources and their Arrays

Scilab code Exa 5.1 1

```
//Chapter 5: Point Source and Their Arrays
//Example 5-6.1
clc;
clear;

//Variable Initialization
Um=1 //Maximum radiation intensity (unitless)

//Calculation
deff('z=f(x,y)','z=cos(x)*sin(x)')//Integrand(
    Unitless)
X=[0 0;%pi/2 %pi/2;%pi/2 0];
Y=[0 0;2*%pi 2*%pi;0 2*%pi];
[I,err]=int2d(X,Y,f)//Total power radiated (relative
     to Um)

D=(4*%pi)/I //Directivity (unitless)

mprintf('The directivity is %f',D)
```

Scilab code Exa 5.2 2

```
//Chapter 5: Point Source and Their Arrays
//Example 5-6.2
clc;
clear;

//Variable Initialization
Um=1 //Maximum radiation intensity (unitless)
deff('z=f(x,y)','z=cos(x)*sin(x)')//Integrand(
   Unitless)
X=[0 0;%pi/2 %pi/2;%pi/2 0];
Y=[0 0;2*%pi 2*%pi;0 2*%pi];
[I,err]=int2d(X,Y,f)//Total power radiated (relative
    to Um)

D=(4*%pi)/(2*I) //Directivity (unitless)

mprintf('The directivity is %f',D)
```

Scilab code Exa 5.3 3

```
//Chapter 5: Point Source and Their Arrays
//Example 5-6.3
clc;
clear;

//Variable Initialization
Um=1 //Maximum radiation intensity (unitless)
deff('z=f(x,y)','z=sin(x)**2')//Integrand(Unitless)
X=[0 0;%pi %pi;%pi 0];
Y=[0 0;2*%pi 2*%pi;0 2*%pi];
```

```
[I,err]=int2d(X,Y,f)//Total power radiated (relative
    to Um)

D=(4*%pi)/I //Directivity (unitless)

mprintf('The directivity is %.3f',D)
```

Scilab code Exa 5.4 4

```
//Chapter 5: Point Source and Their Arrays
//Example 5-6.4
clc;
clear;

//Variable Initialization
Um=1 //Maximum radiation intensity (unitless)
deff('z=f(x,y)','z=sin(x)**3')//Integrand(Unitless)
X=[0 0;%pi %pi;%pi 0];
Y=[0 0;2*%pi 2*%pi;0 2*%pi];
[I,err]=int2d(X,Y,f)//Total power radiated (relative
    to Um)

D=(4*%pi)/I //Directivity (unitless)

mprintf('The directivity is %.1f',D)
```

Scilab code Exa 5.5 5

```
//Chapter 5: Point Source and Their Arrays
//Example 5-6.5
clc;
clear;

```

```
//Variable Initialization
Um=1 //Maximum radiation intensity (unitless)
deff('z=f(x,y)','z=sin(x)*cos(x)**2')//Integrand(Unitless)
X=[0 0;%pi/2 %pi/2;%pi/2 0];
Y=[0 0;2*%pi 2*%pi;0 2*%pi];
[I,err]=int2d(X,Y,f)//Total power radiated (relative to Um)

D=(4*%pi)/I //Directivity (unitless)

mprintf('The directivity is %.1f',D)
```

Scilab code Exa 5.6 6

```
//Chapter 5: Point Source and Their Arrays
//Example 5-6.6
clc;

//Variable Initialization
lobes = [0.25,0.37,0.46,0.12,0.07]   //Normalized power of lobes (unitless)

//Calculation
ohm_a = 0                      //Beam area (sr)
sum_lobes = 0                  //Sum of all lobes (unitless)
for i=lobes
    ohm_a =ohm_a + 2*%pi*(%pi/36)*(i)
    sum_lobes =sum_lobes + i
end
D = 4*%pi/ohm_a           //Directivity (unitless)
D_db = 10*log10(D)      //Directivity (in dBi)
e_m = lobes(1)/sum_lobes     //Beam efficiency (unitless)
```

```

//Result
mprintf("The directivity is %d or %.1f dBi",D,D_db)
mprintf("\nThe beam efficiency is %.1f",e_m)
```

Scilab code Exa 5.7 7

```
//Chapter 5: Point Source and Their Arrays
//Example 5-21.1
clc;

//Variable Initialization
a = 25                  //Height of vertical conducting
    wall (m)
r = 100                 //Distance to the receiver (m)
wave_lt = 10e-2         //Transmitter dimension (m)

//Calculation
k = sqrt(2/(r*wave_lt))     //Constant (unitless)
S_av = (r*wave_lt)/(4*(%pi**2)*(a**2))   //Relative
    signal level (unitless)
S_av_db = 10*log10(S_av)    //Signal level (in db)

//Result
mprintf("The signal level at the receiver is %.5f or
    %.0f dB",S_av,S_av_db)
```

Chapter 6

Electric Dipoles Thin Linear Antennas and Arrays of Dipoles and Apertures

Scilab code Exa 6.1 1

```
//Chapter 6: Electric Dipoles, Thin Linear Antennas and Arrays of Dipoles and Apertures
//Example 6-8.1
clc;

//Variable Initialization
z = 333.0          //Driving point impedance (ohm)
r = 300.0          //Twin-line impedance (ohm)
z1 = 73.0          //Self impedance of lambda/2 dipole (ohm)
z2 = 13.0          //Mutual impedance with lambda/2 spacing (ohm)

//Calculation
pv = (z-r)/(z+r)     //Reflection coefficient (unitless)
vswr = (1+pv)/(1-pv)     //Voltage Standing Wave
```

```
Ratio (unitless)
gain_l2 =sqrt((2*z1)/(z1-z2)) //Field gain over lambda/2 dipole (unitless)
gain_l2_db = 20*log10(gain_l2) //Field gain (in dB)
gain_iso = (gain_l2**2)*1.64         //Gain over isotropic source (unitless)
gain_iso_db = 10*log10(gain_iso) //Gain over isotropic source (in dB)

//Result
mprintf("The VSWR is %.2f", vswr)
mprintf("\nThe field gain over lambda/2 dipole is %.2f or %.1f dB",gain_l2,gain_l2_db)
mprintf("\nThe gain over isotropic source is %.1f or %.1f dB",gain_iso,gain_iso_db)
```

Scilab code Exa 6.2 2

```
//Chapter 6: Electric Dipoles, Thin Linear Antennas and Arrays of Dipoles and Apertures
//Example 6-8.2
clc;

//Variable Initialization
z = 73.0        //Self impedance of lambda/2 dipole (ohm)
zm = 64.4       //Mutual impedance with lambda/8 spacing (ohm)

//Calculation
D = sqrt((2*z)/(z-zm))*sin(%pi/8) //Field gain over lambda/2 dipole (unitless)
D_db = 20*log10(D)      //Field gain over lambda/2 dipole (in dB)

```

```
gain_iso = (D**2)*1.64       //Gain over isotropic source (unitless)
gain_iso_db = 10*log10(gain_iso)    //Gain over isotropic source (in dB)

//Result
mprintf("The field gain over lambda/2 dipole is %.2f or %.2f dB",D,D_db)
mprintf("\nThe gain over isotropic source is %.2f or %.1f dB", gain_iso,gain_iso_db)
```

Scilab code Exa 6.3 3

```
//Chapter 6: Electric Dipoles, Thin Linear Antennas and Arrays of Dipoles and Apertures
//Example 6-12.1
clc;

//Variable Initialization
s1 = 0.4                //Spacing 1(lambda)
s2 = 0.5                //Spacing 2(lambda)
s3 = 0.6                //Spacing 3(lambda)
R_21_1 = 6.3            //Mutual resistance for s1 (ohm)
R_21_2 = -12.691        //Mutual resistance for s2 (ohm)
R_21_3 = -23.381        //Mutual resistance for s3 (ohm)
Z = 73.13               //Self impedance of lambda/2 dipole (ohm)

//Calculation
gain_1 = sqrt(2*(Z/(Z+R_21_1)))      //Gain in field for s1 (unitless)
gain_iso1 = 1.64*(gain_1**2)         //Power gain over isotropic (unitless)
gain_iso_db1 = 10*log10(gain_iso1)   //Power gain (in dBi)
```

```

gain_2 = sqrt(2*(Z/(Z+R_21_2)))      //Gain in field for s2 (unitless)
gain_iso2 = 1.64*(gain_2**2)         //Power gain over isotropic (unitless)
gain_iso_db2 = 10*log10(gain_iso2)  //Power gain (in dBi)

gain_3 = sqrt(2*(Z/(Z+R_21_3)))      //Gain in field for s3 (unitless)
gain_iso3 = 1.64*(gain_3**2)         //Power gain over isotropic (unitless)
gain_iso_db3 = 10*log10(gain_iso3)  //Power gain (in dBi)

//Result
mprintf("The gain in field over half wave antenna for s1 is %.2f",gain_1)
mprintf("\nThe power gain over isotropic for s1 is %.2f or %.1f dBi",gain_iso1,gain_iso_db1)

mprintf("\n\nThe gain in field over half wave antenna for s2 is %.2f",gain_2)
mprintf("\nThe power gain over isotropic for s2 is %.2f or %.2f dBi ", gain_iso2,gain_iso_db2)

mprintf("\n\nThe gain in field over half wave antenna for s3 is %.2f",gain_3)
mprintf("\nThe power gain over isotropic for s3 is %.2f or %.2f dBi ",gain_iso3,gain_iso_db3)
```

Chapter 7

Loop Slot and Horn Antennas

Scilab code Exa 7.1 1

```
//Chapter 7: Loop, Slot and Horn Antennas
//Example 7-8.1
clc;

//Variable Initialization
C_lambda = 0.1*%pi          //Circumference (lambda)
R_m = 1.6                   //Mutual resistance of two
   loops (ohm)
theta1 = 90*%pi/180               //Angle of radiation
    (radians)
theta2 = 2*%pi/10                //Angle of radiation (
   radians)

//Calculation
Rr = 197*(C_lambda)**4       //Self resistance of
   loop (ohm)
D1 = (1.5)*(sin(theta1))**2 //Directivity of loop
   alone (unitless)
D1_db = 10*log10(D1)        //Directivity of loop
   alone (dBi)
D2 = 1.5*(2*sqrt(Rr/(Rr-R_m))*sin(theta2))**2
```

```
                              //Directivity of loop with ground plane (unitless)
D2_db = 10*log10(D2)     //Directivity of loop with ground plane (dBi)

//Result
mprintf("The directivity of loop alone is %.2f or %.2f dBi",D1,D1_db)
mprintf("\nThe directivity of loop with ground plane is %.2f or %.0f dBi",D2,D2_db)
```

Scilab code Exa 7.2 2

```
//Chapter 7: Loop, Slot and Horn Antennas
//Example 7-8.2
clc;

//Variable Initialization
Rr = 197.0          //Self resistance of loop (ohm)
Rm = 157.0         //Mutual resistance of two loops (ohm)
theta = 2*%pi/10 //Angle of radiation (radians)

//Calculation
D = 1.5*(2*sqrt(Rr/(Rr-Rm))*sin(theta))**2  //Directivity (unitless)
D_db = 10*log10(D)      //Directivity (dBi)

//Result
mprintf("The directivity is %.1f or %.1f dBi",D,D_db)
```

Scilab code Exa 7.3 3

```
//Chapter 7: Loop, Slot and Horn Antennas
//Example 7-11.1
clc;

//Variable Initialization
c =%pi      //Circumference (m)
f1 = 1     //Frequency (MHz)
f2 = 10    //Frequency (MHz)
d = 10e-3    //Diameter of copper wire (m)

//Calculation
RL_Rr1 = 3430/((c**3)*(f1**3.5)*d)
RL_Rr2 = 3430/((c**3)*(f2**3.5)*d)            //Ratio of Loss resistance and radiation resistance (unitless)
k1 = 1/(1+RL_Rr1)    //Radiation efficiency (unitless)
k_db1 = 10*log10(k1)     //Radiation efficiency (in dB)
k2 = 1/(1+RL_Rr2)    //Radiation efficiency (unitless)
k_db2 = 10*log10(k2)     //Radiation efficiency (in dB)

//Result
mprintf("The radiation efficiency for 1 MHz is %.1ef or %.1f dB",k1, k_db1)
mprintf("\nThe radiation efficiency for 10 MHz is %.2f or %.1f dB",k2, k_db2)
```

Scilab code Exa 7.4 4

```
//Chapter 7: Loop, Slot and Horn Antennas
//Example 7-11.2
clc;
```

```

//Variable Initialization
n = 10    //Number of turns (unitless)
dia = 1e-3   //Diameter of copper wire (m)
dia_rod = 1e-2    //Diameter of ferrite rod (m)
len_rod = 10e-2    //Length of ferrite rod (m)
mu_r = 250 - 2.5*%i     //Relative permeability (unitless)
mu_er = 50     //Effective relative permeability (unitless)
f = 1e6    //Frequency (Hz)
c = 3e8    //Speed of light (m/s)
mu_0 = %pi*4e-7    //Absolute permeability (H/m)

//Calculations
wave_lt = c/f    //Wavelength (m)
radius = dia_rod/2
C_l = (2*%pi*radius)/(wave_lt)    //Circumference of loop (m)
Rr = 197*(mu_er**2)*(n**2)*(C_l**4)    //Radiation resistance (ohm)
Rf = 2*%pi*f*mu_er*(imag(mu_r)/real(mu_r))*mu_0*(n**2)*(%pi*radius**2)/len_rod    //Loss resistance (ohm)
conduc = 1/((7e-5**2)*f*%pi*mu_er)    //Conductivity (S/m)
delta = 1/(sqrt(f*%pi*mu_er*conduc))    //Depth of penetration(m)

RL = n*(C_l/dia)*sqrt((f*mu_0)/(%pi*conduc))    //Ohmic resistance (ohm)
k = Rr/(RL+abs(Rf))    //Radiation efficiency (unitless)

L = mu_er*(n**2)*(radius**2)*mu_0/len_rod    //Inductance (H)
Q = 2*%pi*f*L/(abs(Rf) + Rr + RL)    //Ratio of energy stored to energy lost per cycle (unitless)
```

```

fHP = f/Q    //Bandwidth at half power (Hz)


//Results
mprintf(”The radiation efficiency is %.2e”,k)
mprintf(”\nThe value of Q is %.3f”,Q)
mprintf(”\nThe half−power bandwidth is %d Hz”,fHP)
```

Scilab code Exa 7.5 5

```
//Chapter 7: Loop, Slot and Horn Antennas
//Example 7−17.1
clc;

//Variable Initialization
Z0 = 376.7       //Intrinsic impedance of free space
    (ohm)
Zd = 73 + 42.5*%i //Impedance of infinitely small
    thin lambda/2 antenna (ohm)

//Calculation
Z1 = (Z0**2)/(4*Zd) //Terminal impedance of the
    lambda/2 slot antenna (ohm)

//Result
mprintf(”The terminal impedance of the thin lambda/2
     slot antenna is %.0f%dj ohm”,real(Z1),imag(Z1))
```

Scilab code Exa 7.6 6

```
//Chapter 7: Loop, Slot and Horn Antennas
//Example 7−17.2
```

```
clc;

//Variable Initialization
Zd = 67      //Terminal impedance of cylindrical
   antenna (ohm)
Z0 = 376.7   //Intrinsic impedance of free space (ohm
   )
L = 0.475    //Length of complementary slot (lambda)

//Calculation
Z1 = Z0**2/(4*Zd)    //Terminal resistance of
   complementary slot (ohm)
w = 2*L/100          //Width of complementary slot (
   lambda)

//Result
mprintf("The terminal resistance of the
   complementary slot is %d ohm",Z1)
mprintf("\nThe width of the complementary slot is %
   .4f lambda", w)
```

Scilab code Exa 7.7 7

```
//Chapter 7: Loop, Slot and Horn Antennas
//Example 7-17.3
clc;

//Variable Initialization
Zd = 710     //Terminal impedance of cylindrical
   dipole
Z0 = 376.7   //Intrinsic impedance of free space (ohm
   )

//Calculation
Z1 = Z0**2/(4*Zd)    //Terminal resistance of
```

```
complementary slot (ohm)

//Result
mprintf("The terminal resistance of the complementary slot is %.0f ohm",Z1)
```

Scilab code Exa 7.8 8

```
//Chapter 7: Loop, Slot and Horn Antennas
//Example 7-20.1
clc;

//Variable Initialization
delta_e = 0.2        //Path length difference in E-plane (lambda)
delta_h = 0.375      //Path length difference in H-plane (lambda)
a_e = 10             //E-plane aperture (lambda)


//Calculation
L = a_e**2/(8*delta_e)     //Horn length(lambda)
theta_e = 2*atan(a_e,2*L)*180/%pi    //Flare angle in E-plane (degrees)
theta_h = 2*acos(L/(L+delta_h))*180/%pi                          //Flare angle in the H-plane (degrees)
a_h = 2*L*tan(theta_h/2*%pi/180)    //H-plane aperture (lambda)

hpbw_e = 56/a_e      //Half power beamwidth in E-plane (degrees)
hpbw_h = 67/a_h      //Half power beamwidth in H-plane (degrees)

```

```
D = 10*log10(7.5*a_e*a_h)  //Directivity (dB)

//Result
mprintf(”The length of the pyramidal horn is %.1f lambda”, L)
mprintf(”\nThe flare angles in E–plane and H–plane are %.1f and %.2f degrees”,theta_e,theta_h)
mprintf(”\nThe H–plane aperture is %.1f lambda”,a_h)
mprintf(”\nThe Half power beamwidths in E–plane and H–plane are %d and %.1f degrees”, hpbw_e,hpbw_h)
mprintf(”\nThe directivity is %.1f dBi”,D)
```

Chapter 8

Helical Antennas

Scilab code Exa 8.1 1

```
//Chapter 8: Helical Antennas
//Example 8-5.1
clc;

//Variable Initialization
w = 5        //Width of flattened tubing at
   termination (mm)
Er = 2.7     //Relative permittivity of the sheet
Z0 = 50      //Characteristic impedance of the sheet

//Calculation
h = w/((377/(sqrt(Er)*Z0))-2)

//Result
mprintf("The required thickness of the polystyrene
   sheet is %.1f mm",h)
```

Scilab code Exa 8.2 2

```
//Chapter 8: Helical Antennas
//Example 8-5.2
clc;

//Variable Initialization
n = 16.0          //Number of turns (unitless)
C = 1         //Circumference (lambda)
S = 0.25          //Turn Spacing (lambda)

//Calculation
hpbw = 52/(C*sqrt(n*S)) //Half power beamwidth (degrees)
ax_rat = (2*n + 1)/(2*n)     //Axial ratio (unitless)
gain = 12*(C**2)*n*S         //Gain of antenna (unitless)
gain_db = 10*log10(gain)     //Gain of antenna (in dBi)

mprintf(”The half power beam width is %d degrees”,hpbw)
mprintf(”\nThe axial ratio is %.2f”,ax_rat)
mprintf(”\nThe gain is %d or %.1f dBi”,gain,gain_db)
```

Scilab code Exa 8.3 3

```
//Chapter 8: Helical Antennas
//Example 8-5.3
clc;

//Variable Initialization
n = 10.0    //Number of turns (unitless)
S = 0.236   //Spacing between turns (lambda)
n_a = 4.0   //Number of helical antennas in the array (unitless)

```

```
//Calculation
D = 12*n*S  //Directivity of a single antenna(unitless)
Ae = D/(4*%pi)   //Effective aperture (lambda^2)

A = sqrt(Ae)    //Area of square/spacing between helixes (lambda)
Ae_total = Ae*n_a   //Total effective aperture (lambda^2)
D_array = (4*%pi*Ae_total)   //Directivity of the array (unitless)
D_array_db = 10*log10(D_array)  //Directivity of the array (dBi)

//Result
mprintf(”The best spacing between the helixes is %.1f lambda”,A)
mprintf(”\nThe directivity of the array is %d or %.1f dBi”,D_array,D_array_db)
```

Scilab code Exa 8.4 4

```
//Chapter 8: Helical Antennas
//Example 8-16.1
clc;

//Variable Initialization
gain = 24.0   //Gain (dB)
alpha = 12.7    //Pitch angle (degrees)
c_lambda = 1.05 //Circumference (lambda)
s_lambda = 0.236    //Spacing between turns (lambda)

//Calculation
D = 10**(gain/10)   //Directivity (unitless)
L = D/(12*(c_lambda**2))    //Helix length (lambda)
```

```
n = L/s_lambda                    //Number of turns (unitless)
D = D/4                //Directivity for four 20-turn helixes(unitless)
Ae = D/(4*%pi)         //Effective aperture of each helix (lambda^2)

//Result
mprintf("The Axial length is %.0f lambda",L)
mprintf("\nThe number of turns for the axial length is %d",n)
mprintf("\nThe effective aperture for 20 turns is %.0f lambda square",Ae)
```

Chapter 9

Reflector Antennas

Scilab code Exa 9.1 1

```
//Chapter 9: Reflector Antennas
//Example 9-2.1
clc;

//Variable Initialization
P_transmit = 25000.0        //Power transmitted by
    station transmitter (W)
gain_dbi = 29.0             //Gain of array (dBi)
r = 7500e3                  //Distance (m)
h = 250e3                   //Height (m)
z = 377.0                   //Intrinsic impedance of
    free space (ohm)

//Calculation
gain = 10**(gain_dbi/10)      //Gain of array (
    unitless)
erp = gain*P_transmit         //Effective radiated
    power (W)
p_area = erp/(2*%pi*r*h)       //Power per unit area
    at distance r (W/m^2)
field_str = sqrt(p_area*z)     //Field strength (mV/m
```

```
)

//Result
disp(erp,"The effective radiated power in W")
mprintf("\nThe field strength at the distance r is %.3f V/m^2",field_str)
```

Chapter 11

Broadband and Frequency Independent Antennas

Scilab code Exa 11.1 1

```
//Chapter 11: Broadband and Frequency-Independent
   Antennas
//Example 11-1.1
clc;

//Variable Initialization
d = 4        //spacing (mm)
D = 100      //distance between the openings (mm)

//Calculation
lambda_short = 10*d      //Shortest wavelength (mm)
lambda_long = 2*D        //Longest wavelength (mm)
bandwidth = lambda_long/lambda_short     //Bandwidth
   (unitless)

//Result
mprintf("The approximate bandwidth is %d to 1",
   bandwidth)
```

Scilab code Exa 11.2 2

```
//Chapter 11: Broadband and Frequency-Independent Antennas
//Example 11-7.1
clc;

//Variable Initialization
gain_dbi = 7.0          //Gain (dBi)
bandwidth = 4           //Relative bandwidth (unitless)
s_lambda = 0.15         //Spacing (lambda)
k = 1.2                 //Scale constant (unitless)

//Calculation
alpha = atan((1-1/k)/(4*s_lambda))*180/%pi    //Apex angle (degrees)
n = round(log(bandwidth)/log(k))     //Number of elements(unitless)
n =n + 1
n =n + 2       //Number of elements considering conservative design (unitless)

//Result
mprintf("The apex angle is %.1f degrees",alpha)
mprintf("\nThe number of elements is %d", n)
```

Chapter 12

The Cylindrical Antenna and the Moment Method

Scilab code Exa 12.1 1

```
//Chapter 12: The Cylindrical Antenna and the Moment
    Method
//Example 12-12.1
clc;

//Variable Initialization
N = 3         //Piecewise sinusoidal dipole modes (
   unitless)
l = 1/10.0      //Dipole length (lambda)
z11_exact = 0.4935 - 3454*%i //Exact impedance
   vector(ohm)
z11_apprx = 0.4944 - 3426*%i //Approximate impedance
    vector(ohm)
z12_exact = 0.4935 + 1753*%i  //Exact impedance
   vector(ohm)
z12_apprx =  0.4945 + 1576*%i //Approximate
   impedance vector(ohm)
z13_exact = 0.4935 + 129.9*%i //Exact impedance
   vector(ohm)
```

```
z13_apprx =  0.4885 + 132.2*%i //Approximate impedance vector(ohm)

//Calculations
N2 = N + 1  //Number of equal segments (unitless)
d = 1/4     //Length of each segment (lambda)
Rmn = 20*(2*%pi*d)**2   //Real part of elements of Z-matrix, Zmn (VA)
zmat_apprx=([z11_apprx+z13_apprx,z12_apprx;2*z12_apprx,z11_apprx])//matrix(unitless)
vmat = ([0;1])  //Voltage matrix (unitless)
[i1]=linsolve(zmat_apprx,vmat)    //Current matrix (unitless)
i1=i1*-1
i_ratio = i1(2)/i1(1)          //Current ratio  (unitless)
zin = vmat(2)/i1(2)         //Input impedance (ohm)


zmat_exact =([z11_exact+z13_exact,z12_exact;2*z12_exact,z11_exact])
[i1_e] = linsolve(zmat_exact,vmat)    //Current matrix (unitless)
i1_e=i1_e*-1
i_ratio_exact = i1_e(2)/i1_e(1)          //Current ratio  (unitless)
zin_exact = vmat(2)/i1_e(2)         //Input impedance (ohm)


//Result
mprintf("The current ratio is %.2f+%.4f j",real(i_ratio),imag(i_ratio))
mprintf("\nThis is nearly equal to 1.9 indicating a nearly triangular current distribution")
mprintf("\nThe input impedance is %.3f%.3fj ohm using approximate values", real(zin),imag(zin))
mprintf("\nThe input impedance is %.3f%.3fj ohm
```

```
using exact values", real(zin_exact),imag(zin_exact))
```

Scilab code Exa 12.2 2

```
//Chapter 12: The Cylindrical Antenna and the Moment Method
//Example 12-12.2
clc;

//Variable Initialization
z_load = 2.083 + 1605*%i     //Conjugate matched load (ohm)
e0 = 1.0                          //Electric field magnitude (unitless)
l = 1/10.0                        //Length of dipole (lambda)
ima = 0+1*%i                     //Imaginary number

z11_exact = 0.4935 - 3454*%i //Exact impedance vector(ohm)
z11_apprx = 0.4944 - 3426*%i //Approximate impedance vector(ohm)
z12_exact = 0.4935 + 1753*%i  //Exact impedance vector(ohm)
z12_apprx =  0.4945 + 1576*%i //Approximate impedance vector(ohm)
z13_exact = 0.4935 + 129.9*%i //Exact impedance vector(ohm)
z13_apprx =  0.4885 + 132.2*%i //Approximate impedance vector(ohm)

//Calculation
d = l/4     //Length of each segment (lambda)
vm = (2*e0/(2*%pi))*tan(2*%pi*d/2)    //Voltage
```

```
vector (VA)
z22 = z11_exact + z_load        //Impedance matrix for loaded dipole (VA)
zmat_exact =([z11_exact+z13_exact,z12_exact;2*z12_exact,z22])//Z(impedance) matrix    (unitless)
vmat = ([vm;vm])  //Voltage matrix (unitless)
[i1]= linsolve(zmat_exact,vmat)    //Current matrix (unitless)
i1=i1*-1
i3 = i1(1)          //Current vector (unitless)
e_zn = (60*tan(2*%pi*d/2))*ima       //Free space electric field (V/m)
e_s = i1(1)*e_zn + i1(2)*e_zn + i3*e_zn          //Scattered field (V/m)
sigma = 4*%pi*(abs(e_s)**2)/(abs(e0)**2) //Radar Cross section (lambda**2)

//Result
mprintf("The radar cross section using exact values of Z matrix is %.4f lambda square",sigma(1))
```

Scilab code Exa 12.3 3

```
//Chapter 12: The Cylindrical Antenna and the Moment Method
//Example 12-12.3
clc;

//Variable Initialization
z11_exact = 2-1921*%i          //Exact impedance vector (ohm)
z12_exact = 1.9971-325.1*%i    //Exact impedance vector (ohm)

```

```
z11_apprx = 1.9739-1992*%i     //Approximate impedance vector (ohm)
z12_apprx = 1.9739-232.8*%i    //Approximate impedance vector (ohm)

vmat =([1;0])

//Calculations
zmat_exact =([z11_apprx,z12_apprx;z12_apprx,z11_apprx])          //Impedance matrix (unitless)
[i1] = linsolve(zmat_exact,vmat)       //Current matrix (unitless)
i1=i1*-1
zin = 1/i1(1)

//Result
mprintf("The input impedance for order N = 2 is %.3f%.3fi ohm",real(zin),imag(zin))
```

Chapter 15

Antennas for Special Applications

Scilab code Exa 15.1 1

```
//Chapter 15: Antennas for Special Applications
//Example 15-2.1
clc;

//Variable Initialization
frequency = 100e3    //Frequency (Hz)
height = 150    //Height of antenna(m)
RL = 2          //Loss resistance (ohm)
c = 3e8         //Speed of light (m/s)

//Calculations
wave_lt = c/frequency    //Wavelength (m)
hp = height/wave_lt //Antenna (physical) height (
   lambda)
he = hp/2           //Effective height (lambda)

Rr = 400*(hp**2)  //Radiation resistance (ohm)

R_E = Rr/(Rr+RL)        //Radiation efficiency (
```

```
unitless)

//Results
mprintf(”The Effective height of the antenna is %.3f lambda”, he)
mprintf(”\nThe Radiation resistance for 150m vertical radiator is %d ohm”, Rr)
mprintf(”\nThe radiation efficiency is %.2f or %.2f percent”, R_E,R_E*100)
```

Scilab code Exa 15.2 2

```
//Chapter 15: Antennas for Special Applications
//Example 15-4.1
clc;

//Variable Initialization
eps_r1 = 16    //Real part of relative permittivity of ground (unitless)
sigma = 1e-2    //conductivity of ground (mho per meter)
eps_0 = 8.85e-12    //Air permittivity (F/m)
f1 = 1e6        //Frequency (Hz)
f2 = 100e6      //Frequency (Hz)

//Calculation
eps_r11 = sigma/(2*%pi*f1*eps_0)    //Loss part of relative permittivity for f1 (unitless)
eps_r11_2 = sigma/(2*%pi*f2*eps_0)    //Loss part of relative permittivity for f2 (unitless)

eps_ra = eps_r1 -(%i)*eps_r11    //Relative permittivity for f1 (unitless)
eps_rb = eps_r1 -(%i)*eps_r11_2    //Relative permittivity for f2 (unitless)
```

```

n1 = sqrt(eps_ra)    //Refractive index for f1 (unitless)
n2 = sqrt(eps_rb)    //Refractive index for f2 (unitless)

E_perp1t=[]
E_perp2t=[]

for i=0:%pi/180:%pi/2
E_perp1 = [1 + (abs((sin(i) - n1)/(sin(i)+n1))*exp(%i*(2*%pi*sin(i) + ((sin(i) - n1)/(sin(i)+n1)))))]
E_perp2 = [1 + (abs((sin(i) - n2)/(sin(i)+n2))*exp(%i*(2*%pi*sin(i) + ((sin(i) - n2)/(sin(i)+n2)))))]
E_perp1t($+1)=E_perp1
E_perp2t($+1)=E_perp2
end

E_perp1_rel = E_perp1/(E_perp1t)    //Relative electric field for f1 (unitless)

E_perp2_rel = E_perp2/(E_perp2t)    //Relative electric field for f2 (unitless)


//Result
mprintf(”The loss parameter for 1MHz is %.0f”, eps_r11)
mprintf(”\nThe loss parameter for 100MHz is %.1f”, eps_r11_2)
mprintf(”\nThe relative permittivity for 1MHz is (%d%.0fj)”, eps_ra,imag(eps_ra))
mprintf(”\nThe relative permittivity for 100MHz is (%d%.1fj)”, eps_rb,imag(eps_rb))
```

Scilab code Exa 15.3 3

```
//Chapter 15: Antennas for Special Applications
//Example 15-12.1
clc;

//Variable Initialization
f = 60e6     //Frequency(Hz)
dep = 20     //Depth of antenna location (m)
sigma = 1.33e-2     //Conductivity (mho per m)
eps0 = 8.85e-12     //Air Permittivity (F/m)
epr1 = 80     //Real part of relative permittivity (
   unitless)
alphat = 10     //Elevation angle (degrees)
cl = 1     //Circumference (lambda)
%pitch = 12.5     //%pitch angle (degrees)
c = 3e8     //Speed of light (m/s)

dir_gb = 3     //Directivity of George Brown
   turnstile (unitless)
Aer_gb = 6     //Effective aperture of George Brown
   turnstile (unitless)
r = 1e3     //Distance between transmitter and
   receiver (m)
Pt = 100     //Transmitted power (W)

//Calculations
epr11 = sigma/(eps0*2*%pi*f)     //Loss term of
   relative permittivity (unitless)
epr = epr1 + %i*epr11     //Relative permittivity (
   unitless)
alphac = acos(sqrt(1/epr1))     //Critical angle (
   degrees)
alpha = acos(cos((alphat)*%pi/180)/sqrt(epr1))     //
```

```
Angle of incidence (degrees)

n1=12    //Number of turns
rad = cl/(2*%pi)    //Radius of loop (lambda)
sl = tan((12.5)*%pi/180)
hpbw1 = 52/(cl*sqrt(n1*sl))    //Half power beamwidth for 12 turns(degrees)
dir1 = 12*(cl**2)*n1*sl    //Directivity for 12 turns (unitless)
n2 = n1*2    //Number of turns
hpbw2 = 52/(cl*sqrt(n2*sl))    //Half power beamwidth for 24 turns(degrees)
dir2 = 12*(cl**2)*n2*sl    //Directivity for 24 turns (unitless)
num = 20    //Number of turns chosen

p_perpt=[]
p_pallt=[]
for i=0:%pi/180:%pi
p_perp = [(sin(i)-sqrt(epr - cos(i)**2))/(sin(i)+sqrt(epr - cos(i)**2))]
p_pall = [(epr*sin(i)-sqrt(epr - cos(i)**2))/(epr*sin(i)+sqrt(epr - cos(i)**2))]
p_perpt($+1)=p_perp
p_pallt($+1)=p_pall
end

Sr = 0.5*((p_perpt)**2 + (p_pallt)**2)    //Relative power density reflected (unitless)
St = 1 - Sr    //Relative power density transmitted (unitless)

theta = 0:%pi/180:%pi

subplot(1,2,1)
plot(theta,St)
title(”Relative Power Vs Elevation Angle”)

```

```
subplot(1,2,2)
polarplot(theta,real(St))
title("Pattern of Transmission")

wave_lt = c/f    //Wavelength (m)
diam = wave_lt/(sqrt(epr1)*%pi)    //Submerged helix diameter (m)
att_cons = (%pi*epr11)/(wave_lt*sqrt(epr1))    //Attenuation constant for water (Np/m)
att_d = 20*log10(exp(-att_cons*dep))    //Attenuation in the water path (dB)
Dir = 12*(cl**2)*num*sl    //Directivity for 20 turn helix (unitless)
Ae = Dir*(wave_lt**2)/(4*%pi)    //Effective aperture (m^2)

Pr = Pt*Ae*dir_gb/((r**2)*(wave_lt**2))    //Received power(W)

loss_inter = 10*log10(St(10))    //Loss at the interface for alpha = 83.68 (dB)
tot_loss = abs(att_d + loss_inter)    //Total loss (dB)
Pr_act = Pr/(10**(ceil(tot_loss)/10))    //Net Actual received power (W)


//Results
mprintf("Half power beamwidth for 12 turns is %.0f degrees",hpbw1)
mprintf("\nDirectivity for 12 turns is %.1f", dir1)
mprintf("\nHalf power beamwidth for 24 turns is %.0f degrees",hpbw2)
mprintf("\nDirectivity for 24 turns is %.1f", dir2)
mprintf("\nA helix of %d turns is chosen for reasonable compromise",num)
mprintf("\nThe signal level at the distance of 1km is %.2e W",Pr_act)
```

Scilab code Exa 15.4 4

```
//Chapter 15: Antennas for Special Applications
//Example 15-13.1
clc;

//Variable Initialization
fre = 3e9  //Frequency (Hz)
Re_Zc = 14.4e-3 //Real part of intrinsic impedance
   of copper (ohm)
Zd = 377    //Intrinsic impedance of air (ohm)

//Calculation
tau = atan(Re_Zc/Zd)*180/%pi //Tilt angle (degrees)

//Result
mprintf("The tilt angle is %.4f degrees",tau)
```

Scilab code Exa 15.5 5

```
//Chapter 15: Antennas for Special Applications
//Example 15-13.2
clc;

//Variable Initialization
fre = 3e9  //Frequency (Hz)
eps_r = 80  //Relative permittivity of water (
   unitless)

//Calculation
tau = atan(1/sqrt(eps_r))*180/%pi //Forward Tilt
   angle (degrees)
```

```

//Result
mprintf("The forward tilt angle is %.1f degrees",tau)
```

Scilab code Exa 15.6 6

```
//Chapter 15: Antennas for Special Applications
//Example 15-13.3
clc;

//Variable Initialization
lambda_g = 1.5      //Wavelength in guide (lambda)
m = -1  //Mode number

//Calculation
phi = acos((1/lambda_g)+m)*180/%pi //Forward tilt angle (degrees)

//Result
mprintf("The beam angle is %.1f degrees",phi)
```

Scilab code Exa 15.7 7

```
//Chapter 15: Antennas for Special Applications
//Example 15-14.1
clc;

//Variable Initialization
fre = 4e9       //Frequency (Hz)
T_sys = 100      //System Temperature (K)
S_N = 20         //Signal to Noise ratio (dB)
bandwidth = 30e6    //Bandwidth (Hz)
```

```
P_trans = 5      //Satellite transponder power (W)
dia = 2          //Satellite parabolic dish diameter (m)
sat_spacing = 2 //Spacing between satellites (degrees)
r = 36000e3        //Downlink distance (m)
k = 1.38e-23     //Boltzmann's constant (J/K)
c = 3e8          //Speed of light (m/s)

//Calculation
wave_lt = c/fre
s_n = (wave_lt**2)/(16*(%pi**2)*(r**2)*k*T_sys*bandwidth)
s_n = 10*log10(s_n)      //Signal to noise ratio for isotropic antennas (dB)

Ae = 0.5*%pi*(dia**2)/4       //Effective Aperture (m^2)
Gs = 4*%pi*Ae/(wave_lt**2)
Gs = 10*log10(Gs)        //Antenna Gain (dB)

Ge = 20 - s_n - Gs - 10*log10(P_trans) //Required earth station antenna gain(dB)
Ae_e = (10**(Ge/10))*(wave_lt**2)/(4*%pi)        //Required earth station effective aperture (m^2)
Ap = Ae_e*2      //Required Physical aperture (m^2)

De = 2*sqrt(Ap/%pi)       //Required diameter of earth-station antenna(m)
hpbw = 65/(De/wave_lt)      //Half power beam width (degree)
bwfn = 145/(De/wave_lt)     //Beamwidth between first null (degree)

//Results
mprintf(”The Required parabolic dish diameter of earth station antenna is %.1f m”,De)
mprintf(”\nThe Half power beamwidth is %.1f degrees”
```

```
,hpbw)
mprintf("\nThe Beamwidth between first null is %.1f",bwfn)
```

Scilab code Exa 15.8 8

```
//Chapter 15: Antennas for Special Applications
//Example 15-20.1
clc;

//Variable Initialization
Tr = 45     //Satellite receiver temperature (K)
rcp_gain = 6    //Right circularly polarized antenna gain (dBi)
rcp_quad_gain = 3   //RCP gain of quadrifilar helix antenna (dBi)
bandwidth = 9.6e3   //Bandwidth (Hz)
snr = 10        //Required Signal-to-Noise ratio (dB)
c = 3e8     //Speed of light (m/s)
f = 1.65e9  //Frequency (Hz)
r = 780e3   //Distance to the satellite (m)
Ta = 300    //Antenna temperature (K)
k = 1.4e-23 //Boltzmann's constant (J/K)
theta = 10  //Zenith angle (degree)
Tr_handheld = 75    //Hand held receiver temperature (K)
Tsky = 6        //Sky Temperature (K)
theta_horz = 80 //Zenith angle for horizontal dipole (degree)

//Calculations
wave_lt = c/f       //Wavelength (m)
Ld = (wave_lt/(4*%pi*r))**2 //Spatial loss factor(unitless)
```

```
Ld_db = 10*log10(Ld)    //Spatial loss factor(dB)
Tsys_up = Ta + Tr      //Satellite system temperature (K)
N = k*Tsys_up*bandwidth    //Noise power(W)
N_db = 10*log10(N)      //Noise power (dB)
E_vert = cos(%pi*cos(theta*%pi/180)/2)/sin(theta*%pi/180)       //Pattern factor for vertical lambda/2 dipole (unitless)
E_vert_db = 20*log10(E_vert)
Pt_vert_up = snr - (2.15 + (E_vert_db) - 3) - rcp_gain + ceil(N_db) - floor(Ld_db)            //Uplink power for vertical lambda/2 antenna (dB)
Pt_vert_up = 10**(Pt_vert_up/10)                  //Uplink power for vertical lambda/2 antenna (W)
Ta_down = 0.5*(Ta)+0.5*(Tsky)+3   //Downlink antenna temperature (K)
Tsys_down = Ta_down + Tr_handheld   //System temperature(K)
N_down = k*Tsys_down*bandwidth  //Noise power (W)
N_down_db = 10*log10(N_down)    //Noise power (dB)
Pt_vert_down = snr -(2.15+ (E_vert_db) - 3) - rcp_gain + ceil(N_down_db) - floor(Ld_db) //Downlink power for vertical lambda/2 antenna (dB)
Pt_vert_down = 10**(Pt_vert_down/10)         //Downlink power for vertical lambda/2 antenna (W)
E_horz = cos(%pi*cos(theta_horz*%pi/180)/2)/sin(theta_horz*%pi/180)        //Pattern factor for horizontal lambda/2 dipole (unitless)
E_horz_db = (20*log10(E_horz))
Pt_horz_up = snr -(2.15 + E_horz_db - 3) - rcp_gain + round(N_db) - round(Ld_db)          //Uplink power for horizonal lambda/2 dipole (dB)
Pt_horz_up = 10**(Pt_horz_up/10)       //Uplink power for horizonal lambda/2 dipole (W)
Pt_horz_down = snr -(2.15 + E_horz_db - 3) - rcp_gain + round(N_down_db) - round(Ld_db)         //Downlink power for horizonal lambda/2
```

```
dipole (dB)
Pt_horz_down = 10**(Pt_horz_down/10)      //Downlink power for horizonal lambda/2 dipole (W)
Pt_quad_up = snr -(rcp_quad_gain + E_horz_db) - rcp_gain + round(N_db) - round(Ld_db)          //Uplink power for RCP quadrifilar helix antenna (dB)
Pt_quad_up = 10**(Pt_quad_up/10)      //Uplink power for RCP quadrifilar helix antenna (W)
Ta_quad = 0.85*(Tsky) + 0.15*(Ta) //Downlink antenna temperature (K)
Tsys_quad = Ta_quad + Tr_handheld   //System temperature(K)
N_quad = k*Tsys_quad*bandwidth  //Noise power (W)
N_quad_db = 10*log10(N_quad)    //Noise power (dB)
Pt_quad_down = snr -(rcp_quad_gain + E_horz_db) - rcp_gain + round(N_quad_db) - round(Ld_db)          //Downlink power for RCP quadrifilar helix antenna (dB)
Pt_quad_down = 10**(Pt_quad_down/10)      //Downlink power for RCP quadrifilar helix antenna (W)


//Results
mprintf("The Uplink power for vertical lambda/2 dipole is %.1f W",Pt_vert_up)
mprintf("\nThe Uplink power for horizontal lambda/2 dipole is %.3f W",Pt_horz_up)
mprintf("\nThe Uplink power for RCP quadrifilar helix antenna is %.3f W",Pt_quad_up)
mprintf("\nThe Downlink power for vertical lambda/2 dipole is %.1f W",Pt_vert_down)
mprintf("\nThe Downlink power for horizontal lambda/2 dipole is %.3f W",Pt_horz_down)
mprintf("\nThe Downlink power for RCP quadrifilar helix antenna is %.3f W",Pt_quad_down)
```

Scilab code Exa 15.9 9

```
//Chapter 15: Antennas for Special Applications
//Example 15-20.2
clc;

//Variable Initialization
f = 1.6e9         //Frequency (Hz)
r = 1400e3        //Height (m)
r_sep = 3500e3    //Height for 10 degree seperation (m)
c = 3e8           //Speed of light(m/s)
Ta = 300          //Satellite antenna temperature (K)
Tr = 45           //Satellite receiver temperature (K)
k = 1.3e-23       //Boltzmann's constant (J/K)
bandwidth = 9.6e3   //Bandwidth (Hz)
snr = 6           //Signal to noise ratio (dB)
rcp_gain = 3      //Helix gain(dB)
beam_angle = 25 //RCP spot beam (degree)
Tsky = 6          //Sky Temperature (K)
Tr_handheld = 75    //Hand held receiver temperature (K)


//Calculations
wave_lt = c/f       //Wavelength (m)
Ld = (wave_lt/(4*%pi*r))**2
Ld = 10*log10(Ld)       //Propagation loss factor (dB)
sat_gain = 40000/(beam_angle**2)
sat_gain = 10*log10(sat_gain)   //Satellite gain (dB)

Tsys = Ta+Tr        //System temperature (K)
```

```
N = k*Tsys*bandwidth     //Noise power (W)
N_db = 10*log10(N)       //Noise power (dB)

Pt_up = snr - (rcp_gain) - (sat_gain) + N_db - Ld    //Uplink power (dB)
Pt_up = 10**(Pt_up/10)       //Uplink power (W)

Ta_quad = 0.85*(Tsky) + 0.15*(Ta) //Downlink antenna temperature (K)
Tsys_quad = Ta_quad + Tr_handheld   //System temperature(K)
N_quad = k*Tsys_quad*bandwidth  //Noise power (W)
N_quad_db = 10*log10(N_quad)    //Noise power (dB)

Pt_down = snr - (rcp_gain) - (sat_gain) + round(N_quad_db) - round(Ld)                                  //Downlink power (dB)
Pt_down = 10**(Pt_down/10)      //Downlink power (W)

Ld_sep = (wave_lt/(4*%pi*r_sep))**2
Ld_sep = 10*log10(Ld_sep)    //Propagation loss factor(dB)

Pt_sep = snr - (rcp_gain) - sat_gain + ceil(N_db) - round(Ld_sep)                              //Uplink power (dB)
Pt_sep = 10**(Pt_sep/10)        //Uplink power (W)

//Results
mprintf( "The Satellite gain is %.1f dB",sat_gain)
mprintf( "\nThe Uplink power required is %.3f W",Pt_up)
mprintf( "\nThe Downlink power required is %.4f W",Pt_down)
mprintf( "\nThe Uplink power required for 10 deg. from horizon is %.3f W",Pt_sep)
```

Scilab code Exa 15.10 10

```
//Chapter 15: Antennas for Special Applications
//Example 15-20.3
clc;

//Variable Initialization
f = 30e9        //Frequency (Hz)
Tr = 300        //Receiver temperature (K)
Ta = 275        //Satellite antenna temperature (K)
r = 1400e3      //Height (m)
c = 3e8         //Speed of light(m/s)
bw = 9.6e3      //Bandwidth per channel (Hz)
rcp_gain = 10   //RCP satellite gain (dBi)
rain_att = 10   //Rain attenuation (dB)
k = 1.4e-23 //Boltzmann's constant (J/K)
snr = 10    //Required SNR (dB)
ap_eff = 0.7    //Aperture efficiency (unitless)
Ta_2 = 10       //Dish antenna temperature (K)

//Calculations
wave_lt = c/f       //Wavelength (m)
Ld = (wave_lt/(4*%pi*r))**2 //Spatial loss factor(unitless)
Ld_db = 10*log10(Ld)    //Spatial loss factor(dB)
Tsys = Ta+Tr        //System temperature (K)

N = k*Tsys*bw   //Propagation loss due to rain (W)
N = 10*log10(N) //Propagation loss due to rain (dB)

Dr = -rcp_gain + snr - Ld_db + N + rain_att //Antenna gain (dB)
Dr = 10**(Dr/10)    //Antenna gain (unitless)

```

```
Dr_req = Dr/ap_eff    //Required antenna gain (
   unitless)
Dr_req_db = 10*log10(Dr_req)    //Required antenna
   gain (dB)

dish_dia = 2*wave_lt*sqrt(Dr_req/28)    //Required
   diameter of dish (m)

hpbw = sqrt(40000/Dr_req)    //Half power beam width
   (degrees)

Tsys2 = Ta_2 + Tr    //System temperature(K)
N2 = k*Tsys2*bw      //Propagation loss due to rain(W
   )
N2 = 10*log10(N2)    //Propagation loss due to rain(
   dB)

Pt_db = snr - Dr_req_db - rcp_gain + N2 - Ld_db +
   rain_att          //Transmitted power (dB)
Pt = 10**(Pt_db/10)

//Results
mprintf("The Uplink antenna gain required is %d dB",
   Dr_req_db)
mprintf("\nThe Required dish size %.3f m",dish_dia)
mprintf("\nThe HPBW is %.1f degrees",hpbw)
mprintf("\nThe Downlink satellite power required is
   %.3f W", Pt)
```

Scilab code Exa 15.11 11

```
//Chapter 15: Antennas for Special Applications
//Example 15-21.1
clc;

```

```
//Variable Initialization
dia = 1000     //Diameter of asteroid (m)
prc = 0.4      //Power reflection coefficient of asteroid (unitless)
f = 4e9        //Frequency (Hz)
P = 1e9        //Power (W)
s = 20e3       //Asteroid speed (m/s)
ast_dis = 0.4     //Distance of asteroid (AU)
au = 1.5e11      //Astronomical Unit (m)
c = 3e8        //Speed of light (m/s)
k = 1.38e-23     //Boltzmann's constant (m^2 kg s^-2 K^-1)
Tsys = 10       //System temperature (K)
B = 1e6      //Bandwidth (Hz)
snr = 10     //Signal to noise ratio (dB)
eap = 0.75     //Aperture efficiency (unitless)

sigma = prc*%pi*s**2     //Radar cross section (m^2)
ast_dm = au*ast_dis     //Astroid distance (m)
lmda = c/f      //Wavelength(m)

d4 = (64*(lmda**2)*(ast_dm**4)*k*Tsys*B*snr)/((eap**2)*%pi*(sigma)*P)
d = d4**(0.25)     //Diameter of dish (m)

delf = 2*s/lmda     //Doppler shift (Hz)
delt = 2*(ast_dm)/c     //Time delay (s)

timp = ast_dm/s     //Time before impact (s)


//Result
mprintf(”The diameter of the dish is %.0f m”,d)
mprintf(”\nThe doppler shift is %.1f Hz”,delf)
mprintf(”\nThe time delay for the radar signal is %d s”, delt)
mprintf(”\nThe time before impact is %d s”, timp)
```

Scilab code Exa 15.12 12

```
//Chapter 15: Antennas for Special Applications
//Example 15-26.1
clc;

//Variable Initialization
t1 = 0.3e-9      //Echo time off the top of pavement (s)
t2 = 2.4e-9      //Echo time off bottom of pavement (s)
t3 = 14.4e-9     //Echo time off bottom of water pocket (s)
er_1 = 4         //Relative permittivity of pavement (unitless)
er_2 = 81        //Relative permittivity of water pocket (unitless)
c = 3e8          //Speed of light (m/s)

//Calculations
d1 = (t2-t1)*c/(2*sqrt(er_1))
d2 = (t3-t2)*c/(2*sqrt(er_2))

//Result
mprintf("The thickness of pavement is %.2f m",d1)
mprintf("\nThe thickness of water pocket is %.1f m",d2)
```

Chapter 16

Practical Design Considerations of Large Aperture Antennas

Scilab code Exa 16.1 1

```
//Chapter 16: Practical Design Considerations of
   Large Aperture Antennas
//Example 16-2.1
clc;

//Variable Initialization
delta = 1/20.0          //rms deviation (lambda)

//Calculations
del_phi = 4*%pi*delta*180/%pi //Phase error (degrees
   )
kg = cos(del_phi*%pi/180)**2         //Gain-loss (
   unitless)
kg = 10*log10(kg)            //Gain-loss (dB)

//Result
mprintf("The gain reduction is %.1f dB",abs(kg))
```

Scilab code Exa 16.2 2

```
//Chapter 16: Practical Design Considerations of Large Aperture Antennas
//Example 16-2.2
clc;

//Variable Initialization
del_phi = 36.0    //rms phase error (degrees)
n_irr = 100.0     //Number of irregularities

//Calculations
max_side = tan(del_phi*%pi/180)**2
max_side = -10*log10(max_side)           //Maximum side-lobe level (dB)
ran_side = (1/n_irr)*tan(del_phi*%pi/180)**2
ran_side = -10*log10(ran_side)            //Random side-lobe level (dB)

//Result
mprintf(”The maximum side lobe level from main lobe is %.1f dB”, max_side)
mprintf(”\nThe random side lobe level from main lobe is %.1f dB”, ran_side)
```

Chapter 17

Antenna Temperature Remote Sensing and Radar Cross Section

Scilab code Exa 17.1 1

```
//Chapter 17: Antenna Temperature, Remote Sensing and Radar Cross Section
//Example 17-1.1
clc;

//Variable Initialization
Ta = 0.24    //Antenna temperature (K)
ang = 0.005      //Subtended angle (degrees)
hpbw = 0.116     //Antenna half power beamwidth (degrees)

//Calculations
Ts = Ta*(hpbw**2)/(%pi*(ang**2/4))

//Result
mprintf("The average temperature of the surface is %d K", Ts)
```

Scilab code Exa 17.2 2

```
//Chapter 17: Antenna Temperature, Remote Sensing and Radar Cross Section
//Example 17-1.2
clc;

//Variable Initialization
eff_aper = 500      //Antenna effective aperture (m^2)
wave_lt = 20e-2         //Wavelength (m)
Tsky = 10.0            //Sky temperature (K)
Tgnd = 300.0           //Ground temperature (K)
beam_eff = 0.7      //Beam efficiency (unitless)
aper_eff = 0.5      //Aperture efficiency (unitless)

//Calculations
phy_aper = aper_eff/eff_aper    //Physical aperture (m^2)
diam = 2*sqrt(phy_aper/%pi)  //Antenna diameter (m)
diam_l = diam/wave_lt       //Antenna diameter (lambda)

ta_sky = Tsky*beam_eff      //Sky contribution to antenna temp. (K)
ta_side = 0.5*Tsky*(1-beam_eff) //Side-lobe contribution to antenna temp. (K)
ta_back = 0.5*Tgnd*(1-beam_eff) //Back-lobe contribution to antenna temp. (K)

Ta = ta_sky + ta_side + ta_back

//Result
mprintf("The total antenna temperature is %.1f K",
```

```
   Ta)
```

Scilab code Exa 17.3 3

```
//Chapter 17: Antenna Temperature, Remote Sensing
   and Radar Cross Section
//Example 17-2.1
clc;

//Variable Initialization
Tn = 50.0      //Noise temperature (K)
Tphy = 300.0       //Physical temperature (K)
Eff = 0.99        //Efficiency (unitless)
Tn_stg = 80.0        //Noise temperature of first 3
   stages (K)
gain_db = 13.0        //Gain (dB)
Tphy_tr = 300    //Transmission line physical
   temperature (K)
Eff_tr = 0.9    //Transmission line efficiency (
   unitless)

//Calculations
gain = 10**(gain_db/10)
T_r = Tn_stg + Tn_stg/(gain) + Tn_stg/(gain**2)
                  //Receiver noise temperature (K)
Tsys = Tn + Tphy*(1/Eff - 1) + Tphy_tr*(1/Eff_tr -
   1) + (1/Eff_tr)*T_r                //System
   temperature (K)

//Result
mprintf("The system temperature is %.0f K",Tsys)
```

Scilab code Exa 17.4 4

```
//Chapter 17: Antenna Temperature, Remote Sensing and Radar Cross Section
//Example 17-2.2
clc;

//Variable Initialization
phy_aper = 2208      //Physical aperture (m^2)
f = 1415e6           //Frequency (Hz)
aper_eff = 0.54      //Aperture efficiency (unitless)
Tsys = 50            //System temperature (K)
bw = 100e6           //RF Bandwidth (Hz)
t_const = 10          //Output time constant (s)
sys_const = 2.2      //System constant (unitless)
k = 1.38e-23         //Boltzmann's constant (J/K)

//Calculations
Tmin = sys_const*Tsys/(sqrt(bw*t_const))     //Minimum detectable temperature(K)
eff_aper = aper_eff*phy_aper         //Effective aperture (m^2)
Smin = 2*k*Tmin/eff_aper         //Minimum detectable flux density (W/m^2/Hz)

//Result
mprintf("The minimum detectable flux density is %.1e W/m^2/Hz" ,Smin)
```

Scilab code Exa 17.5 5

```
//Chapter 17: Antenna Temperature, Remote Sensing and Radar Cross Section
//Example 17-3.1
clc;

//Variable Initialization
```

```
k = 1.38e-23     //Boltzmann's constant (J/K)
trans_pow = 5    //Transponder power (W)
r = 36000e3      //Distance (m)
wave_lt = 7.5e-2     //Wavelength (m)
ant_gain = 30    //Antenna gain (dB)
earth_ant = 38   //Earth station antenna gain (dB)
Tsys = 100       //Earth station receiver system
   temperature (K)
bw = 30e6        //Bandwidth (Hz)

//Calculations
s_n = wave_lt**2/(16*(%pi**2)*(r**2)*k*Tsys*bw)
s_n = 10*log10(s_n)      //Signal to Noise ratio (dB)
trans_pow_db = 10*log10(trans_pow)  //Transponder
   power (dB)
erp = ant_gain + trans_pow_db        //Effective
   radiated power (dB)
s_n_downlink = erp + earth_ant + s_n     //Signal to
   Noise ratio downlink(dB)

//Result
mprintf("The earth station S/N ratio is %.2f dB",
   s_n_downlink)
```

Scilab code Exa 17.6 6

```
//Chapter 17: Antenna Temperature, Remote Sensing
   and Radar Cross Section
//Example 17-4.1
clc;

//Variable Initialization
tf = 0.693       //Absorption co-efficient (unitless)
Te = 305         //Earth temperature (K)
Ta = 300         //Satellite antenna temperature (K)
```

```
//Calculations
Tf = (Ta - Te*exp(-tf))/(1-exp(-tf))

//Result
mprintf("The forest temperature is %.0f K", Tf)
```

Scilab code Exa 17.7 7

```
//Chapter 17: Antenna Temperature, Remote Sensing and Radar Cross Section
//Example 17-5.1
clc;

//Variable Initialization
f = 10e9           //Frequency (Hz)
wind_speed = 350     //Wind speed (km/h)
c = 3e8            //Speed of light (m/s)
vr = 1e3             //Differential velocity (m/h)

//Calculations
wave_lt = c/f    //Wavelength (m)
freq_shift = 2*(wind_speed*1000/3600)/wave_lt                              //Doppler Frequency shift (Hz)
T = 1/(2*freq_shift)     //Pulse repetition interval (s)
prf = 1/T            //Pulse repetition frequency (Hz)

fmin = 2*(vr/3600)/wave_lt   //Frequency resolution (Hz)
N = 1/((fmin)*T)       //Number of pulses

//Result
```

```
mprintf("The minimum pulse repetition frequency is %d Hz",prf)
mprintf("\nThe number of pulses to be sampled is %d", N)
```

Chapter 19

The Fourier Transform Relation betwen Aperture Distribution and Far field Pattern

Scilab code Exa 19.1 1

```
//Chapter 19: The Fourier Transform Relation between Aperture Distribution and Far-field Pattern
//Example 19-8.1
clc;

//Variable Initialization
gal_ext = 400000          //Extent of galaxy (light-years)
alpha = 0.032             //Extent of galaxy (degrees)
f = 5e9                   //Frequency (Hz)
a = 36e3                  //Maximum VLA Spacing (m)
c = 3e8                   //Speed of light (m/s)
wid = 0.03               //Width of image (degrees)
hei = 0.008                //Height of image (degrees)
flux_den = 2.5e-23        //Average flux density (W/m^2)
bw = 1e9                  //Bandwidth (Hz)
```

```

//Calculations
dist = gal_ext/sin(alpha*%pi/180)    //Distance to the galaxy (light-years)
dist_m = dist*(365*24*3600*c)
wave_lt = c/f       //Wavelength (m)
a_lambda = a/wave_lt    //Spacing in wavelength (unitless)
pix_size = 51/a_lambda  //Resolution or pixel size (degrees)
pix_size_arc = pix_size*3600    //Pixel size (arc seconds)
area = wid*hei      //Area of image (square degrees)
area_arc = area*(3600**2)    //Area of image (arc seconds)
num_pix = area_arc/pix_size_arc**2  //Number of pixels
rad_pow = flux_den*4*%pi*(dist_m**2)*bw

//Result
disp(dist,”The distance to the galaxy in light years:”)
disp(pix_size_arc,”The resolution or pixel size in arc seconds”)
disp(num_pix,”The number of pixels is”)
disp( rad_pow,”The radio power of the galaxy in W”)
```

Scilab code Exa 19.2 2

```
//Chapter 19: The Fourier Transform Relation between Aperture Distribution and Far-field Pattern
//Example 19-8.2
clc;

//Variable Initialization
```

```
f = 10e9      //Frequency (Hz)
c = 3e8       //Speed of light (m/s)
dia = 100     //Dish diameter (m)
aper_eff = 0.725     //Aperture efficiency (unitless)

//Calculation
wave_lt = c/f         //Wavelength (m)
hpbw = 66/(dia/wave_lt) //Half power beam width (degrees)

gain = 41000/(hpbw**2)   //Gain from beamwidth (unitless)
gain_db = 10*log10(gain)     //Gain from beamwidth (dBi)

gain_ap = 4*(%pi**2)*(dia/2)**2*(aper_eff)/(wave_lt**2)                //Gain from effective aperture(unitless)
gain_ap_db = 10*log10(gain_ap)   //Gain from effective aperture (dBi)

side_lobe = -23      //First side lobe level from table (dB)

//Result
mprintf("The Half Power Beamwidth is %.2f degrees",hpbw)
mprintf("\nThe gain from beamwidth is %d dBi",gain_db)
mprintf("\nThe gain from effective aperture is %d dBi",gain_ap_db)
mprintf("\nThe first side-lobe level is %d dB",side_lobe)
```

Chapter 21

Antenna Measurements

Scilab code Exa 21.1 1

```
//Chapter 21: Antenna Measurements
//Example 21-2.1
clc;

//Variable Initialization
f = 900e6        //Frequency (Hz)
len = 25e-3   //Length of antenna (m)
len_cell = 110e-3   //Length of handset chassis (m)
c = 3e8          //Speed of light (m/s)
del_L = 0.5      //Peak to Peak measurement uncertainty (dB)

//Calculations
Dm = len + len_cell   //Maximum Dimension of antenna (m)
wave_lt = c/f              //Wavelength (m)
r_rnf = (wave_lt/(2*%pi))     //Outer boundary of reactive near field (m)
r_ff = 2*(Dm**2)/wave_lt     //Fraunhofer region (m)
r2_ff = r_rnf/(10**(del_L/40)-1)                 //Minimum distance where effect of near field is
```

```
small (m)
r3_ff = 2*Dm/(10**(del_L/10)-1)             //Minimum distance where effect of rotation of AUT is small (m)

//Result
mprintf("The Outer boundary of reactive near field is at a distance %.3f m",r_rnf)
mprintf("\nThe Fraunhofer region starts at a distance  %.3f m",r_ff)
mprintf("\nThe Minimum distance where effect of near field is small enough is %.1f m",r2_ff)
mprintf("\nThe Minimum distance where effect of rotation of AUT is small enough is %.1f m",r3_ff)
```

Scilab code Exa 21.2 2

```
//Chapter 21: Antenna Measurements
//Example 21-2.2
clc;

//Variable Initialization
horn_len = 350e-3   //Length of horn (m)
ap_wid = 200e-3     //Aperture width (m)
ap_hei = 150e-3     //Aperture height (m)
del_L = 0.2         //Peak to peak uncertainty (dB)
f = 10e9            //Frequency (Hz)
c = 3e8             //Speed of light (m/s)

//Calculations
wave_lt = c/f       //Wavelength (m)
r_rnf = wave_lt/(2*%pi)  ////Outer boundary of reactive near field (m)
r_ff = 2*(ap_wid**2)/wave_lt    //Fraunhofer region (m)
```

```
r2_ff = r_rnf/(10**(del_L/40)-1)              //Minimum distance where effect of near field is small (m)
r3_ff = 2*horn_len/(10**(del_L/10)-1)             //Minimum distance where effect of rotation of AUT is small (m)

//Result
mprintf( "The Outer boundary of reactive near field is at a distance %.4f m",r_rnf)
mprintf( "\nThe Fraunhofer region starts at a distance %.1f m",r_ff)
mprintf( "\nThe Minimum distance where effect of near field is small enough is %.2f m",r2_ff)
mprintf( "\nThe Minimum distance where effect of rotation of AUT is small enough is %.1f m", r3_ff)
```

Scilab code Exa 21.3 3

```
//Chapter 21: Antenna Measurements
//Example 21-2.3
clc;

//Variable Initialization
D = 0.5      //Antenna diameter (m)
f = 300e9    //Frequency (Hz)
c = 3e8      //Speed of light (m/s)

//Calculations
wave_lt = c/f    //Wavelength (m)
r_ff = 2*(D**2)/wave_lt     //Fraunhofer region (m)

//Result
mprintf("The Fraunhofer region starts at a distance
```

```
   %d m", r_ff)
mprintf("\nAt 300 GHz the attenuation of the
   atmosphere is around 10dB/km making the
   measurement difficult in full-size ranges")
```

Scilab code Exa 21.4 4

```
//Chapter 21: Antenna Measurements
//Example 21-4.1
clc;

//Variable Initialization
D = 1        //Diameter of antenna (m)
f = 10e9     //Frequency (Hz)
c = 3e8      //Speed of light (m/s)

//Calculations
wave_lt = c/f        //Wavelength (m)
hpbw = 70*wave_lt/D //Half power beamwidth (degrees)
mea_dist = 2*(D**2)/wave_lt //Measurement distance (
   m)
trav_dist = hpbw*%pi*mea_dist/180     //Traverse
   distance (m)
taper = ((0.5/(trav_dist/2))**2)*(-3) //Amplitude
   taper (dB)

//Result
mprintf("The amplitude taper is %.1f dB", taper)
```

Scilab code Exa 21.5 5

```
//Chapter 21: Antenna Measurements
//Example 21-4.2
```

```
clc;

//Variable Initialization
pat_lev1 = -22.3     //Pattern level maximum (dB)
pat_lev2 = -23.7     //Pattern level minimum (dB)

//Calculations
S = abs(pat_lev2-pat_lev1)  //Amplitude ripple (dB)
a = (pat_lev1+pat_lev2)/2   //Pattern level (dB)

R = a + 20*log10((10**(S/20) - 1)/(10**(S/20) + 1))
                        //Reflectivity (dB)

//Result
mprintf("The reflectivity is %.1f dB", R)
```

Scilab code Exa 21.6 6

```
//Chapter 21: Antenna Measurements
//Example 21-5.1
clc;

//Variable Initialization
En = 1       //Field illuminating the AUT (unitless)
tilt_diff = 88    //Difference in tilt angles (
   degrees)

//Calculations
En_pol = En*sin(tilt_diff*%pi/180)    //Co-polar
   component of field (unitless)
En_crosspol = En*cos(tilt_diff*%pi/180)
                        //Cross-polar component of
   field (unitless)
meas_cross = 20*log10(En_crosspol)

```

```
//Result
mprintf("The measure cross-polar level is %d dB relative to the co-polar field",meas_cross)
```

Scilab code Exa 21.7 7

```
//Chapter 21: Antenna Measurements
//Example 21-5.2
clc;

//Variable Initialization
f = 1.4e9         //Frequency (Hz)
Tant = 687        //Increase in antenna temperature (K)
phy_ap = 2210     //Physical aperture (m^2)
S = 1590          //Flux density of Cygnus A (Jy)
k = 1.38e-23      //Boltzmann's constant (J/k)
c = 3e8           //Speed of light (m/s)

//Calculations
wave_lt = c/f         //Wavelength (m)
gain = (8*%pi*k*Tant)/(S*(10**-26)*wave_lt**2)    //Gain(unitless)
gain_db = 10*log10(gain)        //Gain (dBi)
Ae = gain*wave_lt**2/(4*%pi)      //Effective area (m^2)
eff_ap = Ae/phy_ap            //Aperture efficiency (unitless)

//Result
mprintf("The gain of the antenna is %d dBi", gain_db)
mprintf("\nThe aperture efficiency is %.2f or %.1f percent",eff_ap,eff_ap*100)
```

Chapter 23

Ground Wave Propagation

Scilab code Exa 23.1 1

```
//Chapter 23: Ground Wave Propagation
//Example 23-1.1
clc;

//Variable Initialization
f1 = 0.1        //Frequency (MHz)
f2 = 1.0          //Frequency (MHz)
f3 = 10.0         //Frequency (MHz)

//Calculation
d1 = 50/(f1**(1.0/3)) //Distance for f1 (miles)
d2 = 50/(f2**(1.0/3)) //Distance for f2 (miles)
d3 = 50/(f3**(1.0/3)) //Distance for f3 (miles)

//Result
mprintf( "The distance for 100kHz is %.2f miles",d1)
mprintf( "\nThe distance for 1MHz is %d miles", d2)
mprintf( "\nThe distance for 10MHz is %.2f miles",
   d3)
```

Scilab code Exa 23.2 2

```
//Chapter 23: Ground Wave Propagation
//Example 23-2.1
clc;

//Variable Initialization
f = 3e6      //Frequency (Hz)
sigma = 0.5      //Standard deviation of surface
   irregularities (unitless)
theta = 30       //Angle of incidence as measured
   from normal angle (degrees)
c = 3e8      //Speed of light (m/s)

//Calculations
wave_lt = c/f    //Wavelength (m)
R = 4*%pi*sigma*sin(theta*%pi/180)/wave_lt       //
   Roughness factor (unitless)

//Result
mprintf("The roughness factor is %.6f",R)
```

Scilab code Exa 23.3 3

```
//Chapter 23: Ground Wave Propagation
//Example 23-2.2
clc;

//Variable Initialization
f = 10e6         //Frequency (Hz)
sigma = 5        //Standard deviation of surface
   irregularities (unitless)
```

```
theta1 = 30       //Angle of incidence as measured from normal angle (degrees)
theta2 = 45       //Angle of incidence as measured from normal angle (degrees)
theta3 = 60       //Angle of incidence as measured from normal angle (degrees)
c = 3e8          //Speed of light (m/s)

//Calculations
wave_lt = c/f        //Wavelength (m)
R1 = 4*%pi*sigma*sin(theta1*%pi/180)/wave_lt                //Roughness factor for theta1 (unitless)
R2 = 4*%pi*sigma*sin(theta2*%pi/180)/wave_lt              //Roughness factor for theta2 (unitless)
R3 = 4*%pi*sigma*sin(theta3*%pi/180)/wave_lt              //Roughness factor for theta3 (unitless)

//Result
mprintf("The roughness factor for 30 degrees is %.4f", R1)
mprintf("\nThe roughness factor for 45 degrees is %.3f", R2)
mprintf("\nThe roughness factor for 60 degrees is %.4f", R3)
```

Scilab code Exa 23.4 4

```
//Chapter 23: Ground Wave Propagation
//Example 23-2.3
clc;

//Variable Initialization
```

```
f1 = 0.3      //Frequency (MHz)
f2 = 1      //Frequency (MHz)
f3 = 3      //Frequency (MHz)
sigma = 4e-5    //Standard deviation of surface irregularities (unitless)

//Calculations
x1 = (18e3)*sigma/f1    //Parameter x for f1 (unitless)
x2 = (18e3)*sigma/f2    //Parameter x for f2 (unitless)
x3 = (18e3)*sigma/f3    //Parameter x for f3 (unitless)

//Result
mprintf("The parameter x for 0.3MHz is %.1f", x1)
mprintf("\nThe parameter x for 1MHz is %.2f", x2)
mprintf("\nThe parameter x for 3MHz is %.2f", x3)
```

Scilab code Exa 23.5 5

```
//Chapter 23: Ground Wave Propagation
//Example 23-5.1
clc;

//Variable Initialization
f1 = 5e3        //Frequency (Hz)
f2 = 50e3        //Frequency (Hz)
f3 = 500e3        //Frequency (Hz)
sigma = 5e-5    //Standard deviation of surface irregularities (unitless)
eps_r = 15.0      //Relative permittivity (unitless)
mu = %pi*4e-7     //Absolute Permeability (H/m)

//Calculations
```

```
w1 = 2*%pi*f1     //Angular frequency (rad/s)
w2 = 2*%pi*f2     //Angular frequency (rad/s)
w3 = 2*%pi*f3     //Angular frequency (rad/s)


Zs1 = sqrt((w1*mu)/sqrt(sigma**2 + (w1**2)*eps_r))          //Surface impedence for f1 (ohm)
Zs2 = sqrt((w2*mu)/sqrt(sigma**2 + (w2**2)*eps_r))          //Surface impedence for f2 (ohm)
Zs3 = sqrt((w3*mu)/sqrt(sigma**2 + (w3**2)*eps_r))          //Surface impedence for f3 (ohm)

//Result
mprintf( "The surface impedence for 5kHz is %.5f ohms",Zs1)
mprintf( "\nThe surface impedence for 50kHz is %.5f ohms", Zs2)
mprintf( "\nThe surface impedence for 500kHz is %.5f ohms", Zs3)

//An error has been made in calculation/substitution of square root of
//(sigma**2 + (w1**2)*eps_r) and in the second case, the mistake in the calculation of (w2*mu)/sqrt(sigma**2 + (w2**2)*eps_r)
```

Scilab code Exa 23.6 6

```
//Chapter 23: Ground Wave Propagation
//Example 23-7.1
clc;

//Variable Initialization
f = 2.0        //Frequency (MHz)
sigma = 5e-5     //Standard deviation of surface
```

```
    irregularities (unitless)
eps_r = 15.0      //Relative permittivity (unitless)
d = 20e3        //Distance (m)
eff = 0.5       //Antenna efficiency (unitless)
c = 3e8         //Speed of light (m/s)
E1 = 0.5e-3     //Ground wave electric field
    strength (V/m)

//Calculations
wave_lt = c/(f*10**6)       //Wavelength (m)
x = (18e3)*sigma/f  //Parameter x (unitless)

b = atan((eps_r + 1)/x)     //Phase constant (
    unitless)

p = (%pi/x)*(d/wave_lt)*cos(b)    //Numerical
    distance (unitless)

A = (2 + 0.3*p)/(2 + p + 0.6*(p**2))    //Reduction
    factor (unitless)

E_t = E1 * d/A

//Result
mprintf(”The Electric field strength at the
    transmitted end is %.2f V/m”, E_t)
```

Chapter 24

Space Wave Propagation

Scilab code Exa 24.1 1

```
//Chapter 24: Space Wave Propagation
//Example 24-9.1
clc;

//Variable Initialization
tx_h = 49.0        //Transmitting antenna height (m)
rx_h = 25.0        //Receiving antenna height (m)
f = 100e6        //Frequency (Hz)
tx_p = 100.0        //Transmitted power (W)
c = 3e8          //Speed of light (m/s)
a = 6370         //Earth's radius (km)

//Calculation
wave_lt = c/f    //Wavelength (m)
d0 = sqrt(2*(4.0/3.0)*(a/1000.0))*(sqrt(tx_h)+sqrt(rx_h))                 //Line of Sight (LOS) distance (km)
d = d0*1000      //LOS (m)
Er = (88*sqrt(tx_p)/(wave_lt*(d**2)))*tx_h*rx_h                  //Received signal strength (W)

```

```
//Result
mprintf( "The Line of Sight distance is %.2f km",d0)
mprintf( "\nThe Received Signal strength is %.6f W", Er)



//There is an error in the calculation of (88*sqrt(tx_p)/(wave_lt*(d**2))) where four orders of magnitude are ignored in the resulting calculation.
```

Scilab code Exa 24.2 2

```
//Chapter 24: Space Wave Propagation
//Example 24-9.2
clc;

//Variable Initialization
tx_h = 144       //Transmitting antenna height (m)
rx_h = 25        //Receiving antenna height (m)
k = 4.0/3.0          //Equivalent earth radius/Actual earth radius (unitless)
a = 6370         //Radius of earth (km)

//Calculations
los = 4.12*(sqrt(tx_h) + sqrt(rx_h))    //Line of sight distance (km)
horz = sqrt(2*k*a*(tx_h/1000.0))  //Surface range to radio horizon from radar (km)

//Result
mprintf("The Radio horizon distance from radar is %.2f km",horz)
```

Scilab code Exa 24.3 3

```
//Chapter 24: Space Wave Propagation
//Example 24-9.3
clc;

//Variable Initialization
tx_h = 100      //Transmitting antenna height (m)
rx_h = 16       //Receiving antenna height (m)
tx_p = 40e3     //Transmitting antenna power radiation (W)
f = 100e6       //Frequency (Hz)
d = 10e3        //Distance (m)
c = 3e8         //Speed of light (m/s)
E = 1e-3        //Signal strength (V/m)

//Calculations
los = 4.12*(sqrt(tx_h) + sqrt(rx_h))     //LOS distance (km)
wave_lt = c/f       //Wavelength (m)

Es = (88*sqrt(tx_p)/(wave_lt*(d**2)))*tx_h*rx_h                     //Field strength at distance d (V/m)

dsig = sqrt(88*sqrt(tx_p)*tx_h*rx_h/(wave_lt*E))    //Distance at which field strength reduces to 1mV/m

//Result
mprintf( "The LOS distance is %.2f km", los)
mprintf( "\nThe field strength at 10km is %.5f V/m", Es)
mprintf( "\nThe distance at which field strength is
```

```
    1mV/m is %.d m",dsig)
```

Scilab code Exa 24.4 4

```
//Chapter 24: Space Wave Propagation
//Example 24-9.4
clc;

//Variable Initialization
gain = 10        //Antenna gain (dB)
Wt = 500          //Power radiation (W)
d = 15e3         //Distance (m)
Wr = 2e-6        //Received power (W)

//Calculations
Ae = Wr*(4*%pi*(d**2))/(Wt*gain)  //Effective area (
   m^2)

//Result
mprintf("The effective area of the receiving antenna
    is %.2f m^2", Ae)
```

Scilab code Exa 24.5 5

```
//Chapter 24: Space Wave Propagation
//Example 24-9.5
clc;

//Variable Initialization
h = 1000         //Height of duct (m)
delM = 0.036     //Change in refractive modulus (
    unitless)
c = 3e8          //Speed of light (m/s)
```

```

//Calculations
wl_max = 2.5*h*sqrt(delM*1e-6)  //Maximum wavelength (m)
fmax = c/wl_max     //Maximum frequency (Hz)

//Result
mprintf(”The maximum frequency that can be transmitted is %.1f MHz”, fmax/1e6)
```

Scilab code Exa 24.6 6

```
//Chapter 24: Space Wave Propagation
//Example 24-12.1
clc;

//Variable Initialization
gain = 10       //Gain of transmitting antenna (dB)
P = 100         //Radiating power (W)
f = 1e6        //Frequency (Hz)
rx_gain = 15    //Gain of receiving antenna (dB)
d = 20e3        //Distance (m)
c = 3e8         //Speed of light (m/s)
v = 1000        //Scattering volume (m^3)
sigma = 0.1     //Effective scattering cross-section (m^2)

//Calculations
wl = c/f        //Wavelength (m)
Pr_a = P*gain*rx_gain*(wl**2)/(4*%pi*(4*%pi*(d**2)))                  //Received power in case (a) (W)
F = (2*sqrt(sigma*v))/(d*sqrt(%pi))    //Attenuation Factor (unitless)
Pr_b = Pr_a*F   //Received power in case (b) (W)

```

```

//Result
mprintf(”The received power in case (a) is %.5f W”, Pr_a)
mprintf(”\nThe received power in case (b) is %e W”, Pr_b)
```

Scilab code Exa 24.7 7

```
//Chapter 24: Space Wave Propagation
//Example 24-14.1
clc;

//Variable Initialization
d = 3000        //Distance (km)
f = 3e3           //Frequency (MHz)

//Calculations
path_l = 32.45 + 20*log10(f) + 20*log10(d)

//Result
mprintf(”The path loss between the two points is %.3f dB”,path_l)
```

Chapter 25

Sky Wave Propagation

Scilab code Exa 25.1 1

```
//Chapter 25: Sky Wave Propagation
//Example 25-5.1
clc;

//Variable Initialization
muf = 10e6     //Maximum usable frequency (Hz)
h = 300     //Height of reflection (km)
n = 0.9       //Maximum value of refractive index (
   unitless)

//Calculations
Nmax = (1 - n**2)*(muf**2)/81     //Max. Number of
   electrons per cubic cm
fc = 9*sqrt(Nmax)     //Critical frequency (Hz)
dskip = 2*h*sqrt((muf/fc)**2 - 1)    //Skip distance
   (km)


//Result
mprintf("The skip distance is %.1f km",dskip)

```

```
//An error has been made in the calculation of sqrt((muf/fc)**2 - 1)
```

Scilab code Exa 25.2 2

```
//Chapter 25: Sky Wave Propagation
//Example 25-5.2
clc;

//Variable Initialization
fE = 3e6          //Critical frequency for E layer (Hz)
fF1 = 5e6         //Critical frequency for F1 layer (Hz)
fF2 = 9e6         //Critical frequency for F2 layer (Hz)

//Calculations
N_E = (fE**2)/81    //Concentration of electrons in E layer (per cubic cm)
N_F1 = (fF1**2)/81 //Concentration of electrons in F1 layer (per cubic cm)
N_F2 = (fF2**2)/81 //Concentration of electrons in F2 layer (per cubic cm)

//Result
mprintf( "The concentration of electrons in E layer is %e per cubic cm",N_E)
mprintf( "\nThe concentration of electrons in F1 layer is %e per cubic cm", N_F1)
mprintf( "\nThe concentration of electrons in F2 layer is %e per cubic cm", N_F2)
```

Scilab code Exa 25.3 3

```

//Chapter 25: Sky Wave Propagation
//Example 25-5.3
clc;

//Variable Initialization
N_E = 0.8*0.111e12  //Concentration of electrons in E layer (per cubic cm)
N_F1 = 0.8*0.3086e12 //Concentration of electrons in E layer (per cubic cm)
N_F2 = 0.8*1e12     //Concentration of electrons in E layer (per cubic cm)

//Calculations
fE = 9*sqrt(N_E)    //Critical frequency in E layer (Hz)
fF1 = 9*sqrt(N_F1)  //Critical frequency in F1 layer (Hz)
fF2 = 9*sqrt(N_F2)  //Critical frequency in F2 layer (Hz)

//Result
disp(fE,”The Critical frequency in E layer in Hz”)
disp(fF1,”The Critical frequency in F1 layer in Hz”)
disp(fF2,”The Critical frequency in F2 layer in Hz”)

//The difference appearing for fE,fF1 is a result of approximation
```

Scilab code Exa 25.4 4

```
//Chapter 25: Sky Wave Propagation
//Example 25-6.1
```

```
clc;

//Variable Initialization
hD = 70      //Height of D layer (km)
hE = 130      //Height of E layer (km)
hF1 = 230      //Height of F1 layer (km)
hF2 = 350      //Height of F2 layer (km)
theta = 10*%pi/180       //Angle of incidence (radians)

//Calculations
temp = sqrt((cos(theta))**-2 - 1)
d1 = 2*hD*temp  //Maximum single hop distance for D layer (km)
d2 = 2*hE*temp  //Maximum single hop distance for E layer (km)
d3 = 2*hF1*temp //Maximum single hop distance for F1 layer (km)
d4 = 2*hF2*temp //Maximum single hop distance for F2 layer (km)

//Result
mprintf(”The Maximum single hop distance for D layer is %.1f km”, d1)
mprintf(”\nThe Maximum single hop distance for E layer is %.2f km”, d2)
mprintf(”\nThe Maximum single hop distance for F1 layer is %.2f km”, d3)
mprintf(”\nThe Maximum single hop distance for F2 layer is %.1f km”, d4)
```

Scilab code Exa 25.5 5

```
//Chapter 25: Sky Wave Propagation
//Example 25-9.1
```

```
clc;
clear;

//Variable Initialization
d = 200      //Height of layer (km)
bet = 20     //Takeoff angle (degrees)
R = 6370     //Earth's radius (km)

//Calculations
phi_0 = 90 - bet    //Take off angle for flat earth (degrees)
h = (d/2)/(sqrt((cos(phi_0*%pi/180)**-2) - 1))     //Skip distance for case (a) (km)

phi_02 = 90 - bet - 57.2*d/(2*R)            //Take off angle for spherical earth (degrees)
h2 = (d/2)/(sqrt((cos(phi_02*%pi/180)**-2) - 1))                      //Skip distance for case (b) (km)

//Result
mprintf("The skip distance for case (a) is %.3f km", h)
mprintf("\nThe skip distance for case (b) is %.2f km", h2)
```