Scilab Textbook Companion for Antenna and Wave Propagation by G. S. N. Raju[1]

Created by
Abhishek Sharma
B.E.
Electronics Engineering
Madhav Institute of Technology and Science,Gwalior
College Teacher
None
Cross-Checked by
Harpreeth Singh

June 3, 2016

[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: Antenna and Wave Propagation

Author: G. S. N. Raju

Publisher: Dorling Kindersley, India

Edition: 1

Year: 2008

ISBN: 9788131701843

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes

Chapter 1

MATHEMATICAL PRELIMINARIES

Scilab code Exa 1.1 magnitude and direction of a vector

```
//Exa 1.1
clc;
clear;
close;
// given :
A=[1 2 3] // A is a vector
l=norm(A) // magnitude or length of vector A
a=A/norm(A) // direction of vector A
disp(l,"magnitude of vector")
disp(a,"direction of vector")
```

Scilab code Exa 1.2 Addition and Subtraction of two vectors

```
//Exa 1.2
clc;
clear;
```

```
close;
// given :
A=[2 5 6] // vector A
B=[1 -3 6] // vector B
A+B // summation of two vectors
A-B // subtraction of two vectors
disp(A+B,"summation of two vectors:")
disp(A-B,"subtraction of two vectors:")
```

Scilab code Exa 1.3 Dot product of two vectors

```
// Exa 1.3
clc;
clear;
close;
// given :
A=[1 1 2] // vector A
B=[2 1 1] // vector B
k=sum(A.*B) // dot product of vector A and B
disp(k,"dot product of vector A and B:")
```

Scilab code Exa 1.4 cross product of two vectors A and B

```
//Exa 1.4
clc;
clear;
close;
function[V]=crossprod(A,B) // defining a function v
V(1)=A(2)*B(3)-A(3)*B(2)
V(2)=A(3)*B(1)-A(1)*B(3)
V(3)=A(1)*B(2)-A(2)*B(1)
endfunction
//given:
```

```
A=[2,1,2] // vector A
B=[1,2,1] // vector B
P=crossprod(A,B)
disp(P,"cross product of vectors A and B:")
```

Scilab code Exa 1.5 Dot product of two vectors

```
// Exa 1.5
clc;
clear;
close;
// given :
A=[1 3 4] // vector A
B=[1 0 2] // vector B
k=sum(A.*B) // dot product of two vectors A and B
disp(k,"dot product of two vectors A and B:")
```

Scilab code Exa 1.9 Representation of point in cylindrical coordinates

```
//Exa 1.9
clc;
clear;
close;
// given :
p=[1,2,3] // coordinates of point p
x=1 // x coordinate of P
y=2 // y coordinate of P
z=3 // z coordinate of P
rho=sqrt(x^2+y^2) //radius of cylinder in m
phi=atand(y/x) // azimuthal angle in degrees
z=3 // in m
disp(rho,"radius of cylinder in m:")
disp(phi,"azimuthal angle in degrees:")
```

```
disp(z,"z coordinate in Degrees:")
```

Scilab code Exa 1.10 Representation of point in cylindrical coordinates

```
//Exa 1.10
clc;
clear;
close;
// given :
A=[4,2,1] // vector A
A_x=4 // x coordinate of P
A_y=2 // y coordinate of P
A_z=1 // z coordinate of P
phi=atand(A_y/A_x) // azimuthal in degrees
A_rho=A_x*cosd(phi)+A_y*sind(phi) // x coordinate of
    cylinder
A_phi=-A_x*sind(phi)+A_y*cosd(phi) // y coordinate
   of cylinder
A_z=1 // z coordinate of cylinder
A=[A_rho,A_phi,A_z] // cylindrical coordinates if
   vector A
disp(A,"cylindrical coordinates of vector A:")
```

Scilab code Exa 1.12 Representation of point in spherical coordinates

```
// Exa 1.12
clc;
clear;
close;
// given :
P=[1,2,3] // coordinates of point P in cartezian
   system
x=1// x coordinate of point P in cartezian system
```

```
y=2// y coordinate of point P in cartezian system
z=3// z coordinate of point P in cartezian system
r=sqrt(x^2+y^2+z^2) // radius of sphere in m
theta=acosd(z/(r)) // angle of elevation in degrees
phi=atand(x/y) // azimuthal angle in degrees
disp(r,"radius of sphere in m:")
disp(theta,"angle of elevation in degrees:")
disp(phi,"azimuthal angle in degrees:")


// note : answer in the book is incomplete they find
    only one coordinate but there are three.
```

Scilab code Exa 1.17 Power gain in Decibels

```
// Exa 1.17
clc;
clear;
close;
// given :
A_p=22 // power gain
A_p_dB=10*log10(A_p) // power gain in dB
disp(A_p_dB,"power gain in dB:")
```

Scilab code Exa 1.18 Current gain in Decibels

```
// Exa 1.18
clc;
clear;
close;
// given :
A_v=95 // voltage gain
A_v_dB=20*log10(A_v) // voltage gain in dB
```

```
disp(A_v_dB,"voltage gain in dB:")
```

Scilab code Exa 1.19 Power gain in nepers

```
// Exa 1.19
clc;
clear;
close;
// given :
A_p=16 // power gain
A_p_Np=log(sqrt(A_p)) // power gain in nepers
disp(A_p_Np,"power gain in nepers:")
```

Scilab code Exa 1.20 Current gain in nepers

```
// Exa 1.20
clc;
clear;
close;
// given :
A_i=34 // current gain
A_i_Np=log(A_i) // current gain in nepers
disp(A_i_Np,"current gain in nepers:")
```

Scilab code Exa 1.21 magnitude and phase of a complex number

```
// Exa 1.21
clc;
clear;
close;
```

```
// given :
A=2+4*%i // complex number A
magnitude=sqrt((real(A))^2+(imag(A))^2) // magnitude
    of complex number A
phi=atand(imag(A)/real(A)) // phase of complex
   number A in degrees
disp(magnitude,"magnitude of complex number A:")
disp(phi,"phase of complex number A in degrees:")
```

Scilab code Exa 1.22 magnitude complex conjugate and phase of complex number

```
// Exa 1.22
clc;
clear;
close;
// given :
A=1+3*%i // complex no. A
c=conj(A) // conjugate of complex no. A
magnitude=sqrt((real(A))^2+(imag(A))^2) // magnitude
    of complex number A
phi=atand(imag(A)/real(A)) // phase of complex
   number A in degrees
disp(magnitude,"magnitude of complex number A:")
disp(phi,"phase of complex number A in degrees:")
disp(c,"conjugate of complex no. A:")
```

Scilab code Exa 1.23 real and imaginary part of a complex number

```
// Exa 1.23
clc;
clear;
close;
```

```
// given :
rho=5 // magnitude of the complex number A
phi=45 // phase of a complex number A in Degrees
x=rho*cosd(phi) // real part of complex number A
y=rho*sind(phi) // imaginary part of complex number A
A=x+y*(%i) // complex number A
disp(x,"real part of complex number A:")
disp(y,"imaginary part of complex number A:")
disp(A,"complex number A:")
```

Scilab code Exa 1.24 Addition of two complex numbers

```
// Exa 1.24
clc;
clear;
close;
// given :
A_1=2+%i*3 // complex number A_1
A_2=4+%i*5 // complex number A_2
A=A_1+A_2
disp(A,"sum of complex numbers A_1 and A_2:")
```

Scilab code Exa 1.25 Subtraction of two complex numbers

```
// Exa 1.25
clc;
clear;
close;
// given :
A_1=%i*6 // complex number A_1
A_2=1-%i*2 // complex number A_2
A=A_1-A_2
```

```
disp(A,"difference of complex numbers A_1 and A_2:")
```

Scilab code Exa 1.26 Product of two complex numbers

```
// Exa 1.26
clc;
clear;
close;
// given :
A=0.4+%i*5 // complex number A
B=2+%i*3 // complex number B
P=A*B // product of complex numbers A and B
disp(P,"product of complex numbers A and B:")
```

Scilab code Exa 1.27 ratio of two complex numbers

```
// Exa 1.27
clc;
clear;
close;
// given :
A=10+%i*6 // complex number A
B=2-%i*3 // complex number B
D=A/B // division of complex numbers A and B
disp(D,"division of complex numbers A and B:")
```

Scilab code Exa 1.28 Roots of the quadratic equation

```
//Exa 1.28
clc;
```

```
clear;
close;
// given :
x=poly([0],'x')
p=(x)^2+2*x+4
roots(p) // roots of given quadratic equation
disp(roots(p),"The roots of the given quadratic
    equation are:")
```

Scilab code Exa 1.31 factorial of 4 and 6

```
//Exa 1.31
clc;
clear;
close;
f1=factorial(4) // factorial of 4
f2=factorial(6) // factorial of 6
disp(f1,"factorial of 4 is:")
disp(f2,"factorial of 6 is:")
```

Chapter 2

MAXWELL EQUATIONS AND ELECTROMAGNETIC WAVES

Scilab code Exa 2.8 magnetic field and its magnitude

```
// Exa 2.8
clc;
clear;
close;
// given :
mu_0=4*%pi*10^(-7) // permeability in free space
mu_r1=3 // region 1 relative permeability
mu_r2=5 // region 2 relative permeability
mu_1=mu_r1*mu_0 // region 1 permeability
mu_2=mu_r2*mu_0 // region 2 permeability
H1=[4 1.5 -3] // magnetic field in region 1 in A/m
Ht1=[0 1.5 -3] // tangential component of magnetic
    field H1
Hn1=[4 0 0] // normal component of magnetic field H1
Ht2=[0 1.5 -3] // as tangential componenet of
    magnetic field H2=tangential component of
    magnetic field H1
```

```
Hn2=(mu_1/mu_2)*Hn1 // normal component of magnetic
    field H2
H2=Ht2+Hn2 // magnetic field in region 2 in A/m
h2=norm(H2) // magnitude of the magnetic field H2 in
     A/m
disp(H2,"magnetic field in region 2 in A/m:")
disp(h2,"magnitude of magnetic field in region 2 in
   A/m:")
```

Scilab code Exa 2.9 electric field and electric flux density

```
// Exa 2.9
clc;
clear;
close;
// given :
epsilon_0=8.854*10^(-12) // permittivity in free
    space
sigma_1=0 //conductivity of medium 1
sigma_2=0 // conductivity of medium 2
epsilon_r1=1 // region 1 relative permittivity
epsilon_r2=2 // region 2 relative permittivity
epsilon_1=epsilon_r1*epsilon_0 // region 1
    permittivity
epsilon_2=epsilon_r2*epsilon_0 // region 2
    permittivity
E1=[1 2 3] // Electric field in region 1 in V/m
Et1=[0 2 3] // tangential component of electric
    field E1
En1=[1 0 0] // normal component of electric field E1
Et2=[0 2 3] // as tangential componenet of electric
    field E2=tangential component of electric field
    E1
En2=(epsilon_1/epsilon_2)*En1 // normal component of
     electric field E2
```

```
E2=Et2+En2 // electric field in region 2 in V/m
Dt1=epsilon_0*Et1 // tangential component of electric flux density D1
D2=epsilon_2*E2 // electric flux density in region 2 in C/m^2
disp(E2,"electric field in region 2 in V/m:")
disp(D2,"electric flux density in region 2 in C/m^2:")
```

Scilab code Exa 2.12 frequency wavelength intrinsic impedance and phase constant

```
// Exa 2.12
clc;
clear;
close;
// given :
//H=cos(10^8*t-Beta*z)ay // magnetic field in A/m
// E=377*cos(10^8*t-Beta*z)ax // electric field in V/m
omega=10^8 // angular frequency in Hz
f=omega/(2*%pi) // frequency in Hz
v_0=3*10^8 // speed of light in m/s
lambda=v_0/f // wavelength in m
Beta=2*%pi/lambda // phase constant in rad/m
disp("eta_0= E/H = 377*cos(10^8*t-Beta*z)/cos(10^8*t-Beta*z) => E/H=377")
eta_0=abs(377) // intrinsic impedence in ohm
disp(eta_0,"intrinsic impedence in ohm:")
disp(f/10^6,"frequency in MHz:")
disp(Beta,"phase constant in rad/m:")
disp(lambda,"wavelength in m:")

//note : answer of lambda in book is rounded-off, it is 18.86 meter.
```

Scilab code Exa 2.14 propagation constant

```
// Exa 2.14
clc;
clear;
close;
// given :
f=100 // frequency in MHz
f=100*10^6 // frequency in Hz
v_0=3*10^8 // speed of light in m/s
// formula : Gamma=%i*omega*sqrt(mu_0*epsilon_0)=%i*
    omega/v_0 =(%i*2*pi*f)/v_0
Gamma=%i*2*%pi*f/v_0 //propagation constant
disp(Gamma,"propagation constant in m^-1:")
funcprot(0)
```

Scilab code Exa 2.15 amplitude frequency wavelength and phase constant

```
// Exa 2.15
clc;
clear;
close;
// given : H(z,t)=48*cos(10^8*t+40*z)ay // equation
    of magnetic field
A=48 // amplitude of the magnetic field in A/m
omega=10^8 // angular frequency in radians/sec
f=omega/(2*%pi) // frequency in Hz
Beta=40 // phase constant in rad/m
lambda=2*%pi/Beta // wavelength in m
disp(A,"amplitude of the magnetic field in A/m:")
disp(f/10^6,"frequency in MHz:")
```

```
disp(Beta,"phase constant in rad/m:")
disp(lambda,"wavelength in m:")
```

Scilab code Exa 2.16 electric field in free space and in medium

```
// Exa 2.16
clc;
clear;
close;
// given :
H=2 // ampliutude of magnetic field in A/m
sigma=0 // conductivity
mu_0=4*%pi*10^-7 // permeability in free space in H/
    m
epsilon_0=8.854*10^-12 // permittivity in free space
     in F/m
mu=mu_0 // permeability in F/m
epsilon=4*epsilon_0 // permittivity in F/m
Eta_0=120*%pi // intrinsic impedence in free space
    in ohm
E=Eta_0*H // electric field in V/m
disp(E,"magnitude of electric field in V/m in free
    space:")
Eta=sqrt(mu/epsilon) // intrinsic impedence in ohm
E=Eta*H // magnitude of electric field
disp(E,"magnitude of electric field in V/m:")
```

Scilab code Exa 2.17 propagation constant and intrinsic impedance

```
//Exa 2.17
clc;
clear;
close;
```

```
// given :
sigma=0 // conductivity in mho/m
f=0.3 // frequency in GHz
f=0.3*10^9 // frequency in Hz
omega=2*%pi*f // angular frequency in rad/sec
// formula : Gamma=sqrt(%i*omega*mu*(sigma+%i*omega*epsilon))=%i*omega*sqrt(mu*epsilon)
epsilon_0=8.854*10^-12 // permittivity in free space in F/m
epsilon=9*epsilon_0 // permittivity in F/m
mu_0=4*%pi*10^-7 //  permeability in free space in H/m
mu=mu_0 // permeability in H/m
Gamma=%i*omega*sqrt(mu*epsilon) // propagation constant im m^-1
disp(Gamma,"propagation constant im m^-1:")
// formula : eta=sqrt((%i*omega*mu)/(sigma+omega*epsilon))=sqrt(mu/epsilon)
eta=sqrt(mu_0/(9*epsilon_0)) // intrinsic impedence in ohm
disp(eta,"intrinsic impedence in ohm:")


// note : answer of propagation constant in book is wrong.they put mu_0=4*10^-7 in part 1 which is wrong the correct value of mu_0 is 4*%pi*10^-7.
```

Scilab code Exa 2.18 frequency and permittivity

```
//Exa 2.18
clc;
clear;
close;
// given :
lambda=0.25 // wavelength in m
```

```
v=1.5*10^10 // velocity of propagation of wave in cm/sec
v=1.5*10^8 // velocity of propagation of wave in m/sec
f=v/lambda // frequency in Hz
disp(f/10^6,"frequecy in MHz:")
epsilon_0=8.854*10^-12 // permittivity in free space in F/m
mu_0=4*%pi*10^-7 //  permeability in free space in H/m
mu=mu_0 // permeability in H/m
v_0=3*10^8 // speed of light in m/s
// formula : v=1/(mu*epsilon)=1/(mu_0*epsilon_0*epsilon_r)=v_0/sqrt(epsilon_r)
epsilon_r=(v_0/v)^2 // relative permittivity
disp(epsilon_r,"relative permittivity:")


//note : answer in the book is wrong.
```

Scilab code Exa 2.19 frequency phase constant and wavelength

```
// Exa 2.19
clc;
clear;
close;
// given : E=5*sin(10^8*t+4*x)az // equation of electric field
A=5 // amplitude of the electric field
omega=10^8 // angular frequency in radians/sec
f=omega/(2*%pi) // frequency in Hz
Beta=4 // phase constant in rad/m
v_0=3*10^8 // speed of light in m/s
lambda=v_0/f // wavelength in m
disp(f/10^6,"frequency in MHz:")
```

```
disp(Beta,"phase constant in rad/m:")
disp(lambda,"wavelength in m:")
```

Scilab code Exa 2.20 conducting characteristics of earrth

```
// Exa 2.20
clc;
clear;
close;
// given :
sigma=10^-2 // conductivity of earth in mho/m
epsilon_r=10 // relative permittivity
mu_r=2 // relative permeability
epsilon_0=(1/(36*%pi))*10^-9 // permittivity in free
    space
epsilon=epsilon_r*epsilon_0 // permittivity
f1=50 // frequency in Hz
omega1=2*%pi*f1 // angular frequency in rad/sec
disp("When frequency=50Hz:")
k1=sigma/(omega1*epsilon)
disp(k1,"K1 is equal to")
disp("since k1>>1 hence it behaves like a good
    conductor:")
f2=1 // frequency in kHz
f2=1*10^3 // frequency in Hz
omega2=2*%pi*f2 // angular frequency in rad/sec
disp("When frequency=1kHz:")
k2=sigma/(omega2*epsilon)
disp(k2,"K2 is equal to")
disp("since k2>>1 hence it behaves like a good
    conductor:")
f3=1 // frequency in MHz
f3=1*10^6 // frequency in Hz
omega3=2*%pi*f3 // angular frequency in rad/sec
disp("When frequency=1MHz:")
```

```
k3=sigma/(omega3*epsilon)
disp(k3,"K3 is equal to")
disp("since k3=18 hence it behaves like a moderate conductor:")
f4=100 // frequency in MHz
f4=100*10^6 // frequency in Hz
omega4=2*%pi*f4 // angular frequency in rad/sec
disp("When frequency=100MHz:")
k4=sigma/(omega4*epsilon)
disp(k4,"K4 is equal to")
disp("since k4=0.18 hence it behaves like a quasi-dielectric:")
f5=10 // frequency in GHz
f5=10*10^9 // frequency in Hz
omega5=2*%pi*f5 // angular frequency in rad/sec
disp("When frequency=10GHz:")
k5=sigma/(omega5*epsilon)
disp(k5,"K5 is equal to")
disp("since k5<<1 hence it behaves like a good dielectric:")
```

Scilab code Exa 2.21 attenuation constant phase constant phase velocity propagation constant and intrinsic impedance

```
// Exa 2.21
clc;
clear;
close;
// given :
f=60 // frequency in Hz
omega=2*%pi*f // angular frequency in rad/sec
sigma=5.8*10^7 // conductivity in mho/m
epsilon_0=8.854*10^-12 // permittivity in free space in F/m
mu_0=4*%pi*10^-7 //  permeability in free space in H
```

```
/m
epsilon_r=1 // relative permittivity
mu_r=1 // relative permeability
epsilon=epsilon_r*epsilon_0 // permittivity
mu=mu_0*mu_r // permeability
k=sigma/(omega*epsilon) // ratio
disp(k,"ratio k is equal to")
disp("since k>>1 therefore it is very good conductor :")
alpha=sqrt(omega*mu*sigma/2) // attenuation constant in m^-1
Beta=sqrt(omega*mu*sigma/2) // phase constant in m^-1
Gamma=alpha+(%i*Beta) // propagation constant in m^-1
lambda=2*%pi/Beta // wavelength
eta=sqrt((%i*omega*mu/sigma)) // intrinsic impedence in ohm
v=lambda*f // phase velocity of wave in m/s
disp(alpha,"attenuation constant in m^-1:")
disp(Beta,"phase constant in m^-1:")
disp(Gamma,"propagation constant in m^-1:")
disp(eta,"intrinsic impedence in ohm:")
disp(lambda*100,"wavelength in cm:")
disp(v,"phase velocity of wave in m/s:")
```

Scilab code Exa 2.22 depth of penetation

```
// Exa 2.22
clc;
clear;
close;
// given :
f1=60 // frequency in Hz
omega1=2*%pi*f1 // angular frequency in Hz
```

```
f2=100 // frequency in MHz
f2=100*10^6 // frequency in Hz
omega2=2*%pi*f2 // angular frequency in Hz
sigma=5.8*10^7 // conductivity in mho/m
epsilon_0=8.854*10^-12 // permittivity in free space
    in F/m
mu_0=4*%pi*10^-7 //  permeability in free space in H
   /m
epsilon_r=1 // relative permittivity
mu_r=1 // relative permeability
epsilon=epsilon_r*epsilon_0 // permittivity
mu=mu_0*mu_r // permeability

disp("At f=60Hz")
k1=(sigma)/(omega1*epsilon) // ratio
disp(k1," ratio k is equal to")
disp("since k>>1 therefore it is very good conductor
    at f=60Hz:")
delta1=(sqrt(2/(omega1*mu*sigma))) // depth of
   penetration in m
disp(delta1,"depth of penetration delta1 in m:")

disp("At f=100Hz")
k2=sigma/(omega2*epsilon) // ratio
disp(k2,"ratio k is equal to")
disp("since k2>>1 therefore it is very good
   conductor at f=100Hz:")
delta2=(sqrt(2/(omega2*mu*sigma))) // depth of
   penetration in m
disp(delta2,"depth of penetration delta2 in m:")
```

Scilab code Exa 2.23 displacement current

```
// Exa 2.23
clc;
```

```
clear;
close;
// given :
Ic=10 // conduction current in ampere
epsilon_r=1 // relative permittivity
epsilon_0=8.854*10^-12 // permittivity in free space
epsilon=epsilon_r*epsilon_0 // permittivity
sigma=5.8*10^7 // conductivity in mho/m
disp("when f=1MHz")
f=1 // frequency in MHz
f=1*10^6 // frequency in Hz
Id=2*%pi*f*epsilon*Ic/sigma // displacement current
disp(Id,"displacement current when f=1MHz in A:")
disp("when f=100MHz")
f=100 // frequency in MHz
f=100*10^6 // frequency in Hz
Id=2*%pi*f*epsilon*Ic/sigma // displacement current
disp(Id,"displacement current when f=100MHz in A:")
```

Scilab code Exa 2.25 reachable depth of the sea

```
// Exa 2.25
clc;
clear;
close;
// given :
Em=20 // minimum signal level required for vessel under sea water in microV/m
Em=20*10^-6 // minimum signal level required for vessel under sea water in V/m
E=100 // electric intensity of wave in V/m
v=3*10^8 // speed of light in m/s
f=4 // frequency in MHz
f=4*10^6 // frequency in Hz
omega=2*%pi*f // angular frequency in Hz
```

```
sigma=4 // conductivity of sea water in mho/m
epsilon_r=81 // relative permittivity
epsilon_0=8.854*10^-12 // permittivity in free space
epsilon=epsilon_r*epsilon_0 // permittivity
mu_r=1 // relative permeability
mu_0=4*%pi*10^(-7) // permeability in free space
mu=mu_r*mu_0 // permeability
k=(sigma)/(omega*epsilon)//ratio
disp("ratio k is equal to: ")
disp(k,"ratio:")
disp("K is >>1 so sea water is a good conductor")
eta_1=377 // intrinsic impedance in free space in ohm
alpha_1=0 // attenuation constant in free space in m^-1
beta_1=omega/v // phase constant in m^-1
mageta_2=sqrt(omega*mu/sigma) // magnitude of eta_2(intrinsic impedance of sea water in ohm)
argeta_2=45 // argument of eta_2 in degrees
eta_2=mageta_2*cosd(argeta_2)+%i*mageta_2*sind(argeta_2) //intrinsic impedance in complex form (r*cos(theta)+%i*r*sin(theta))
TC=2*eta_2/(eta_1+eta_2) // transmission cofficient
Et=abs(TC)*E // transmitted electric field in V/m
alpha_2=sqrt(omega*mu*sigma/2) // attenuation constant for sea water in m^-1
// formula: Et*exp(-alpha_2*d)=Es
d=-(1/alpha_2)*log(Em/Et) // depth in the sea that can be reached by the aeroplane in m
disp(d,"depth in the sea that can be reached by the aeroplane in m:")


// note 1: the value of alpha_2 in book is 7.905 but it is "7.94" exactly calculated by scilab
//note 2 : The correct answer of the Depth(d) is "1.41094" the answer in the book is wrong.
```

Scilab code Exa 2.27 Power per unit area

```
// Exa 2.27
clc;
clear;
close;
// given :
eta_0=377 // intrinsic impedance in free space in ohm
disp("E=sin(omega*t-beta*z)ax+2*sin(omega*t-beta*z+75)ay // electric field in V/m")
Ex=1// magnitude of Ex
Ey=2 // magnitude of Ey
E=sqrt(Ex^2+Ey^2) // resultant magnitude
Pav=(1/2)*E^2/(eta_0) // power per unit area conveyed by the wave in free space
disp(Pav*1000,"power per unit area conveyed by the wave in free space in mW/m^2:")
```

Scilab code Exa 2.28 average power and maximum energy density of wave

```
// Exa 2.28
clc;
clear;
close;
// given :
epsilon_0=8.854*10^-12 // permittivity in free space in F/m
mu_0=4*%pi*10^-7 //  permeability in free space in H/m
epsilon_r=4 // relative permittivity
mu_r=1 // relative permeability
```

```
epsilon=epsilon_r*epsilon_0 // permittivity
mu=mu_0*mu_r // permeability
H=5 // magnitude of magnetic field in mA/m
H=5*10^-3 // magnitude of magnetic field in A/m
eta=sqrt(mu/epsilon)// intrinsic impedence in ohm
E=H*sqrt(mu/epsilon) // magnitude of electric field
P_av=E^2/(2*eta) // average power
W_E=epsilon*E^2 // maximum energy density of the wave
disp(P_av*10^6,"Average power in micro*w/m^2:")
disp(W_E*10^12,"maximum energy density of the wave in PJ/m^3:")


// note: P_av is = 2353.75 in book but it is 2354.58 correctly calculated by scilab.
```

Scilab code Exa 2.29 energy density and total energy

```
// Exa 2.29
clc;
clear;
close;
// given :
epsilon_0=8.854*10^-12 // permittivity in free space in F/m
mu_0=4*%pi*10^-7 //  permeability in free space in H/m
epsilon_r=1 // relative permittivity
mu_r=1 // relative permeability
epsilon=epsilon_r*epsilon_0 // permittivity
mu=mu_0*mu_r // permeability
E=100*sqrt(%pi) // magnitude of electric field in V/m
W_E=(1/2)*epsilon*E^2 // electric energy density of
```

```
the wave
disp(W_E*10^9,"electric energy density of the wave in nJ/m^3:")
W_H=W_E // as the energy density is equal to that of magnetic field for a pla'ne travelling wave
W_T=W_E+W_H // total energy density
disp(W_H*10^9,"magnetic energy density of wave in nJ/m^3:")
disp(W_T*10^9,"Total energy density in nJ/m^3:")
```

Scilab code Exa 2.30 transmitted distance of an electromagnetic wave

```
// Exa 2.30
clc;
clear;
close;
// given :
sigma=5 // conductivity of sea water in mho/m
f1=25 // frequency in kHz
f1=25*10^3 // frequency in Hz
omega1=2*%pi*f1 // angular frequency in Hz
f2=25 // frequency in MHz
f2=25*10^6 // frequency in Hz
omega2=2*%pi*f2 // angular frequency in Hz
epsilon_r=81 // relative permittivity
epsilon_0=8.854*10^-12 // permittivity in free space
epsilon=epsilon_r*epsilon_0 // permittivity
mu_r=1 // relative permeability
mu_0=4*%pi*10^(-7) // permeability in free space
mu=mu_r*mu_0 // permeability
disp("when frequency=25kHz")
alpha_1=omega1*sqrt((mu*epsilon)/2*(sqrt(1+(sigma^2/(omega1^2*epsilon^2)))-1)) // attenuation constant when f=25kHz
//formula: exp(-alpha*x)=0.1
```

```
x1=2.3/alpha_1 // transmitted distance in m
disp(x1,"transmitted distance in m:")
disp("when frequency=25MHz")
alpha_2=omega2*sqrt((mu*epsilon)/2*(sqrt(1+(sigma^2/(omega2^2*epsilon^2)))-1)) // attenuation constant when f=25MHz
x2=2.3/alpha_2 // transmitted distance in m
disp(x2,"transmitted distance in m:")


//note: the values of epsilon_r=81 and of mu_r=1 for sea water which are not given in the book.
```

Scilab code Exa 2.31 incident and reflected magnetic field and reflected electric field

```
// Exa 2.31
clc;
clear;
close;
// given :
E_i=1 // magnitude of incident electric field in mV/m
E_i=1*10^-3 // magnitude of incident electric field in V/m
epsilon_0=8.854*10^-12 // permittivity in free space in F/m
mu_0=4*%pi*10^-7 //  permeability in free space in H/m
theta_i=15 // incident angle in degrees
epsilon_r1=8.5 // relative permittivity of medium 1
mu_r1=1 // relative permeability of medium 1
epsilon1=epsilon_r1*epsilon_0 // permittivity
mu1=mu_0*mu_r1 // permeability
eta1=sqrt(mu1/epsilon1) // intrinsic impedence of
```

```
medium 1 in ohm
epsilon2=epsilon_0 // permittivity of medium 2
mu2=mu_0 // permeability of medium 2
eta2=sqrt(mu2/epsilon2) // intrinsic impedence of medium 2 in ohm
//formula : sind(theta_i)/sind(theta_t)=sqrt(epsilon2/epsilon1)
theta_t=asind(sind(theta_i)/(sqrt(epsilon2/epsilon1))) // transmitted angle in degrees
E_r=E_i*((eta2*cosd(theta_i)-(eta1*cosd(theta_i)))/(eta2*cosd(theta_i)+eta1*cosd(theta_i))) // reflection coefficient of electric field
disp(E_r*1000,"reflection coefficient of electric field in mV/m:")
H_i=E_i/eta1 // incident cofficient of magnetic field
disp(H_i*10^6,"incident cofficient of magnetic field in micro*A/m:")
H_r=E_r/eta1 // reflection coefficient of electric field
disp(H_r*10^6,"reflection coefficient of magnetic field in micro*A/m:")


// note : minute difference in decimal answer between scilab and book.
```

Scilab code Exa 2.32 average power density absorbed

```
// Exa 2.32
clc;
clear;
close;
// given :
sigma=5.8*10^7 // conductivity in mho/m
```

```
f=2 // frequency in MHz
f=2*10^6 // frequency in Hz
omega=2*%pi*f // angular frequency in rad/sec
E=2 // magnitude of electric field in mV/m
E=2*10^-3 // magnitude of electric field in V/m
epsilon_0=8.854*10^-12 // permittivity in free space
    in F/m
mu_0=4*%pi*10^-7 //  permeability in free space in H
   /m
epsilon_r=1 // relative permittivity
mu_r=1 // relative permeability
epsilon=epsilon_r*epsilon_0 // permittivity
mu=mu_0*mu_r // permeability
eta=sqrt(mu*omega/sigma) // intrinsic impedence in
   ohm
P_av=(1/2)*E^2/eta // average power density
   anbsorbed by copper
disp(P_av*1000,"average power density anbsorbed by
   copper in mW/m^2:")
```

Chapter 3

RADIATION AND ANTENNAS

Scilab code Exa 3.1 radiation resistance

```
// Exa 3.1
clc;
clear;
close;
Lm=poly(0,'Lm')// defining Lm as lambda
dl=Lm/40 // dipole length
Rr=80*(%pi)^(2)*(dl/Lm)^2
Rr=horner(Rr,1)
disp(Rr,"radiation resistance of dipole in ohm if dl=Lm/40 :")
dl=Lm/60 // dipole length
Rr=80*(%pi)^(2)*(dl/Lm)^2
Rr=horner(Rr,1)
disp(Rr,"radiation resistance of dipole in ohm if dl=Lm/60 :")
dl=Lm/80 // dipole length
Rr=80*(%pi)^(2)*(dl/Lm)^2
Rr=horner(Rr,1)
disp(Rr,"radiation resistance of dipole in ohm if dl
```

=Lm/80 :")
```

**Scilab code Exa 3.3** Directivity of half wave dipole

```
//Exa 3.3
clc;
clear;
close;
// given :
Pr=1 //power in watt
I=sqrt(Pr/73) // current in A
Eta0=120*(%pi) // constant
r=poly(0,'r')
E_max=60*I/r
RI=r^2*E_max^2/Eta0 // radiation intensity
Gd_max=4*(%pi)*(RI)/Pr
Gd_max=horner(Gd_max,1)
disp(Gd_max,"Directivity of a half wave dipole:" )
```

Scilab code Exa 3.4 Power radiated

```
//Exa3.4
clc;
clear;
close;
Rr=300 // radiation resistance in ohm
I=3 // in A
//formula: Pr=I^2*R
Pr=I^2*Rr // power radiated in watt
disp(Pr,"power radiated by antenna in watts:")
```
```

Scilab code Exa 3.5 effective area of half wave dipole

```
//Exa 3.5
clc;
close;
clear;
// given:
f=500 //frequency in mega hertz
f=500*10^6 //frequency in hertz
c=3*10^8 //speed of light in m/s
Gdmax=1.644 // directivity of a half wave dipole
lambda=c/f //wavelength in meter
Ae=((lambda)^2*Gdmax)/(4*(%pi)) // Effective area in m^2
disp(Ae,"effective area of half wave dipole in m^2:")
```

Scilab code Exa 3.6 effective area of hertzian dipole

```
// Exa 3.6
clc;
clear;
close;
// given :
f=100 //frequency in Mhz
f=100*10^6 //frequency in hertz
c=3*10^8 //speed of light in m/s
D=1.5 // directivity
lambda=c/f //wavelength in meter
Ae=(lambda^2*D)/(4*(%pi)) // effective area in m^2
disp(Ae,"Effective area of hertezian dipole in m^2:")
```

Chapter 4

ANALYSIS OF LINEAR ARRAYS

Scilab code Exa 4.1 null to null beam width of a broadside array

```
//Exa 4.1
clc;
clear;
close;
L=poly(0,'L') //Defining L as lambda
l=10*L
N=20 // number of elements
d=l/N
// formula : BW=(2*(L/d)*1/N)
BW1=(horner((2*L/(N*d)),1))
disp(BW1,"Null-to-null BW of broadside array in
    radians when l=10*L,N=20:")
l=50*L
N=100 // number of elements
d=l/N
// formula : BW=(2*(L/d)*1/N)
BW2=(horner((2*L/(N*d)),1))
disp(BW2,"Null-to-null BW of broadside array in
    radians when l=50*L,N=100:")
```

```
l=20*L
N=50 // number of elements
d=l/N
// formula : BW=(2*(L/d)*1/N)
BW3=(horner((2*L/(N*d)),1))
disp(BW3,"Null-to-null BW of broadside array in
    radians when l=20*L,N=50:")
```

Scilab code Exa 4.2 null to null beam width of a endfire array

```
//Exa 4.2
clc;
clear;
close;
L=poly(0,'L') //Defining L as lambda
l=10*L
N=20 // number of elements
d=l/N
// formula : BW=(2*sqrt(2*(L/d)*1/N))
BW1=2*sqrt(horner((2*L/(N*d)),1))
disp(BW1,"Null-to-null BW of end-fire array in
    radians when l=10*L,N=20:")
l=50*L
N=100 // number of elements
d=l/N
// formula : BW=(2*sqrt(2*(L/d)*1/N))
BW2=2*sqrt(horner((2*L/(N*d)),1))
disp(BW2,"Null-to-null BW of end-fire array in
    radians when l=50*L,N=100:")
l=20*L
N=50 // number of elements
d=l/N
// formula : BW=(2*sqrt(2*(L/d)*1/N))
BW3=2*sqrt(horner((2*L/(N*d)),1))
disp(BW3,"Null-to-null BW of end-fire array in
```

```
   radians when l=20*L,N=50:")
```

Scilab code Exa 4.3 null to null beam width and directivity

```
//Exa 4.3
clc;
clear;
close;
f=6 //frequency in GHz
f=6*10^9 //frequency in Hz
c=3*10^8 //speed of light in m/s
l=10 // array length in meter
lambda=c/f //wavelength in meter
// formula : BWFN = 2*lambda/l
BWFN = 2*(lambda/l) // band width in radians
disp(BWFN,"null-to-null Beamwidth of broad side
   array in radians:")
D=2*(l/lambda) // directivity
disp(D,"Directivity:")
```

Scilab code Exa 4.4 Progressive phase shift and array length

```
//Exa 4.4
clc;
clear;
close;
//given :
f=10 //frequency in Ghz
f=10*10^9 //frequency in hertz
c=3*10^8 //speed of light in m/s
lambda=c/f //wavelength in meter
N=50 // number of elements
d=0.5*lambda // element spacing in meter
```

```
Beta=2*(%pi)/lambda // phase shift
alpha=Beta*d // progressive phase shift in radians
l=N*d // Araay length in meter
disp(alpha,"progressive phase shift in radians:")
disp(l,"Array length in meter")
```

Scilab code Exa 4.5 null to null and half power beam width and directivity

```
//Exa 4.5
clc;
clear;
close;
// given :
N=100 // no. of elements
Lm=poly(0,'Lm') // defining Lm as lambda
d=0.5*Lm
l=N*d // array length
B.W.F.N = 114.6 /(l/Lm) // beam width in degrees
B.W.F.N=horner(B.W.F.N,1)
disp(B.W.F.N ,"null-to-null beamwidth in degrees:")
H.P.B.W = B.W.F.N/2 // half power beam width in degrees
disp(H.P.B.W ,"half power beamwidth in degrees:")
D1=2*(l/Lm) // directivity of broad side array
D1=horner(D1,1)
D2=4*(l/Lm) // directivity of end fire array
D2=horner(D2,1)
disp(D1,"directivity of broad side array:")
disp(D2,"directivity of end fire array:")

// note : answer in the book is mis-printed,the HPBW is not 11.46 it should be 1.146 degrees.

// note: misprint in second step of part a in book
```

```
correct is l=N*d=100*0.5*lambda=50*lambda
```

Scilab code Exa 4.9 relative excitation levels

```
// Exa 4.9
clc;
clear;
close;
// given :
// formula :combination(n,r)=(factorial(n))/(
   factorial(r)*factorial(n-r))
disp("when n=2")
n=2
a_0=factorial(1)/factorial(0)*factorial(1) //
   relative excitation level 1
a_1=factorial(1)/factorial(1)*factorial(0) //
   relative excitation level 2
disp((string(a_0)+" "+string(a_1)),"relative
   excitation levels of binomial array at n=2:")
disp("when n=3")
n=3
a_1=factorial(1)/factorial(1)*factorial(0) //
   relative excitation level 2
a_0=2*a_1 // relative excitation level 1
disp((string(a_1)+" "+string(a_0)+" "+string(a_1)),"
   relative excitation levels of binomial array at n
   =3:")
```

Scilab code Exa 4.10 basic and actual transmission loss

```
// Exa 4.10
clc;
clear;
```

```
close;
// given :
d=30 //separation distance in meter
f=10 //frequency in mega hertz
f=10*10^6 //frequency in hertz
c=3*10^8 //speed of light in m/s
lambda=c/f //wavelength in meter
Gt=1.65 //transmitting gain in dB
Gr=1.65 //receiving gain in dB
// basic transmission loss :
// formula : Lb=10*log(((4*(%pi)*d)^2/(lambda)^2))
Lb=10*log10((4*(%pi)*d)^2/(lambda)^2) // basic
   transmmision loss in dB
disp(Lb,"basic transmmision loss in dB:")
// actual transmission loss :
La=Lb-Gt-Gr // actual transmisson loss in dB
disp(La,"actual transmisson loss in dB:")
```

Scilab code Exa 4.11 basic transmission loss

```
//Exa 4.11
clc;
clear;
close;
//when frequency=0.3GHz
// given :
f=0.3 //frequency in Ghz
f=0.3*10^9 //frequency in hertz
c=3*10^8 //speed of light in m/s
lambda=c/f //wavelength in meter
d1=1.6 // in Km
d1=1.6*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb1=20*log10(4*%pi*d1/lambda) // basic transmission
   loss in dB
```

```
disp(Lb1,"basic transmission loss in dB when d=1.6Km, f=0.3GHz:")
d2=16 // in Km
d2=16*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb2=20*log10(4*%pi*d2/lambda) // basic transmission loss in dB
disp(Lb2,"basic transmission loss in dB when d=16Km, f=0.3GHz:")
d3=160 // in Km
d3=160*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb3=20*log10(4*%pi*d3/lambda) // basic transmission loss in dB
disp(Lb3,"basic transmission loss in dB when d=160Km, f=0.3GHz:")
d4=320 // in Km
d4=320*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb4=20*log10(4*%pi*d4/lambda) // basic transmission loss in dB
disp(Lb4,"basic transmission loss in dB when d=320Km, f=0.3GHz:")
// when frequency is 3Ghz
// given :
f=3 //frequency in Ghz
f=3*10^9 //frequency in hertz
c=3*10^8 //speed of light in m/s
lambda=c/f //wavelength in meter
d1=1.6 // in Km
d1=1.6*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb1=20*log10(4*%pi*d1/lambda) // basic transmission loss in dB
disp(Lb1,"basic transmission loss in dB when d=1.6Km, f=3GHz:")
d2=16 // in Km
d2=16*10^3 // in meter
```

```
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb2=20*log10(4*%pi*d2/lambda) // basic transmission loss in dB
disp(Lb2,"basic transmission loss in dB when d=16Km, f=3GHz:")
d3=160 // in Km
d3=160*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb3=20*log10(4*%pi*d3/lambda) // basic transmission loss in dB
disp(Lb3,"basic transmission loss in dB when d=160Km, f=3GHz:")
d4=320 // in Km
d4=320*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb4=20*log10(4*%pi*d4/lambda) // basic transmission loss in dB
disp(Lb4,"basic transmission loss in dB when d=320Km, f=3GHz:")
```

Scilab code Exa 4.12 Actual transmission loss

```
//Exa 4.12
clc;
clear;
close;
// given :
Gt=10 // transmission gain in dB
Gr=10 // receiving gain in dB
//when frequency=0.3GHz
// given :
f=0.3 //frequency in Ghz
f=0.3*10^9 //frequency in hertz
c=3*10^8 //speed of light in m/s
lambda=c/f //wavelength in meter
```

```
d1=1.6 // in Km
d1=1.6*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb1=20*log10(4*%pi*d1/lambda) // basic transmission
    loss in dB
La1=Lb1-Gt-Gr // Actual transmission loss in dB
disp(La1,"Actual transmission loss in dB when d=1.6
    Km, f=0.3GHz:")
d2=16 // in Km
d2=16*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb2=20*log10(4*%pi*d2/lambda) // basic transmission
    loss in dB
La2=Lb2-Gt-Gr // Actual transmission loss in dB
disp(La2,"Actual transmission loss in dB when d=16Km
    , f=0.3GHz:")
d3=160 // in Km
d3=160*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb3=20*log10(4*%pi*d3/lambda) // basic transmission
    loss in dB
La3=Lb3-Gt-Gr // Actual transmission loss in dB
disp(La3,"Actual transmission loss in dB when d=160
    Km, f=0.3GHz:")
d4=320 // in Km
d4=320*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb4=20*log10(4*%pi*d4/lambda) // basic transmission
    loss in dB
La4=Lb4-Gt-Gr // Actual transmission loss in dB
disp(La4,"Actual transmission loss in dB when d=320
    Km, f=0.3GHz:")
// when frequency is 3Ghz
// given :
f=3 //frequency in Ghz
f=3*10^9 //frequency in hertz
c=3*10^8 //speed of light in m/s
lambda=c/f //wavelength in meter
```

```
d1=1.6 // in Km
d1=1.6*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb1=20*log10(4*%pi*d1/lambda) // basic transmission loss in dB
La1=Lb1-Gt-Gr // Actual transmission loss in dB
disp(La1,"Actual transmission loss in dB when d=1.6Km, f=3GHz:")
d2=16 // in Km
d2=16*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb2=20*log10(4*%pi*d2/lambda) // basic transmission loss in dB
La2=Lb2-Gt-Gr // Actual transmission loss in dB
disp(La2,"Actual transmission loss in dB when d=16Km, f=3GHz:")
d3=160 // in Km
d3=160*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb3=20*log10(4*%pi*d3/lambda) // basic transmission loss in dB
La3=Lb3-Gt-Gr // Actual transmission loss in dB
disp(La3,"Actual transmission loss in dB when d=160Km, f=3GHz:")
d4=320 // in Km
d4=320*10^3 // in meter
// formula : Lb=20*log10((4*(%pi)*d)/(lambda))
Lb4=20*log10(4*%pi*d4/lambda) // basic transmission loss in dB
La4=Lb4-Gt-Gr // Actual transmission loss in dB
disp(La4,"Actual transmission loss in dB when d=320Km, f=3GHz:")
```

Scilab code Exa 4.13 receiving power

```
//Exa 4.13
clc;
clear;
close;
// given :
Wt=15 // radaited power in watt
f=60 // in MHz
f=60*10^6 // in Hz
d=10 // in m
c=3*10^8 // in m/s
lambda=c/f // in meter
Gt=1.64 // transmitting gain in dB
Gr=1.64 // receiving gain in dB
Wr=(Wt*Gt*Gr*(lambda)^2/(4*(%pi)*d)^2) // receiving
   power in watt
disp(Wr*1000,"receiving power in mW:")
```

Chapter 6

HF VHF AND UHF ANTEENAS

Scilab code Exa 6.1 Designing of a rhombic antenna

```
//Exa 6.1
clc;
clear;
close;
//given :
f=30 // frequency in MHz
f=30*10^6 // frequency in Hz
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in meter
Delta=30 // angle of elevation in Degrees
H=lambda/(4*sind(Delta)) // Rhombic height in m
phi=90-Delta // tilt angle in Degrees
l=lambda/(2*(cosd(phi)^2)) // wire length in m
disp(H," Rhombic height in m:")
disp(phi,"Tilt angle in Degrees:")
disp(l,"length of wire in meter:")
```

Scilab code Exa 6.2 Designing of a rhombic antenna

```
//Exa 6.2
clc;
clear;
close;
//given :
f=20 //frequency in MHz
f=20*10^6 // frequency in Hz
c=3*10^8 //speed of light in m/s
lambda=c/f //wavelength in meter
Delta=10 // angle of elevation in Degrees
H=lambda/(4*sind(Delta)) // Rhombic height in m
phi=90-Delta // tilt angle in Degrees
l=lambda/(2*(cosd(phi)^2)) // wire length in m
disp(H," Rhombic height in m:")
disp(phi,"Tilt angle in Degrees:")
disp(l,"length of wire in meter:")
```

Scilab code Exa 6.3 Designing of a rhombic antenna

```
//Exa 6.3
clc;
clear;
close;
//given :
f=30 //frequency in MHz
f=30*10^6 //frequency in Hz
c=3*10^8 //speed of light in m/s
lambda=c/f //wavelength in meter

disp("for Delta=10 degrees")

Delta1=10 // angle of elevation in Degrees
H1=lambda/(4*sind(Delta1)) // Rhombic height in m
```

```
phi1=90-Delta1 // tilt angle in Degrees
l1=lambda/(2*(cosd(phi1)^2)) // wire length in m
disp(H1," Rhombic height in m:")
disp(phi1,"Tilt angle in Degrees:")
disp(l1,"length of wire in meter:")

disp("for Delta=15 degrees")

Delta2=15 // angle of elevation in Degrees
H2=lambda/(4*sind(Delta2)) // Rhombic height in m
phi2=90-Delta2 // tilt angle in Degrees
l2=lambda/(2*(cosd(phi2)^2)) // wire length in m
disp(H2," Rhombic height in m:")
disp(phi2,"Tilt angle in Degrees:")
disp(l2,"length of wire in meter:")

disp("for Delta=20 degrees")

Delta3=20 // angle of elevation in Degrees
H3=lambda/(4*sind(Delta3)) // Rhombic height in m
phi3=90-Delta3 // tilt angle in Degrees
l3=lambda/(2*(cosd(phi3)^2)) // wire length in m
disp(H3," Rhombic height in m:")
disp(phi3,"Tilt angle in Degrees:")
disp(l3,"length of wire in meter:")

disp("for Delta=25 degrees")

Delta4=25 // angle of elevation in Degrees
H4=lambda/(4*sind(Delta4)) // Rhombic height in m
phi4=90-Delta4 // tilt angle in Degrees
l4=lambda/(2*(cosd(phi4)^2)) // wire length in m
disp(H4," Rhombic height in m:")
disp(phi4,"Tilt angle in Degrees:")
disp(l4,"length of wire in meter:")

disp("for Delta=30 degrees")

```

```
Delta5=30 // angle of elevation in Degrees
H5=lambda/(4*sind(Delta5)) // Rhombic height in m
phi5=90-Delta5 // tilt angle in Degrees
l5=lambda/(2*(cosd(phi5)^2)) // wire length in m
disp(H5," Rhombic height in m:")
disp(phi5,"Tilt angle in Degrees:")
disp(l5,"length of wire in meter:")

disp("for Delta=35 degrees")

Delta6=35 // angle of elevation in Degrees
H6=lambda/(4*sind(Delta6)) // Rhombic height in m
phi6=90-Delta6 // tilt angle in Degrees
l6=lambda/(2*(cosd(phi6)^2)) // wire length in m
disp(H6," Rhombic height in m:")
disp(phi6,"Tilt angle in Degrees:")
disp(l6,"length of wire in meter:")

disp("for Delta=40 degrees")

Delta7=40 // angle of elevation in Degrees
H7=lambda/(4*sind(Delta7)) // Rhombic height in m
phi7=90-Delta7 // tilt angle in Degrees
l7=lambda/(2*(cosd(phi7)^2)) // wire length in m
disp(H7," Rhombic height in m:")
disp(phi7,"Tilt angle in Degrees:")
disp(l7,"length of wire in meter:")
```

Scilab code Exa 6.4 Design parameters of rhombic anteena

```
//Exa 6.4
clc;
clear;
close;
// given :
```

```
f=30 //frequency in MHz
f=30*10^6 //frequency in Hz
c=3*10^8 //speed of light in m/s
K=0.74 // constant
lambda=c/f // in meter
Delta=30 // angle of elevation in Degrees
H=lambda/(4*sind(Delta)) // Rhombic height in m
phi=90-Delta // tilt angle in Degrees
l=(lambda/(2*(cosd(phi)^2)))*K // wire length in m
disp(H," Rhombic height in m:")
disp(phi,"Tilt angle in Degrees:")
disp(l,"length of wire in meter:")
```

Scilab code Exa 6.5 Design parameters of rhombic anteena

```
//Exa 6.5
clc;
clear;
close;
// given :
f=20 //ferquency in MHz
f=20*10^6 //frequency in Hz
c=3*10^8 //speed of light in m/s
K=0.74 // constant
lambda=c/f //wavelength in meter
Delta=20 // angle of elevation in Degrees
H=lambda/(4*sind(Delta)) // Rhombic height in m
phi=90-Delta // tilt angle in Degrees
l=(lambda/(2*(cosd(phi)^2)))*K // wire length in m
disp(H,"Rhombic height in m:")
disp(phi,"Tilt angle in Degrees:")
disp(l,"length of wire in meter:")
```

Scilab code Exa 6.6 Design a three element yagi uda antenna

```
//Exa 6.6
clc;
clear;
close;
// given :
f_MHz=172 // frequency in MHz
c=3*10^8 // speed of light in m/s
lambda=c/f_MHz // wavelength in m
La=478/f_MHz // length of driven element in feet
Lr=492/f_MHz // length of reflector in feet
Ld=461.5/f_MHz // length of director in feet
S=142/f_MHz // element spacing in feet
disp(La,"length of driven element in feet:")
disp(Lr,"length of reflector in feet:")
disp(Ld,"length of director in feet:")
disp(S,"element spacing in feet:")
```

Scilab code Exa 6.7 Designing of a six element yagi uda antenna

```
//Exa6.7
clc;
clear;
close;
// given :
G=12 // required gain in dB
f=200 // frequency in MHz
f=200*10^6 // frequency in Hz
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in m
La=0.46*lambda // length of driven element in m    (
    note: in book La is given 0.416*lambda misprint)
Lr=0.475*lambda // length of reflector in m
Ld1=0.44*lambda // length of director1 in m
```

```
Ld2=0.44*lambda // length of director2 in m
Ld3=0.43*lambda // length of director3 in m
Ld4=0.40*lambda // length of director4 in m
SL=0.25*lambda // spacing between reflector and
   driver in m
Sd=0.31*lambda // spacing director and driving
   element in m
d=0.01*lambda // diameter of elements in m
l=1.5*lambda // length of array in m
disp(La,"length of driven element in m:")
disp(Lr,"length of reflector in m:")
disp(Ld1,"length of director1 in m:")
disp(Ld2,"length of director2 in m:")
disp(Ld3,"length of director3 in m:")
disp(Ld4,"length of director4 in m:")
disp(SL,"spacing between reflector and driver in m:"
   )
disp(Sd,"spacing director and driving element in m:"
   )
disp(d,"diameter of elements in m:")
disp(l,"length of array in m:")
```

Scilab code Exa 6.8 Designing of a long periodic antenna

```
//Exa 6.8
clc;
clear;
close;
// given :
G=9 // required gain in dB
f_l=125 // lowest frequency in MHz
f_l=125*10^6 // lowest frequency in Hz
f_h=500 // highest frequency in MHz
f_h=500*10^6 // lowest frequency in Hz
c=3*10^8 // speed of light in m/s
```

```
lambda_l=c/f_l // longest wavelength in m
lambda_s=c/f_h // shortest wavelength in m
tau=0.861 // scaling factor
sigma=0.162 // spacing factor
alpha=2*atand((1-tau)/(4*sigma)) // wedge angle in
    Degrees
L1=lambda_l/2 // in m
L2=tau*L1 // in m
L3=tau*L2 // in m
L4=tau*L3 // in m
L5=tau*L4 // in m
L6=tau*L5 // in m
L7=tau*L6 // in m
L8=tau*L7 // in m
L9=tau*L8 // in m
L10=tau*L9 // in m
L11=tau*L10 // in m

// element spacing relation
//formula : sn=2*sigma*Ln
S1=2*sigma*L1 // in m
S2=2*sigma*L2 // in m
S3=2*sigma*L3 // in m
S4=2*sigma*L4 // in m
S5=2*sigma*L5 // in m
S6=2*sigma*L6 // in m
S7=2*sigma*L7 // in m
S8=2*sigma*L8 // in m
S9=2*sigma*L9 // in m
S10=2*sigma*L10 // in m
S11=2*sigma*L11 // in m


disp("designing of log-periodic antenna:")

disp(L1,"L1 in m:")
disp(L2,"L2 in m:")
disp(L3,"L3 in m:")
```

```
disp(L4,"L4 in m:")
disp(L5,"L5 in m:")
disp(L6,"L6 in m:")
disp(L7,"L7 in m:")
disp(L8,"L8 in m:")
disp(L9,"L9 in m:")
disp(L10,"L10 in m:")
disp(L11,"L11 in m:")

disp("elements spacing relation:")
disp(S1,"S1 in m:")
disp(S2,"S2 in m:")
disp(S3,"S3 in m:")
disp(S4,"S4 in m:")
disp(S5,"S5 in m:")
disp(S6,"S6 in m:")
disp(S7,"S7 in m:")
disp(S8,"S8 in m:")
disp(S9,"S9 in m:")
disp(S10,"S10 in m:")
disp(S11,"S11 in m:")
```

Scilab code Exa 6.9 induced voltage in a loop antenna

```
//Exa 6.9
clc;
clear;
close;
// given :
E_rms=10 // electric field in mV/m
E_rms=10*10^-3 // electric field in V/m
f=2 // frequency in MHz
f=2*10^6 // frequency in Hz
N=10 // number of turns
phi=0 // angle between the plane of loop and
```

```
direction of incident wave in Degrees
S=1.4 // area of loop antenna in m^2
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in m
E_max=sqrt(2)*E_rms // electric field in V/m
V_rms=(2*%pi*E_max*S*N/lambda)*cosd(phi) // induced voltage
disp(V_rms*1000,"induced voltage in mV:")
```

Scilab code Exa 6.10 radiation resistance of a loop antenna

```
//Exa6.10
clc;
clear;
close;
//given :
D=0.5 // diameter of loop antenna in m
a=D/2 // radius of loop antenna in m
f=1 // frequency in MHz
f=1*10^6 // frequency in Hz
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in m
Rr=3720*(a/lambda) // radiation resistance of loop antenna in ohm
disp(Rr,"radiation resistance of loop antenna in ohm:")
```

Scilab code Exa 6.11 Directivity of a loop antenna

```
//Exa 6.11
clc;
clear;
close;
```

```
//given :
a=0.5 // radius of loop antenna in m
f=0.9 // frequency in MHz
f=0.9*10^6 // frequency in Hz
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in m
k=(2*%pi*a)/lambda // constant
disp("the value of k is:")
disp(k)
disp("since ,k<1/3")
disp("So Directivity of loop antenna is D=1.5")
```

Scilab code Exa 6.13 Directivity and radiation resistance of a loop antenna

```
//Exa 6.13
clc;
clear;
close;
Lm=poly(0,'Lm') // defining Lm as lambda
d=1.5*Lm // diameter of antenna in m
a=d/2 // radius of antenna in m
// formula : Rr=3720*(a/Lm)
Rr=3720*(a/Lm) // radiation resistance of loop
    antenna in ohm
Rr=horner(Rr,1)
// formula : D=4.25*(a/Lm)
D=4.25*(a/Lm)// Directivity of the loop antenna
D=horner(D,1)
disp(Rr,"radiation resistance of the loop antenna in
     ohm:")
disp(D,"Directivity of the loop antenna:")
```

Scilab code Exa 6.14 array length number of elements and null to null beam width

```
//Exa 6.14
clc;
clear;
close;
//given :
Gp=28 // power gain
Lm=poly(0,'Lm') // defining Lm as lambda
d=Lm/2 // length of dipole
//formula : Gp=4*(L/lambda)
L=Gp*Lm/4 // array length
disp(L,"array length(where Lm is wavelength in m):")
N=7*2 // Number of elements in the array when spaced
    at lambda/2
disp(N,"Number of elements in the array when spaced
    at lambda/2:")
// formula : B.W=2*sqrt((2*/N)*(lambda/d))
BW=2*sqrt(horner((2*Lm/(N*d)),1)) // null-to-null
    beam width in radians
BW_d=BW*180/%pi // null-to-null beam width in
    degrees
disp(BW_d,"null-to-null beam width in degrees:")



// Answer of null-to-null beam width in degrees is
    rounded-off in book.
```

Scilab code Exa 6.15 null to null and half power beam width and directivity

```
//Exa 6.15
clc;
```

```
clear;
close;
// given :
S=0.05 // spacing in m
Dh=0.1 // diameter of helical antenna in m
N=20 // number of turns
f=1000 // frequency in MHz
f=1000*10^6 // frequency in MHz
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in m
C=%pi*Dh // circumfrence of helix in m
La=N*S // axial legth in m
phi_not=(115*(lambda^(3/2))/(C*sqrt(La))) // B.W.F.N., null-to-null beamwidth of main beam in Degreess
phi=(52*lambda^(3/2)/(C*sqrt(La))) // H.P.B.W, half power beamwidth in Degreess
D=(15*N*C^2*S/(lambda)^3) // Directivity
disp(phi_not,"B.W.F.N., null-to-null beamwidth of main beam in Degrees:")
disp(phi,"H.P.B.W, half power beamwidth in Degrees:")
disp(D,"Directivity:")
```

Chapter 7

MICROWAVE ANTENNAS

Scilab code Exa 7.1 null to null and half power beam width of a paraboloid reflector

```
//Exa 7.1
clc;
clear;
close;
// given :
D=2 // Diameter of paraboloid reflector in m
c=3*10^8 // speed of light in m/s
f=5 // frequency in GHz
f=5*10^9 // frequency in Hz
lambda=c/f // wavelength in m
BWFN=140*(lambda/D) // null-to-null beamwidth in
    degrees
HPBW=70*(lambda/D) // half power beamwidth in
    degrees
disp(BWFN,"null-to-null beamwidth in degrees:")
disp(HPBW,"half power beamwidth in degrees:")
```

Scilab code Exa 7.2 gain of the paraboloid reflector antenna

```
//Exa 7.2
clc;
clear;
close;
// given :
D=2 // mouth diameter of paraboloid reflector in m
c=3*10^8 // speed of light in m/s
f=5 // frequency in GHz
f=5*10^9 // frequency in Hz
lambda=c/f // wavelength in m
G=6.4*(D/lambda)^2 // power gain of paraboloid
G_p=10*log10(G) //power gain in dB
disp(G_p,"power gain in dB:")
```

Scilab code Exa 7.3 band width between first null and half power points

```
//Exa 7.3
clc;
clear;
close;
// given :
D_a=0.15 // mouth Diameter of paraboloid in m
c=3*10^8 // speed of light in m/s
f=10 // frequency in GHz
f=10*10^9 // frequency in Hz
lambda=c/f // wavelength in m
BWFN=140*(lambda/D_a) // null-to-null beamwidth in
    degrees
HPBW=70*(lambda/D_a) // half power beamwidth in
    degrees
disp(BWFN,"null-to-null beamwidth in degrees:")
disp(HPBW,"half power beamwidth in degrees:")
G_p=6.4*(D_a/lambda)^2 // power gain of paraboloid
G_p=10*log10(G_p) // power gain in dB
disp(G_p,"power gain in dB")
```

Scilab code Exa 7.4 Power gain in Decibels

```
//Exa 7.4
clc;
clear;
close;
// given :
D_a=1.8 //mouth diameter of paraboloid reflector in
   m
c=3*10^8 // speed of light in m/s
f=2 // frequency in GHz
f=2*10^9 // frequency in Hz
lambda=c/f // wavelength in m
G_p=6.4*(D_a/lambda)^2 // power gain of paraboloid
G_p=10*log10(G_p) // power gain in dB
disp(G_p,"power gain in dB")
```

Scilab code Exa 7.5 mouth diameter HPBW and power gain

```
//Exa 7.5
clc;
clear;
close;
// given :
c=3*10^8 // speed of light in m/s
f=5 // frequency in GHz
f=5*10^9 // frequency in Hz
lambda=c/f // wavelength in m
BWFN=10 // null-to-null beamwidth in degrees
// formula: BWFN=140*(lambda/D_a)
D_a=140*lambda/BWFN // mouth Diameter of paraboloid
    reflector in m
```

```
disp(D_a,"mouth Diameter of paraboloid reflector in
   m:")
HPBW=70*(lambda/D_a) // half power beamwidth in
   degrees
disp(HPBW,"half power beamwidth in degrees:")
G_p=6.4*(D_a/lambda)^2 // power gain of paraboloid
disp(G_p,"power gain of paraboloid:")
```

Scilab code Exa 7.6 beam width directivity and capoture area

```
//Exa 7.6
clc;
clear;
close;
// given :
b=0.65 // illumination efficiency
D_a=6 // mouth diameter of paraboloid reflector in m
c=3*10^8 // speed of light in m/s
f=10 // frequency in GHz
f=10*10^9 // frequency in Hz
lambda=c/f // wavelength in m
A=%pi*(D_a)^2/4 // Actual area in m^2
A_c=0.65*A // capture area in m^2
D=6.4*(D_a/lambda)^2 // directivity
D=10*log10(D) // directivity in dB
phi=70*(lambda/D_a) // half power beam width in
   degrees
phi_not=2*phi // null-to-null main beam width in
   degrees
disp(D,"directivity in dB:")
disp(phi,"half power beam width in degrees:")
disp(phi_not,"null-to-null main beam width in
   degrees:")
disp(A_c,"capture area in m^2:")
```

Scilab code Exa 7.7 minimum distance required between two antennas

```
//Exa 7.7
clc;
clear;
close;
// given :
D_a=6 // Diameter of paraboloid reflector in m
c=3*10^8 // speed of light in m/s
f=4 // frequency in GHz
f=4*10^9 // frequency in Hz
lambda=c/f // wavelength in m
r=2*D_a^2/lambda // required minimum distance
    between two antennae in m
disp(r,"required minimum distance between two
    antennae in m:")
```

Scilab code Exa 7.8 mouth diameter and beam width

```
//Exa 7.8
clc;
clear;
close;
// given :
G_p=1000 // gain
c=3*10^8 // speed of light in m/s
f=3 // frequency in GHz
f=3*10^9 // frequency in Hz
lambda=c/f // wavelength in m
// formula : G_p=6.4*(D_a/lambda)^2 // power gain
D_a=lambda*(sqrt(G_p/6.4)) // mouth Diameter of
    paraboloid in m
```

```
BWFN=140*(lambda/D_a) // null-to-null beamwidth in degrees
HPBW=70*(lambda/D_a) // half power beamwidth in degrees
disp(D_a,"mouth Diameter of paraboloid in m")
disp(BWFN,"null-to-null beamwidth in degrees:")
disp(HPBW,"half power beamwidth in degrees:")
```

Scilab code Exa 7.9 capture area and beam width of paraboloid antenna

```
//Exa 7.9
clc;
clear;
close;
// given :
c=3*10^8 // speed of light in m/s
f=10 // frequency in GHz
f=10*10^9 // frequency in Hz
lambda=c/f // wavelength in m
G_p=75 // power gain in dB
// formula : G_p=10*log10(G_p) // power gain in dB
G=10^(G_p/10) // simple power gain
// formula : G=6.4*(D_a/lambda)^2 // power gain
D_a=lambda*(sqrt(G/6.4)) // mouth Diameter of paraboloid in m
A=%pi*(D_a)^2/4 // Actual area in m^2
A_c=0.65*A // capture area in m^2
BWFN=140*(lambda/D_a) // null-to-null beamwidth in degrees
HPBW=70*(lambda/D_a) // half power beamwidth in degrees
disp(BWFN,"null-to-null beamwidth in degrees:")
disp(HPBW,"half power beamwidth in degrees:")
disp(A_c,"capture area in m^2:")

```

```
//note : answer of A_c in book is 2269.83 m^2  but
    in scilab 2270.20 m^2
```

Scilab code Exa 7.10 HPBW BWFN and power gain

```
//Exa 7.10
clc;
clear;
close;
// given :
D_a=60 // mouth diameter of paraboloid reflector in
    m
c=3*10^8 // speed of light in m/s
f=2 // frequency in GHz
f=2*10^9 // frequency in Hz
lambda=c/f // wavelength in m
phi=70*(lambda/D_a) // half power beam width in
    degrees
phi_not=140*(lambda/D_a) // null-to-null main beam
    width in degrees
disp(phi,"half power beam width in degrees:")
disp(phi_not,"null-to-null main beam width in
    degrees:")
G_p=6.4*(D_a/lambda)^2 // power gain of paraboloid
G_p=10*log10(G_p) //power gain in dB
disp(G_p,"power gain in dB:")
```

Scilab code Exa 7.11 power gain

```
//Exa 7.11
clc;
```

```
clear;
close;
// given :
D=22 // mouth diameter of paraboloid reflector in m
c=3*10^8 // speed of light in m/s
f=5 // frequency in GHz
f=5*10^9 // frequency in Hz
lambda=c/f // wavelength in m
b=0.6 // illumination efficiency
G_p=b*(D/lambda)^2 // power gain of paraboloid
G_p=10*log10(G_p) //power gain in dB
disp(G_p,"power gain in dB:")
```

Scilab code Exa 7.12 mouth diameter and capture area of a paraboloid antenna

```
//Exa 7.12
clc;
clear;
close;
// given :
c=3*10^8 // speed of light in m/s
f=2 // frequency in GHz
f=2*10^9 // frequency in Hz
lambda=c/f // wavelength in m
BWFN=12 // null-to-null main beam width in degrees
 // formula : BWFN=140*(lambda/D_a)
D_a=140*lambda/BWFN // mouth diameter of paraboloid
    reflector in m
A=%pi*(D_a)^2/4 // Actual area in m^2
A_c=0.65*A // capture area in m^2
disp(D_a,"mouth diameter of paraboloid reflector in
   m:")
disp(A_c,"capture area in m^2:")
```

Scilab code Exa 7.13 mouth diameter and power gain of paraboloid reflector antenna

```
//Exa 7.13
clc;
clear;
close;
// given :
c=3*10^8 // speed of light in m/s
f=2.5 // frequency in GHz
f=2.5*10^9 // frequency in Hz
lambda=c/f // wavelength in m
BWFN=3 // null-to-null main beam width in degrees
 // formula : BWFN=140*(lambda/D_a)
D_a=140*lambda/BWFN // mouth diameter of paraboloid
   reflector in m
G=6.4*(D_a/lambda)^2 // power gain of paraboloid
G_p=10*log10(G) //power gain in dB
disp(G_p,"power gain in dB:")
disp(D_a,"mouth diameter of paraboloid reflector in
   m:")
```

Scilab code Exa 7.14 null to null beam width and power gain of paraboloid reflector antenna

```
// Exa7.14
clc;
clear;
close;
// given :
phi=5 // HPBW, half power beam width in Degrees
```

```
phi_not=2*phi // BWFN, null-to-null beam width in
    degrees
Lm=poly(0,'Lm') // defining Lm as lambda
// formula : phi=70*(Lm/D_a) // where Lm is
    wavelength in m and D_a is mouth diameter in m
D_a=(70*Lm)/phi
G_p=6.4*(D_a/Lm)^2
G_p=horner(G_p,1)
G_p=10*log10(G_p) // power gain in dB
disp(phi_not,"BWFN, null-to-null beam width in
    degrees:")
disp(G_p,"power gain in dB:")
```

Scilab code Exa 7.15 Power gain of a paraboloid reflector antenna

```
// Exa 7.15
clc;
clear;
close;
Lm=poly(0,'Lm')// defining Lm as lambda
D_a=8*Lm // where D_a is mouth diameter in m and Lm
    is wavelength in m
// formula : G_p=6.4*(D/lambda)^2
G_p=6.4*(D_a/Lm)^2 //power gain
G_p=horner(G_p,1)
G_p=10*log10(G_p) // power gain in dB
disp(G_p,"power gain in dB:")
```

Scilab code Exa 7.16 null to null and half power beam width and directivity

```
//Exa 7.16
clc;
```

```
clear;
close;
Lm=poly(0,'Lm') // defining Lm as lambda
D_a=6*Lm // where D_a is mouth diameter in m and Lm is wavelength
// formula : HPBW=phi=70*(lambda/D_a)
phi=70*(Lm/D_a) // half power beam width in degrees
phi=horner(phi,1)
phi_not=2*phi // null-to-null beam width in degrees
// formula : D=6.4*(D_a/lambda)^2
D=6.4*(D_a/Lm)^2
D=horner(D,1)
disp(D,"Directivity:")
disp(phi,"half power beam width in degrees:")
disp(phi_not,"null-to-null beam width in degrees:")
```

Scilab code Exa 7.17 beam width power gain and directivity

```
//Exa 7.17
clc;
clear;
close;
// given :
f=6 // frequency in GHz
f=6*10^9 // frequency in Hz
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in m
d=12 // aperture length in cm
d=12*10^-2 // aperture length in m
w=6 // aperture width in cm
w=6*10^-2 // aperture width in m
phi_E=56*(lambda/d) // half power beam width for aperture length d in Degrees
phi_H=67*(lambda/w) // half power beam width for aperture width w in Degrees
```

```
G_p=(4.5*w*d)/(lambda)^2 // power gain
G_p=10*log10(G_p) // power gain in dB
D=(7.5*w*d)/(lambda)^2 // Directivity
disp(phi_E,"half power beam width for aperture
   length d in Degrees:")
disp(phi_H,"half power beam width for aperture width
    w in Degrees:")
disp(G_p,"power gain in dB:")
disp(D,"Directivity:")
```

Scilab code Exa 7.18 power gain of a square horn antenna

```
// Exa 7.18
clc;
clear;
close;
Lm=poly(0,'Lm') // defining Lm as lambda
d=8*Lm // where d is aperture length and Lm is
   wavelength
w=8*Lm // where w is aperture width
//formula : G_p=(4.5*w*d)/lambda^2
G_p=(4.5*w*d)/Lm^2 // power gain
G_p=horner(G_p,1)
G_p=10*log10(G_p) // power gain in dB
disp(G_p,"power gain in dB:")
```

Scilab code Exa 7.19 power gain and directivity of a horn antenna

```
//Exa 7.19
clc;
clear;
close;
// given :
```

```
f=6 // frequency in GHz
f=6*10^9 // frequency in Hz
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in m
d=10 // aperture length in cm
d=10*10^-2 // aperture length in m
w=5 // aperture width in cm
w=5*10^-2 // aperture width in m
G_p=(4.5*w*d)/(lambda)^2 // power gain
G_p=10*log10(G_p) // power gain in dB
D=(7.5*w*d)/(lambda)^2 // Directivity
D=10*log10(D) // directivity in dB
disp(G_p,"power gain in dB:")
disp(D,"Directivity in dB:")
```

Scilab code Exa 7.20 complementary slot impedances

```
//Exa 7.20
clc;
clear;
close;
// given :
eta_0=377 //intrinsic impedance in ohm
disp("when Zd=73+%i*42.5")
Zd=73+%i*42.5 // dipole impedance
// formula : zs*zd=(eta_0)^2/4
Zs=eta_0^2/(4*Zd) // slot impedance in ohm
disp(Zs,"complementary slot impedance in ohm:")

disp("when Zd=67+%i*0")
Zd=67+%i*0 // dipole impedance
// formula : zs*zd=(eta_0)^2/4
Zs=eta_0^2/(4*Zd) // slot impedance in ohm
disp(Zs,"complementary slot impedance in ohm:")

```

```
disp("when Zd=710+%i*0")
Zd=710+%i*0 // dipole impedance
// formula : zs*zd=(eta_0)^2/4
Zs=eta_0^2/(4*Zd) // slot impedance in ohm
disp(Zs,"complementary slot impedance in ohm:")


disp("when Zd=500+%i*0")
Zd=500+%i*0 // dipole impedance
// formula : zs*zd=(eta_0)^2/4
Zs=eta_0^2/(4*Zd) // slot impedance in ohm
disp(Zs,"complementary slot impedance in ohm:")


disp("when Zd=50+%i*20")
Zd=50+%i*20 // dipole impedance
// formula : zs*zd=(eta_0)^2/4
Zs=eta_0^2/(4*Zd) // slot impedance in ohm
disp(Zs,"complementary slot impedance in ohm:")


disp("when Zd=50-%i*25")
Zd=50-%i*25 // dipole impedance
// formula : zs*zd=(eta_0)^2/4
Zs=eta_0^2/(4*Zd) // slot impedance in ohm
disp(Zs,"complementary slot impedance in ohm:")


disp("when Zd=300+%i*0")
Zd=300+%i*0 // dipole impedance
// formula : zs*zd=(eta_0)^2/4
Zs=eta_0^2/(4*Zd) // slot impedance in ohm
disp(Zs,"complementary slot impedance in ohm:")
```

Chapter 9

WAVE PROPAGATION

Scilab code Exa 9.1 Required transmitter power

```
//Exa 9.1
clc;
clear;
close;
// given :
f=1.7 //frequency in MHz
f=1.7*10^6 //frequency in Hz
E=0.5 //electric field in mV/m
E=0.5*10^(-3) //electric field in V/m
sigma=5*10^-5 // conductivity in mho/cm
eta=0.5 // efficiency
epsilon_r=10 //relative permittivity
c=3*10^8 // speed of light in m/s
lambda=c/f //wavelength in m
d=10 // distance in km
d=10*10^3 // distance in m
Df=1.8*10^12*(sigma/f) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in degrees
p=(%pi/Df)*(d/lambda)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp
```

```
((-5*p)/8))) // factor of the ground losses
// formula: E=(A*E_not)/d
//E_not=300*sqrt(P_kW) // E_not is in mV/m
//E_not=300*sqrt(P_kW)*10^(-3) // E_not is in V/m
P_kW=(E*d*10^-3/(A*300*10^-3))^2 // Power in kW and d is in km
// formula : P=Ptx*eta
Ptx=P_kW/eta //transmitter power required in kW
disp(Ptx*1000,"transmitter power required in W:")


//note: answer in the book is wrong.scilab give Ptx=11.48W whereas answer in the book is Ptx=12.6W
```

Scilab code Exa 9.2 field strength of the ground wave

```
//Exa 9.2
clc;
clear;
close;
// given :
disp("for frequency=500kHz")
f1=500*10^3 //frequency in Hz
c=3*10^8 //speed of light in m/s
lambda1=c/f1 //  wavelength in m
d=100 //distance in km
d=100*10^3 //distance in m
Pt=100 // transmitter power in kW
eta=50 // efficiency in percentage
eta=0.5 // efficiency
P_kW=Pt*eta // radiated power in kW
E_not=300*1.28*sqrt(P_kW) // field strength in mV/m without ground losses at 1km

 // for sea water earth
```

```
epsilon_r=81 // permittivity
sigma=45*10^-3 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f1) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in
    degrees
p=(%pi/Df)*(d/lambda1)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp
    ((-5*p)/8))) // factor of the ground losses
E1=(A*E_not)/(d) // in V/m
disp(E1*1000,"electric field strength for sea water
    earth in mV/m when f=500kHz:")

// for good soil
epsilon_r=20 // permittivity
sigma=10^-4 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f1) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in
    degrees
p=(%pi/Df)*(d/lambda1)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp
    ((-5*p)/8))) // factor of the ground losses
E2=(A*E_not)/(d) // in V/m
disp(E2*1000,"electric field strength for good soil
    in mV/m when f=500kHz:")

// for poor soil
epsilon_r=10 // permittivity
sigma=0.2*10^-4 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f1) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in
    degrees
p=(%pi/Df)*(d/lambda1)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp
    ((-5*p)/8))) // factor of the ground losses
E3=(A*E_not)/(d) // in V/m
disp(E3*1000,"electric field strength for poor soil
    in mV/m when f=500kHz:")

```

```
// for cities,industrial areas
epsilon_r=5 // permittivity
sigma=10^-5 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f1) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in degrees
p=(%pi/Df)*(d/lambda1)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp((-5*p)/8))) // factor of the ground losses
E4=(A*E_not)/(d) // in V/m
disp(E4*1000,"electric field strength for cities,industrial areas in mV/m when f=500kHz:")

// for Rockey soil,flat sandy
epsilon_r=10 // permittivity
sigma=2*10^-3 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f1) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in degrees
p=(%pi/Df)*(d/lambda1)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp((-5*p)/8))) // factor of the ground losses
E5=(A*E_not)/(d) // in V/m
disp(E5*1000,"electric field strength for Rockey soil,flat sandy in mV/m when f=500kHz:")

// for medium hills,forestation
epsilon_r=13 // permittivity
sigma=5*10^-5 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f1) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in degrees
p=(%pi/Df)*(d/lambda1)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp((-5*p)/8))) // factor of the ground losses
E6=(A*E_not)/(d) // in V/m
disp(E6*1000,"electric field strength for medium hills,forestation in mV/m when f=500kHz:")
```

```



disp("for frequency=1500kHz")

f2=1500 // in kHz
f2=1500*10^(3) // in Hz
lambda2=c/f2 // in m
 // for sea water earth
epsilon_r=81 // permittivity
sigma=45*10^-3 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f2) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in degrees
p=(%pi/Df)*(d/lambda2)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp((-5*p)/8))) // factor of the ground losses
E1=(A*E_not)/(d) // in V/m
disp(E1*1000,"electric field strength for sea water earth in mV/m when f=1500kHz:")

// for good soil
epsilon_r=20 // permittivity
sigma=10^-4 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f2) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in degrees
p=(%pi/Df)*(d/lambda2)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp((-5*p)/8))) // factor of the ground losses
E2=(A*E_not)/(d) // in V/m
disp(E2*1000,"electric field strength for good soil in mV/m when f=1500kHz:")

// for poor soil
epsilon_r=10 // permittivity
sigma=0.2*10^-4 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f2) // dissipation factor
```

```
b=atand((epsilon_r+1)/Df) // phase constant in
    degrees
p=(%pi/Df)*(d/lambda2)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp
    ((-5*p)/8))) // factor of the ground losses
E3=(A*E_not)/(d) // in V/m
disp(E3*1000,"electric field strength for poor soil
    in mV/m when f=1500kHz:")

// for cities,industrial areas
epsilon_r=5 // permittivity
sigma=10^-5 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f2) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in
    degrees
p=(%pi/Df)*(d/lambda2)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp
    ((-5*p)/8))) // factor of the ground losses
E4=(A*E_not)/(d) // in V/m
disp(E4*1000,"electric field strength for cities,
    industrial areas in mV/m when f=1500kHz:")

// for Rockey soil,flat sandy
epsilon_r=10 // permittivity
sigma=2*10^-3 // conductivity in mho/cm
Df=1.8*10^12*(sigma/f2) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in
    degrees
p=(%pi/Df)*(d/lambda2)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp
    ((-5*p)/8))) // factor of the ground losses
E5=(A*E_not)/(d) // in V/m
disp(E5*1000,"electric field strength for Rockey
    soil,flat sandy in mV/m when f=1500kHz:")

// for medium hills,forestation
epsilon_r=13 // permittivity
sigma=5*10^-5 // conductivity in mho/cm
```

```
Df=1.8*10^12*(sigma/f2) // dissipation factor
b=atand((epsilon_r+1)/Df) // phase constant in
    degrees
p=(%pi/Df)*(d/lambda2)*cosd(b) // numerical distance
A=((2+0.3*p)/(2+p+0.6*p^2))-(sind(b)*(sqrt(p/2)*exp
    ((-5*p)/8))) // factor of the ground losses
E6=(A*E_not)/(d) // in V/m
disp(E6*1000,"electric field strength for medium
    hills,forestation in mV/m when f=1500kHz:")



// note1 : misprint value of sigma in part (e) when
    f=500khz correct is sigma=2*10^-3 for rockey soil
//note 2:  The ans is rounded off at the F = 1500
    kHz in poor soil and Cities, industrial area.
```

Scilab code Exa 9.3 maximum range of tropospheric transmission

```
//Exa9.3
clc;
clear;
close;
//given :
Ht=100 // height of transmittin antenna in feet
Hr=50 // height of receiving antenna in feet
Dmax=sqrt(2*Ht)+sqrt(2*Hr) // in miles
disp(Dmax,"maximum range of tropospheric
    transmission in miles:")
```

Scilab code Exa 9.4 Radio horizon distance

```
//Exa 9.4
```

```
clc;
clear;
close;
//given :
Ht=80 // height of transmitting antenna in meter
d=sqrt(17*Ht) // radio horizon distance of transmitting antenna
disp(d,"radio horizon distance of transmitting antenna in Km")
```

Scilab code Exa 9.5 maximum distance covered by the space wave

```
//Exa 9.5
clc;
clear;
close;
//given :
Ht=80 // height of transmitting antenna in meter
Hr=50 // height of receiving antenna in meter
Dmax=sqrt(17*Ht)+sqrt(17*Hr) // in Km
disp(Dmax,"maximum range of tropospheric transmission in Km:")
```

Scilab code Exa 9.6 Required height of receiving antenna

```
//Exa 9.6
clc;
clear;
close;
//given :
Dmax=80 //distance in Km
Ht=100 // height of transmitting antenna in meter
// formula : Dmax=sqrt(17*Ht)+sqrt(17*Hr)
```

```
Hr=(Dmax-(sqrt(17*Ht)))^2/17 // height of receiving
    antenna in m
disp(Hr,"height of receiving antenna in m")
```

Scilab code Exa 9.7 Radio horizon distance for transmitting and receiving antenna and maximum range

```
//Exa 9.7
clc;
clear;
close;
// given ;
Ht=300 // height of antenna in feet
Hr=100 // height of receiving antenna in feet
dt=sqrt(2*Ht) // radio horizon distance for a
    transmitting antenna in miles
dr=sqrt(2*Hr) // radio horizon distance for a
    transmitting antenna in miles
dmax=dt+dr // maximum range of space wave
    propagation in miles
disp(dt,"radio horizon distance for a transmitting
    antenna in miles:")
disp(dr,"radio horizon distance for a receiving
    antenna in miles:")
disp(dmax,"maximum range of space wave propagation
    in miles:")
```

Scilab code Exa 9.8 range of the space wave

```
//Exa 9.8
clc;
clear;
close;
```

```
// given :
f=60 // in MHz
f=60*10^6 // in Hertz
c=3*10^8 // speed of light in m/s
lambda=c/f // wavelength in meter
Ptx=1 // transmitting power in kilo watt
ht=50 // height of transmiting antenna in meter
hr=5 // height of receiving antenna in meter
E=80 // electric field in micro V/m
E=80*10^(-6) // electric field in V/m
E0=3*137.6*sqrt(Ptx)*(8/5)*10^3 // field in mV/m at
    1 meter where Ptx is in kW
E0=3*137.6*sqrt(Ptx)*(8/5)*10^3*10^(-3) // field in
    V/m at 1 meter where Ptx is in kW
// formula : E=(4*(%pi)*ht*hr*E0)/(lambda*d^2)
d=sqrt((4*%pi*ht*hr*E0)/(lambda*E)) // range of
    space wave in meter
disp(d/1000,"range of space wave in Km:")
```

Scilab code Exa 9.9 maximum wavelength at which propagation is possible

```
//Exa 9.9
clc;
clear;
close;
//given :
hd=30 // height of duct in m
delta_M=30 // unitless
LAMBDA_max=2.5*hd*sqrt(delta_M*10^-6) // maximum
    wavelength at which duct propagation is possible
disp(LAMBDA_max,"maximum wavelength at which duct
    propagation is possible in m:")
```

Scilab code Exa 9.10 Electron density of the layer

```
//Exa 9.10
clc;
clear;
close;
// given :
fc=1.5 // critical frequency in MHz
fc=1.5*10^(6) // critical frequency in Hz
// formula : fc=9*sqrt(Nmax)
Nmax=(fc)^2/81 // electron density in electrons/m^3
disp(Nmax,"electron density in electrons/m^3")
```

Scilab code Exa 9.11 range if the frequency is MUF itself

```
// Exa 9.11
clc;
clear;
close;
// given :
n=0.92 // refrective index
MUF=10 // maximum usable frequency in MHz
MUF=10*10^6 // maximum usable frequency in Hz
f=10*10^6 // in Hz ordinary frequency and maximum
    usable frequency are same
h=400 // height of ray reflection point on the
    ionospheric layer in Km
h=400*10^3 // height of ray reflection point on the
    ionospheric layer in m
// formula :n=sqrt(1-(81*Nmax/f^2))
Nmax=(1-n^2)*f^2/81 // electron density in electrons
    /m^3
```

```
fc=9*sqrt(Nmax) // critical frequency in Hz
// MUF=fc*sec(thetai)
//sec(thetai)=MUF/fc
// also, sec(thetai)=sqrt(h^2+(d^2/4))/h so on
   comparing,
d=sqrt(((MUF*h/fc)^2-h^2)*4) // range in km
disp(d/1000,"Range in Km")


// note :answer in the book is 1876.59 where as in
   scilab is 1877.94 minute difference only
```

Scilab code Exa 9.12 Relative permittivity of D E F layers

```
//Exa 9.12
clc;
clear;
close;
// given ;
f=50 // in hz
f=50*10^3 // in KHz
N1=400 // electron density of D layer in electrons/
   cm^3
N2=5*10^5 // electron density of E layer in
   electrons/cm^3
N3=2*10^6 // electron density of F layer in
   electrons/cm^3
// formula :n=sqrt(epsilon_r)=sqrt(1-(81*N/f^2)) //
   WHERE f IS IN KhZ
// for D layer
epsilon_r1=1-(81*N1/f^2) // relative permittivity of
   D layer
// for E layer
epsilon_r2=1-(81*N2/f^2) // relative permittivity of
   E layer
```

```
// for F layer
epsilon_r3=1-(81*N3/f^2) // relative permittivity of
    F layer
disp(round(epsilon_r1),"relative permittivity of D
   layer:")
disp(epsilon_r2,"relative permittivity of E layer:")
disp(epsilon_r3,"relative permittivity of F layer:")
```

Scilab code Exa 9.13 Angle of refraction

```
// Exa 9.13
clc;
clear;
close;
//given :
f=50 // in hz
f=50*10^3 // in KHz
theta_i=30 // in degrees
N=400 // electron density of D layer in electrons/cm
   ^3
// formula :n=sqrt(epsilon_r)=sqrt(1-(81*N/f^2)) //
   WHERE f IS IN KhZ
// for D layer
epsilon_r=1-(81*N/f^2) // relative permittivity of D
    layer
n=sqrt(epsilon_r) // refrective index
// formula :nsin(theta_r)=sin(theta_i) // snell's
   law
theta_r=asind(sind(theta_i/n)) // angle of
   refraction in degrees
disp(theta_r,"angle of refraction in degrees:")
```

Scilab code Exa 9.14 critical frequency of an electromagnetic wave

```
//Exa 9.14
clc;
clear;
close;
// given :
N1=400 // electron density of D layer in electrons/
    cm^3
N2=5*10^5 // electron density of E layer in
    electrons/cm^3
N3=2*10^6 // electron density of F layer in
    electrons/cm^3
// formula : fc=9*sqrt(N)
fc1=9*sqrt(N1) // critical frequency in Khz of EM
    wave for D layer
fc2=9*sqrt(N2) // critical frequency in MHz of EM
    wave for E layer
fc3=9*sqrt(N3) // critical frequency in MHz of EM
    wave for F layer
disp(fc1,"critical frequency of EM wave for D layer
    in kHz:")
disp(fc2/10^3,"critical frequency of EM wave for E
    layer in Mhz:")
disp(fc3/10^3,"critical frequency of EM wave for F
    layer in Mhz:")



//note: the value of fc3 in book is equal to 12.8MHz
     but the correct is 12.72MHz.
```

Scilab code Exa 9.15 Critical frequency

```
// Exa 9.15
clc;
clear;
```

```
close;
// given :
Nmax=1.3*10^6 // maximum electron density in electrons/cm^3
// formula : fc=9*sqrt(Nmax)
fc_khz=9*sqrt(Nmax) // critical frequency in Khz
disp(fc_khz/1000,"critical frequency in Mhz:")
```

Scilab code Exa 9.16 maximum usable frequency

```
//Exa 9.16
clc;
clear;
close;
// given :
d=2600 // distance between the points in Km
d=2600*10^3 // distance between the points in m
fc=4 // critical frequency in MHz
fc=4*10^6 // critical frequency in Hz
h=200 // height of ionospheric layer in km
h=200*10^3 // height of ionospheric layer in m
MUF=fc*sqrt(1+(d/(2*h))^2) // maximum usable frequency  (this step is Misprinted in the book)
disp(MUF/10^6,"maximum usable frequency in MHz:")
//note: Answer in the book is wrong.
```

Scilab code Exa 9.17 frequency of propagating wave for D layer

```
// Exa 9.17
```

```
clc;
clear;
close;
// given :
N=400 // electron density in electrons/cm^3
n=0.5 // refrective index
// formula : n=sqrt(1-(81*N/f^2))
f=sqrt(81*N/(1-n^2)) // frequency in kHz
disp(f,"frequency of propagating wave in kHz:")
```

Scilab code Exa 9.18 Range of line of sight

```
//Exa 9.18
clc;
clear;
close;
//given :
Ht=60 // height of transmitting antenna in meter
Hr=6 // height of receiving antenna in meter
d=sqrt(17*Ht)+sqrt(17*Hr) // in Km
disp(d,"range of line of sight in Km:")
```

Scilab code Exa 9.19 Critical angle of propagation for D layer

```
//Exa 9.19
clc;
clear;
close;
// given :
d=500 // distance between transmitter and receiver
    in km
h=70 // height of D layer in km
```

```
theta_c=asind(h/(sqrt(h^2+(d^2/4)))) // critical
    angle in degrees
disp(theta_c,"critical angle of propagation in
    degrees:")



//it can also be calculated from
theta_c=atand(2*h/d) // critical angle in degrees
disp(theta_c,"critical angle of propagation in
    degrees by method 2:")
```