Scilab Textbook Companion for The Fundamentals of Engineering Physics by P. S. Khare and A. Swarup[1]

Created by
Ranjeev Salathia
Physics
Mechanical Engineering
National Institute of Technology Calicut
College Teacher
Dr. kamni
Cross-Checked by
K. V. P. Pradeep

June 21, 2014

[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: The Fundamentals of Engineering Physics

Author: P. S. Khare and A. Swarup

Publisher: Laxmi Publications, New Delhi

Edition: 2

Year: 2008

ISBN: 978-81-318-0274-8

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

Quantum Physics

Scilab code Exa 1.1 de Broglie wavelength of a golf ball and sub atomic particles

```
// Scilab code Ex1.1 : Pg:18 (2008)
clc;clear;
e = 1.6e-019;    // Energy equivalent of 1 eV, J
m = 0.05;    // Mass of the golf ball, kg
v = 20;    // Velocity of golf ball, m/s
h = 6.625e-034;    // Planck's constant, joule-sec
Lambda1 = h/(m*v);    // de Broglie wavelength of a
   golf ball, m
m = 1.67e-027;    // mass of proton, kg
v = 2200;    // Velocity of proton, m/s
Lambda2 = h/(m*v);    // de Broglie wavelength of a
   proton, m
E = 10*e;    // Kinetic energy of an electron, eV
m = 9.11e-031;    // Mass of electron, kg
Lambda3 = h/sqrt(2*m*E);    // de Broglie wavelength
    of an electron, m
printf("\nThe de-Broglie wavelength of a golf ball =
    %5.3e m", Lambda1);
printf("\nThe de-Broglie wavelength of a proton = %4
   .2e m", Lambda2);
```

```
printf("\nThe de-Broglie wavelength of a electron = %3.1f m", Lambda3/1e-010);

// Result
// The de-Broglie wavelength of a golf ball = 6.625e-034 m
// The de-Broglie wavelength of a proton = 1.80e-010 m
// The de-Broglie wavelength of a electron = 3.9 m
```

Scilab code Exa 1.2 de Broglie wavelength of an electron

```
// Scilab code Ex1.2: : Pg:19 (2008)
clc; clear;
V = 100;    // potential difference, volt
Lambda = 12.25/sqrt(V);    // de Broglie wavelength, angstorm
printf("\nThe de-Broglie wavelength of an electron = %5.3f angstorm", Lambda);

// Result
// The de-Broglie wavelength of an electron = 1.225 angstorm
```

Scilab code Exa 1.3 de Broglie wavelength of a proton

```
// Scilab code Ex1.3: : Pg:19 (2008)
clc;clear;
m = 1.67e-027;    // Mass of proton, kg
h = 6.62e-034;    // Planck's constant, joule-sec
c = 3e+08;    // Velocity of light, m/s
v = c/20;    // Velocity of proton, m/sec
```

```
Lambda = h/(m*v);    // de-Broglie wavelength of a proton, m
printf("\nThe de-Broglie wavelength of a proton = %4.2e m", Lambda);

// Result
// The de-Broglie wavelength of a proton = 2.64e-014 m
```

Scilab code Exa 1.4 Energy of neutron in electron volt

```
// Scilab code Ex1.4: Pg:19 (2008)
clc;clear;
m = 1.674e-027;    // Mass of neutron, kg
h = 6.60e-034;    // Planck's constant, joule-sec
Lambda = 1e-010;    // de-Broglie wavelength of neutron,
E = h^2/(2*m*Lambda^2);    // Energy of neutron, joule
printf("\nThe energy of neutron in electron volt = %4.2e eV", E/1.6e-019);

// Result
// The energy of neutron in electron volt = 8.13e-002 eV
```

Scilab code Exa 1.5 Energy of an electron wave in electron volt

```
// Scilab code Ex1.5: Pg:20 (2008)
clc;clear;
m = 9.1e-031;    // Mass of the electron, kg-m
h = 6.62e-034;    // Planck's constant, joule-sec
```

```
Lambda = 3e-002;     // de-Broglie wavelength of the electron , m
E = h^2/(2*m*Lambda^2);     // Energy of the electron wave, joule
printf("\nThe energy of the electron wave = %4.2e eV", E/1.6e-019);

// Result
// The energy of the electron wave = 1.67e-015 eV
```

Scilab code Exa 1.6 Voltage applied to an electron microscope to produce electrons

```
// Scilab code Ex1.6:  Pg:20 (2008)
clc; clear;
e = 1.6e-019;     // Energy equivalent of 1 eV, J
m = 9.1e-031;     // Mass of an electron , kg-m
h = 6.6e-034;     // Planck's constant , joule-sec
Lambda = 0.4e-010;     // de-Broglie wavelength of an electron , m
// Since E = e*V and Lambda = h/sqrt(2*m*e*V), solving for V we have
V = h^2/(2*Lambda^2*m*e);     // Voltage that must be applied to an electron microscope, volt
printf("\nThe voltage that must be applied to the electron microscope = %3d V", V);

// Result
// The voltage that must be applied to the electron microscope = 934 V
// The answer is given wrongly in the textbook
```

Scilab code Exa 1.7 Wavelength of quantum of radiant energy

```
// Scilab code Ex1.7: Pg:20 (2008)
clc;clear;
m = 9.1e-031;     // Mass of an electron, kgm
h = 6.6e-034;     // Planck's constant, joule-sec
c = 3e+08;    // Velocity of light, m/s
// Energy of one quantum of radiation is given by E = h*nu and
// furhter, E = m*c^2 where nu = c/Lambda, the frequency of radiation
// On compairing the energies and solving for Lambda
Lambda = h/(m*c);     // de Broglie wavelength of an electron, m
printf("\nThe wavelength of quantum of radiant energy = %6.4f angstrom", Lambda/1e-010);

// Result
// The wavelength of quantum of radiant energy = 0.0242 angstrom
```

Scilab code Exa 1.8 de Broglie wavelength of neutron

```
// Scilab code Ex1.8: Pg:20 (2008)
clc;clear;
m = 1.675e-027;     // Mass of a neutron, kg
h = 6.625e-034;     // Planck's constant, joule-sec
E = 1.6e-005;     // Kinetic energy of the neutron, joule
// Since (1/2)*m*v^2 = 1.6e-005, solving for v
v = (2*E/m)^(1/2);
Lambda = h/(m*v);     // de Broglie wavelength of a neutron, m
printf("\nThe de-Broglie wavelength of neutron = %4.2e m", Lambda);

// Result
```

```
// The de-Broglie wavelength of neutron = 2.86e-018 m
```

Scilab code Exa 1.9 de Broglie wavelength of proton whose kinetic energy is equal to the rest energy of an electron

```
// Scilab code Ex1.9:  Pg:21 (2008)
clc;clear;
h = 6.62e-034;     // Planck's constant, joule-sec
c = 3e+008;     // Velocity of light, m/s
m_0 = 9.1e-031;     // Rest mass of an electron, kg
m = 1836*m_0;     // Mass of a proton, kg
E = m_0*c^2;     // Energy of an electron, joule
// Since (1/2)*m*v^2 = 81.9e-015, solving for v
v = (2*E/m)^(1/2);     // Velocity of the electron, m/s
Lambda = h/(m*v);     // The de-Broglie wavelength of a proton, m
printf("\nThe de-Broglie wavelength of proton whose kinetic energy is equal to the rest energy of an electron = %1.0e angstrom", Lambda/1e-010);

// Result
// The de-Broglie wavelength of proton whose kinetic energy is equal to the rest energy of an electron = 4e-004 angstrom
```

Scilab code Exa 1.10 Maximum speed of electrons striking anticathode in an X ray tube

```
// Scilab code Ex1.10: Pg:36 (2008)
clc;clear;
m = 9.13e-031;     // Mass of an electron, kg
```

```
e = 1.6e-019;     // Charge of electron, coulomb
V = 20000;     // Potential difference applied between cathode and anode, volt
// Since (1/2)*m*v^2 = e*V, solving for v
v = sqrt(2*e*V/m);     // Maximum speed of electrons striking the anti cathode, m/s
printf("\nThe maximum speed of electrons striking anticathode in an X-ray tube = %4.2e m/s", v);

// Result
// The maximum speed of electrons striking anticathode in an X-ray tube = 8.37e+007 m/s
```

Scilab code Exa 1.11 Shortest wavelength of X rays in an X ray tube

```
// Scilab code Ex1.11:  Pg:36 (2008)
clc;clear;
h = 6.62e-034;     // Planck's constant, joule-sec
c = 3e+08;     // Velocity of light, m/s
m = 9.13e-031;     // Mass of an electron, kg
e = 1.6e-019;     // Charge of electron, coulomb
V = 18000;     // Potential difference applied between cathode and anode, volts
E = e*V;     // Energy of the electron, joule
// Since energy of X-rays is equal to energy of the electron thus
// h*c/Lambda = e*V, solving for Lambda
Lambda = h*c/E;     // Wavelength of X-rays, angstorm
printf("\nThe shortest wavelength of X-rays in an X-ray tube = %4.2f angstorm", Lambda/1e-010);

// Result
// The shortest wavelength of X-rays in an X-ray tube = 0.69 angstorm
```

Scilab code Exa 1.12 Energy and velocity of an electron beam

```
// Scilab code Ex1.12:  Pg:37 (2008)
clc;clear;
Lambda = 1e-010;    // Wavelength of X-rays, cm
c = 3e+08;    // Velocity of light, m/s
m = 9.13e-031;    // Mass of an electron, kg
h = 6.62e-034;    // Planck's constant, joule-sec
e = 1.6e-019;    // Charge of electron, coulomb
f = c/Lambda;    // Frequency of X-rays, cycles/sec
E = h*f;    // Energy of X-ray photon, joule
// Since energy of X-ray photon is converted into energy of electrons thus
// h*f = (1/2)*m*v^2, solving for v
v = sqrt(2*h*f/m);    // Velocity of the electron, m/s
printf("\nThe energy of an electron beam = %5.0f eV", E/e);
printf("\nThe velocity of an X-ray beam = %5.3e m/s", v);

// Result
// The energy of an electron beam = 12413 eV
// The velocity of an X-ray beam = 6.596e+007 m/s
```

Scilab code Exa 1.13 Minimum voltage applied to an X ray tube to produce X rays

```
// Scilab code Ex1.13:  Pg:37 (2008)
clc;clear;
Lambda = 1e-010;    // Wavelength of X-rays, m
c = 3e+08;    // Velocity of light, m/s
```

```
h = 6.625e-034;     // Planck's constant, joule-sec
e = 1.6e-019;     // Charge of electron, coulomb
E = h*c/Lambda;     // Energy of X-rays, cycles/sec
// Since h*c/Lambda = e*V, solving for V
V = E/e;     // voltage applied to an X-ray tube,
    volts
printf("\nThe minimum voltage applied to an X-ray
    tube to produce X-rays = %5.2e volt", V);

// Result
// The minimum voltage applied to an X-ray tube to
    produce X-rays = 1.24e+004 volt
```

Scilab code Exa 1.14 Wavelength of X rays in Bragg reflection

```
// Scilab code Ex1.14: Pg:43 (2008)
clc;clear;
d = 2.82e-008;     // Interplanar spacing in sodium
    chloride crystal, cm
n = 1;     // Order of reflection
theta = 10;     // Glancing angle, degree
// Since 2*d*sin theta = n*Lambda, solving for
    Lambda
Lambda = 2*d*sind(theta);     // Wavelength of X-rays
     in Bragg's reflection, cm
printf("\nThe wavelength of X-rays in Bragg
    reflection = %4.2f angstrom", Lambda/1e-008);

// Result
// The wavelength of X-rays in Bragg reflection =
    0.98 angstrom
```

Scilab code Exa 1.15 Glancing angle for the first order Bragg spectrum in Sylvine crystal

```
// Scilab code Ex1.15: Pg:44 (2008)
clc;clear;
function [deg, minute] = deg2min(theta)
    deg = floor(theta);
    minute = (theta-deg)*60;
endfunction
d = 3.14e-010;    // Interplanar spacing in sylvine
   crystal, cm
n = 1;    // Order of reflectio
h = 6.62e-034;    // Planck's constant, joule-sec
c = 3e+08;    // Velocity of light, m/s
E = 0.01*1e+06*1.6e-019;    // Energy of X-ray beam,
    joule
Lambda = h*c/E;    // Wavelength of X-rays, m
// Since 2*d*sin theta = n*Lambda, solving for theta
theta = asind(n*Lambda)/(2*d)    // Glancing angle,
   degree
[deg, minute] = deg2min(theta);
printf("\nThe glancing angle for the first order
   Bragg spectrum in Sylvine crystal = %2d degree
   %2d minute", deg, minute);

// Result
// The glancing angle for the first order Bragg
   spectrum in Sylvine crystal = 11 degree 19 minute
// The answer is given wrongly in the textbook
```

Chapter 2

Electron Optics

Scilab code Exa 2.1 Potential difference between two regions of an electric field

```
// Scilab code Ex2.1: Pg:55 (2008)
clc;clear;
V1 = 250;     // Accelerating potential of electron
   in first region, volts
theta1 = 50;     // Angle of incidence, degrees
theta2 = 30;     // Angle of refraction, degrees
// According to Bethe's law  sind(theta1)/sind(
   theta2) = []V2/V1]^1/2
// On solving for V2
V2 = V1*(sind(theta1)/sind(theta2))^2;     //
   Potential in second region, volts
deltaV = (V2-V1);     // Potential difference between
    two regions, volts
printf("\nPotential difference between two regions
   of an electric field = %5.1f V", deltaV);

// Result
// Potential difference between two regions of an
   electric field = 336.8 V
```

Scilab code Exa 2.2 Linear separation between the lines on a photographic plates

```
// Scilab code Ex2.2: Pg:79(2008)
clc;clear;
amu = 1.67e-027;     // Mass of a nucleon, kg
E = 8e+004;     // Electric field in a Bainbridge mass spectrograph, V/m
B = 0.55;     // Magnetic induction, Wb per square meter
M1 = 20;     // Atomic mass of first isotope of neon, amu
M2 = 22;     // Atomic mass of second isotope of neon, amu
q = 1.602e-019;     // Charge of the ion, coulomb
delta_x = 2*E*(M2-M1)*amu/(q*B^2);     // Separation between the lines, mm
printf("\nLinear separation between the lines on a photographic plates = %4.2f m", delta_x);

// Result
// Linear separation between the lines on a photographic plates= 0.01 m
```

Chapter 3

Geometrical Optics

Scilab code Exa 3.1 Positions of the cardinal points

```
// Scilab code Ex3.1  Pg:89 (2008)
clc;clear;
f1 = 30;    // Focal length of first lens, cm
f2 = 10;    // Focal length of second lens, cm
d = 25;    // Distance of separation between two lenses, cm
F = f1*f2/(f1 + f2 - d);    // Focal length of the combination of lenses, cm
// Positions of Principal Points
alpha = F*d/f2;    // Distance of the first principal point from the first lens, cm
bita = -F*d/f1;    // Distance of the second principal point from the second lens, cm
// Positions of Focal Points
L1F1 = -F*(1-d/f2);    // Distance of the first focal point from the first lens, cm
L2F2 = F*(1-d/f1);    // Distance of the second focal point from the second lens, cm
printf("\nThe positions of Principal points = %2.0f cm and %4.2f cm", alpha, bita);
printf("\nThe positions of Focal points = %2.0f cm
```

```
    and %3.1f cm", L1F1, L2F2);

// Result
// The positions of Principal points = 50 cm and
    -16.67 cm
// The positions of Focal points = 30 cm and 3.3 cm
```

Scilab code Exa 3.2 Coaxial converging and diverging lenses held at a distance

```
// Scilab code Ex3.2: Pg:90 (2008)
clc;clear;
f1 = 10;    // Focal length of converging lens, cm
f2 = -10;    // Focal length of diverging lens, cm
d = 5;    // Distance of separation between two
    lenses, cm
F = f1*f2/(f1 + f2 - d);    // Focal length of the
    combination of lenses, cm
P = 100/F;    // Power of the combination of lenses,
     diopter
// Positions of Principal Points
alpha = F*d/f2;    // Distance of the first
    principal point from the first lens, cm
bita = -F*d/f1;    // Distance of the second
    principal point from the second lens, cm
printf("\nThe focal length of the combination of
    lenses = %2.0f cm", F);
printf("\nThe power of the combination of lenses =
    %1.0f diopter", P);
printf("\nThe positions of Principal points = %2.0f
    cm and %2.0f cm", alpha, bita);

// Result
// The focal length of the combination of lenses =
    20 cm
```

```
// The power of the combination of lenses = 5
   diopter
// The positions of Principal points = −10 cm and
   −10 cm
```

Scilab code Exa 3.3 Combination of a convex and a concave lens placed at a distance

```
// Scilab code Ex3.3 : Pg:91 (2008)
clc;clear;
f1 = 30;    // Focal length of convex lens, cm
f2 = -50;    // Focal length of concave lens, cm
d = 20;    // Distance of separation between two
   lenses, cm
F = f1*f2/(f1 + f2 - d);    // Focal length of the
   combination of lenses, cm
// Positions of Principal Points
alpha = F*d/f2;    // Distance of the first
   principal point from the first lens, cm
bita = -F*d/f1;    // Distance of the second
   principal point from the second lens, cm
// Positions of Focal Points
L1F1 = -F*(1-d/f2);    // Distance of the first
   focal point from the first lens, cm
L2F2 = F*(1-d/f1);    // Distance of the second
   focal point from the second lens, cm
// Positions of Final image
u = -25;    // Object distance from principal point,
    cm
// As from thin lens formula, 1/v − 1/u = 1/F,
   solving for v
v = (u*F)/(u+F);    // Image distance from principal
    point, cm
m = v/u;    // Linear magnification
printf("\nThe positions of Principal points = %2.0f
```

```
    cm and %4.2f cm", alpha, bita);
printf("\nThe positions of Focal points = %4.1f cm
    and %4.1f cm", L1F1, L2F2);
printf("\nThe image distance from principal point =
    %2.0f cm", v);
printf("\nThe linear magnification = %1.0f cm", m);



// Result
// The positions of Principal points = -15 cm and
    -25.00 cm
// The positions of Focal points = -52.5 cm and 12.5
    cm
// The image distance from principal point = -75 cm
// The linear magnification = 3 cm
```

Scilab code Exa 3.4 Lens combination in Huygen eye piece

```
// Scilab code Ex3.4 : Pg:97 (2008)
clc;clear;
f = 4;     // Focal length of eye lens of Huygen eye-
    piece, cm
f1 = 3*f;     // Focal length of first lens, cm
f2 = f;     // Focal length of second lens, cm
d = 2*f;     // Distance of separation between two
    lenses, cm
F = f1*f2/(f1 + f2 - d);     // Focal length of the
    combination of lenses, cm
// Positions of Principal Points
alpha = F*d/f2;     // Distance of the first
    principal point from the first lens, cm
bita = -F*d/f1;     // Distance of the second
    principal point from the second lens, cm
// Positions of Focal Points
L1F1 = -F*(1-d/f2);     // Distance of the first
```

```
    focal point from the first lens, cm
L2F2 = F*(1-d/f1);    // Distance of the second focal point from the second lens, cm
// Positions of Final image
u = -18;    // Object distance from principal point, cm
// As from thin lens formula, 1/v - 1/u = 1/F, solving for v
v = (u*F)/(u+F);    // Image distance from principal point, cm
L2I = v + bita;    // The position of image to the right of eye lens, cm
printf("\nThe positions of Principal points = %2.0f cm and %1.0f cm", alpha, bita);
printf("\nThe positions of Focal points = %1.0f cm and %1.0f cm", L1F1, L2F2);
printf("\nThe The position of image to the right of eye lens = %1.0f cm", L2I);


// Result
// The positions of Principal points = 12 cm and -4 cm
// The positions of Focal points = 6 cm and 2 cm
// The The position of image to the right of eye lens = 5 cm
```

Scilab code Exa 3.5 Focal lengths of the plano convex lenses and the equivalent focal length of the Huygen eye piece

```
// Scilab code Ex3.5 : Pg:98 (2008)
clc;clear;
d = 10;    // Distance of separation of two lenses, cm
// As 2*f1 = d, solving for f1
```

```
f1 = d/2;     // Focal length of the first plano-convex lens, cm
f2 = 3*f1;     // Focal length of the second plano-convex lens, cm
F = f1*f2/(f1 + f2 - d);     // Focal length of the eye-piece, cm
printf("\nThe focal lengths of the plano-convex lenses are %1.0f cm and %2.0f cm", f1, f2);
printf("\nThe focal length of the eye-piece = %3.1f cm", F);

// Result
// The focal lengths of the plano-convex lenses are 5 cm and 15 cm
// The focal length of the eye-piece = 7.5 cm
```

Scilab code Exa 3.6 Focal lengths of two lenses and their separation distance in Huygen and Ramsden eye pieces

```
// Scilab code Ex3.6 : Pg:101 (2008)
clc;clear;
F = 12;     // Focal length of the eye-piece, cm
// For Huygen's eye-piece
// As F = f1*f2/(f1 + f2 - d) and f1 = 3*f; f2 = f; d = 2*f, solving for f
f = poly(0, 'f');
f = roots(3*f*f-F*(3*f+f-2*f));     // Focal length of the eye-lens, cm
d = 2*f(1);     // Distance of separation of two lenses, cm
f1 = 3*f(1);     // Focal length of the first plano-convex lens, cm
f2 = f(1);     // Focal length of the second plano-convex lens, cm
printf("\nFor Huygen eye-piece:");
```

```
printf("\nThe focal lengths of the plano−convex lenses are %1.0f cm and %2.0f cm", f1, f2);
printf("\nThe distance between the lenses = %2.0f cm", d);
// For Ramsden eye−piece
// As F = f1∗f2/(f1 + f2 − d) and f1 = f; f2 = f; d = 2/3∗f, solving for f
f = poly(0, 'f');
f = roots(f*f-12*(f+f-2/3*f));     // Focal length of the eye−lens, cm
d = 2/3*f(1);     // Distance of separation of two lenses, cm
f1 = f(1);     // Focal length of the first plano−convex lens, cm
f2 = f(1);     // Focal length of the second plano−convex lens, cm
printf("\n\nFor Ramsden eye−piece:");
printf("\nThe focal lengths of the plano−convex lenses are %1.0f cm and %2.0f cm", f1, f2);
printf("\nThe distance between the lenses = %5.2f cm", d);

// Result
// For Huygen eye−piece:
// The focal lengths of the plano−convex lenses are 24 cm and  8 cm
// The distance between the lenses = 16 cm

// For Ramsden eye−piece:
// The focal lengths of the plano−convex lenses are 16 cm and 16 cm
// The distance between the lenses = 10.67 cm
// The distance between the lenses for Ramsden eye−piece is wrong in the textbook
```

Scilab code Exa 3.7 Composition and cardinal points of a Ramsden eye piece

```
// Scilab code Ex3.7  : Pg:102 (2008)
clc;clear;
F = 9.0;    // Focal length of the eye-piece, cm
// As F = f1*f2/(f1 + f2 - d) and f1 = f; f2 = f; d = 2/3*f, solving for f
f = poly(0, 'f');
f = roots(f*f-F*(f+f-2/3*f));    // Focal length of the eye-lens, cm
d = 2/3*f(1);    // Distance of separation of two lenses, cm
f1 = f(1);    // Focal length of the first plano-convex lens, cm
f2 = f(1);    // Focal length of the second plano-convex lens, cm
alpha = F*d/f2;    // Distance of first principal point from the field lens L1, cm
bita = -F*d/f1;    // Distance of second principal point from the field lens L2, cm
L1F1 = -F*(1-d/f2);    // Distance of first focal point from the lens L1, cm
L2F2 = F*(1-d/f1);    // Distance of second focal point from the lens L2, cm
printf("\nThe focal lengths of the plano-convex lenses are %1.0f cm and %2.0f cm", f1, f2);
printf("\nThe distance between the lenses = %1.0f cm", d);
printf("\nThe distance of first principal point from the field lens L1 = %1.0f cm", alpha);
printf("\nThe distance of second principal point from the field lens L2 = %1.0f cm", bita);
printf("\nThe distance of first focal point from the field lens L1 = %1.0f cm", L1F1);
printf("\nThe distance of second focal point from the field lens L2 = %1.0f cm", L2F2);
```

```
// Result
// The focal lengths of the plano-convex lenses are
   12 cm and 12 cm
// The distance between the lenses = 8 cm
// The distance of first principal point from the
   field lens L1 = 6 cm
// The distance of second principal point from the
   field lens L2 = -6 cm
// The distance of first focal point from the field
   lens L1 = -3 cm
// The distance of second focal point from the field
    lens L2 = 3 cm
```

Scilab code Exa 3.8 Longitudinal chromatic abberation for an object at infinity

```
// Scilab code Ex3.8 : Pg:108 (2008)
clc;clear;
mu_v = 1.5230;    // Refractive index of violet
   color
mu_r = 1.5145;    // Refractive index of red color
R1 = 40;    // Radius of curvature of first
   curvature of lens, cm
R2 = -10;    // Radius of curvature of second
   curvature of lens, cm
// As 1/f_r = (mu_r - 1)*(1/R1 - 1/R2), solving for
   f_r
f_r = 1/((mu_r-1)*(1/R1 - 1/R2));    // Focal length
    for red color, cm
f_v = 1/((mu_v-1)*(1/R1 - 1/R2));    // Focal length
    for violet color, cm
CA = f_r - f_v;    // The longitudinal chromatic
   abberation, cm
printf("\nThe longitudinal chromatic abberation for
   the object at infinity = %5.3f cm", CA);
```

```
// Result
// The longitudinal chromatic abberation for the object at infinity = 0.253 cm
```

Scilab code Exa 3.9 Longitudinal chromatic abberation for a lens of crown glass

```
// Scilab code Ex3.9 : Pg:109 (2008)
clc;clear;
mu_F = 1.5249;    // Refractive index of violet color
mu_C = 1.5164;    // Refractive index of red color
mu_D = (mu_F + mu_C)/2;    // Mean refractive index
omega = (mu_F - mu_C)/(mu_D - 1);    // Dispersive power of the lens
f = 40;    // Focal length of the crown glass lens, cm
CA = omega*f;    // The longitudinal chromatic abberation, cm
printf("\nThe longitudinal chromatic abberation = %6.4f cm", CA);

// Result
// The longitudinal chromatic abberation = 0.6530 cm
// The answer is given wrong in the textbook
```

Scilab code Exa 3.10 Focal length of the crown glass convex lens forming an achromatic doublet with a flint glass concave lens

```
// Scilab code Ex3.10 : Pg:113 (2008)
clc;clear;
```

```
omega1 = 0.02;    // Dispersive power of the convex lens
omega2 = 0.04;    // Dispersive power of the concave lens
f2 = -80;    // Focakl length of the concave lens, cm
// As omega1/omega2 = -f1/f2, solving for f1
f1 = -omega1/omega2*f2;    // Focal length of the crown glass convex lens, cm
printf("\nThe focal length of the crown glass convex lens = %2.0f cm", f1);

// Result
// The focal length of the crown glass convex lens = 40 cm
```

Scilab code Exa 3.11 Dispersive power of the flint glass

```
// Scilab code Ex3.11 : Pg:113 (2008)
clc;clear;
mu_V = 1.55;    // Refractive index of violet color
mu_R = 1.53;    // Refractive index of red color
mu_Y = (mu_V + mu_R)/2;    // Refractive index of yellow color
omega1 = (mu_V - mu_R)/(mu_Y - 1);    // Dispersive power of the crown glass convex lens
F = 150;    // Focal length of the combination of lenses, cm
R = 54;    // Radius of curvature of the convex lens, cm
f1 = R/(2*(mu_Y-1));    // Focal length of the convex lens from thin lens maker formula, cm
f2 = F*f1/(f1 - F);    // Focal length of the second lens, cm
// As omega1/omega2 = -f1/f2, solving for omega2
```

```
omega2 = -f2/f1*omega1;     // Dispersive power of flint glass
printf("\nThe dispersive power of flint glass = %5.3f", omega2);

// Result
// The dispersive power of flint glass = 0.056
```

Scilab code Exa 3.12 Radius of curvature of the second surface each for crown glass and flint glass lens

```
// Scilab code Ex3.12 : Pg:114 (2008)
clc;clear;
omega1 = 0.017;     // Dispersive power of the crown glass lens
omega2 = 0.034;     // Dispersive power of flint glass lens
F = 40;     // Focal length of the combination of lenses, cm
f1 = (omega2 - omega1)/omega2*F;     // Focal length of crown glass lens, cm
f2 = (omega1 - omega2)/omega1*F;     // Focal length of flint glass lens, cm
mu = 1.5;     // Refractive index of crown glass
R2 = -25;     // Radius of curvature of the first surface of convex lens, cm
// Now from lens maker's formula
R1 = (mu - 1)/(1/f1+(mu-1)/R2);     // Radius of curvature of second surface of convex lens, cm
printf("\nThe radius of curvature of the second surface of convex lens = %5.2f cm", R1);
mu = 1.7;     // Refractive index of flint glass
R1 = -25;     // Radius of curvature of the first surface of concave lens, cm
R2 = (mu - 1)/(1/f2-(mu-1)/R1);     // Radius of
```

```
    curvature of second surface of concave lens, cm
printf("\nThe radius of curvature of the second
    surface of concave lens = %6.2f cm", R2);

// Result
// The radius of curvature of the second surface of
    convex lens = 16.67 cm
// The radius of curvature of the second surface of
    concave lens = 233.33 cm
```

Scilab code Exa 3.13 Radius of curvature of convex lens from given data

```
// Scilab code Ex3.13 : Pg:115 (2008)
clc;clear;
P = 5;    // Power of combination of a convex lens
    and a plano-convex lens, dioptre
mu1 = 1.50;    // Refractive index of crown glass
mu2 = 1.60;    // Refractive index of flint glass
omega1 = 0.01;    // Dispersive power of the crown
    glass convex lens
omega2 = 0.02;    // Dispersive power of flint glass
     plano-convex lens
F = 100/P;    // Focal length of the combination of
    lenses, cm
f_ratio = -omega2/omega1;    // Ratio of f2 to f1
// From thin lens formula, 1/F = 1/f1 + 1/f2 and as
    f2 = f_ratio*f1, solving for f1
f1 = -F/f_ratio;    // Focal length of flint glass
    lens, cm
f2 = f_ratio*f1;    // Focal length of crown glass
    lens, cm
mu = 1.60;    // Refractive index of flint glass
R2 = %inf;    // Radius of curvature of the first
    surface of convex lens, cm
// Now from lens maker's formula
```

```
R1 = (mu - 1)/(1/f2+(mu-1)/R2);    // Radius of curvature of second surface of convex lens, cm
mu = 1.5;    // Refractive index of crown glass
R2 = R1;    // Radius of curvature of the first surface of convex lens, cm
R1_prime = (mu - 1)/(1/f1+(mu-1)/R2);    // Radius of curvature of second surface of concave lens, cm
printf("\nThe radii of curvature of the convex lens are = %-3.1f cm and %2.0f cm", R1_prime, R1);

// Result
// The radii of curvature of the convex lens are = 8.6 cm and -12 cm
```

Scilab code Exa 3.15 Distance between two achromatic lenses

```
// Scilab code Ex3.15 : Pg:117 (2008)
clc;clear;
omega1 = 0.01;    // Dispersive power of the crown glass convex lens
omega2 = 0.02;    // Dispersive power of flint glass plano-convex lens
f1 = 20;    // Focal length of crown glass lens, cm
f2 = 30;    // Focal length of crown flint lens, cm
d = (omega1*f2+omega2*f1)/(omega1 + omega2);    // The distance between two achromatic lenses of different material, cm
// For same material
printf("\nThe distance between two achromatic lenses of different material = %5.2f cm", d);
omega1 = 1, omega2 = 1;
d = (omega1*f2+omega2*f1)/(omega1 + omega2);    // The distance between two achromatic lenses of same material, cm
```

```
printf("\nThe distance between two achromatic lenses of same material = %2.0f cm", d);

// Result
// The distance between two achromatic lenses of different material = 23.33 cm
// The distance between two achromatic lenses of same material = 25 cm
```

Scilab code Exa 3.16 Spherical aberration for a spherical surface

```
// Scilab code Ex3.16 : Pg:121 (2008)
clc;clear;
R = 20;    // Radius of curvature of the spherical surface, cm
mu = 1.5;    // Refractive index of the material
h = 5;    // First height of the incident ray from the principal axis, cm
delta_f_h = h^2/(2*mu*(mu - 1)*R);    // Spherical aberration of the spherical surface, cm
printf("\nFor h = %d, the Spherical aberration of the spherical surface = %4.2f cm", h, delta_f_h);
h = 7;  // Second height of the incident ray from the principal axis, cm
delta_f_h = h^2/(2*mu*(mu - 1)*R);    // Spherical aberration of the spherical surface, cm
printf("\nFor h = %d, the Spherical aberration of the spherical surface = %4.2f cm", h, delta_f_h);

// Result
// For h = 5, the Spherical aberration of the spherical surface = 0.83 cm
// For h = 7, the Spherical aberration of the spherical surface = 1.63 cm
```

Scilab code Exa 3.17 Focal length of component lenses of a convergent doublet

```
// Scilab code Ex3.17 : Pg:125(2008)
clc;clear;
F = 10;     // Equivalent focal length of the
    combination of lenses, cm
d = 2;     // Distance between the lenses of doublet,
     cm
// The condition of minimum spherical aberration
    gives
// f1 = f2 = d or f2 = f1 - d
f1 = 2*F;     // Focal length of the first lens, cm
f2 = f1 - d;     // Focal length of the second lens,
    cm
printf("\nThe focal length of component lenses of a
    convergent doublet, f1 = %2d cm and f2 = %2d cm",
     f1, f2);

// Result
// The focal length of component lenses of a
    convergent doublet, f1 = 20 cm and f2 = 18 cm
```

Scilab code Exa 3.18 Design of a no chromatic aberration and minimum spherical abberation doublet lens

```
// Scilab code Ex3.18 : Pg:125(2008)
clc;clear;
F = 5.0;     // Equivalent focal length of the
    combination of lenses, cm
// As F = 3*d/4, solving for d
```

```
d = 4/3*F;     // // Distance between the lenses of doublet, cm
// The condition of minimum spherical aberration gives
// 2*d = f1 + f2 and f1 - f2 = d, solving for f1 and f2
f1 = 3*d/2;    // Focal length of the first lens, cm
f2 = d/2;    // Focal length of the second lens, cm
printf("\nTo have no chromatic aberration and minimum spherical abberation, the doublet lens should be designed with the following parameters:\n");
printf(" d = %4.2f cm; f1 = %2d cm and f2 = %4.2f cm", d, f1, f2);

// Result
// To have no chromatic aberration and minimum spherical abberation, the doublet lens should be designed with the following parameters:
//  d = 6.67 cm; f1 = 10 cm and f2 = 3.33 cm
```

Chapter 4

Wave Theory of Light

Scilab code Exa 4.1 Ratio between the amplitude and intensities of the two interfering waves

```
// Scilab code Ex4.1 : Pg:139 (2008)
clc;clear;
I_max = 36;    // Maxiumum intensity of interfering waves
I_min = 1;    // Minimum intensity of interfering waves
// As (a + b)/(a - b) = sqrt(I_max/I_min), solving for a/b
a1 = sqrt(I_max)+1;    // Amplitude of first wave, unit
a2 = sqrt(I_max)-1;    // Amplitude of second wave, unit
I1 = a1^2;    // Intensity of the first wave, unit
I2 = a2^2;    // Intensity of the second wave, unit
printf("\nThe ratio between the amplitudes of the two interfering waves, a1:a2 = %d:%d", a1, a2);
printf("\nThe ratio between the intensities of the two interfering waves, I1:I2 = %d:%d", I1, I2);

// Result
```

```
// The ratio between the amplitudes of the two
    interfering waves, a1:a2 = 7:5
// The ratio between the intensities of the two
    interfering waves, I1:I2 = 49:25
```

Scilab code Exa 4.2 Ratio of maximum intensity to minimum intensity of the two interfering waves

```
// Scilab code Ex4.2 : Pg:139 (2008)
clc;clear;
I1 = 100;    // Maxiumum intensity of interfering
    waves
I2 = 1;    // Minimum intensity of interfering waves
a1_ratio_a2 = sqrt(I1/I2);    // Ratio of two
    amplitudes
a2 = 1;    // Assume the amplitude of second wave to
      be unity
a1 = a2*a1_ratio_a2;    // The amplitude of second
    wave
I_max = (a1+a2)^2;    // Maximum intensity of
    interfering waves
I_min = (a1-a2)^2;    // Minimum intensity of
    interfering waves
printf("\nThe ratio of maximum intensity to minimum
    intensity of the two interfering waves, I_max:
    I_min = %d:%d", I_max, I_min);

// Result
// The ratio of maximum intensity to minimum
    intensity of the two interfering waves, I_max:
    I_min = 121:81
```

Scilab code Exa 4.4 Lowest phase difference between the waves at interfering point

```
// Scilab code Ex4.4 : Pg:140 (2008)
clc;clear;
I1 = 1.44;     // Intensity of first wave
I2 = 4.00;     // Intensity of second wave
I = 0.90;     // Intensity of resultant wave
// As I_delta = I1 + I2 + 2*sqrt(I1*I2)*cos(delta),
    solving for delta
delta = acosd((I-I1-I2)/(2*sqrt(I1*I2)));
printf("\nThe lowest phase difference between the
    waves at interfering point = %3d degree", delta);

// Result
// The lowest phase difference between the waves at
    interfering point = 161 degree
```

Scilab code Exa 4.6 Value of fringe width

```
// Scilab code Ex4.6: : Pg:146 (2008)
clc;clear;
D = 60;     // Distance between the source and the
    screen, cm
Lambda = 5.9e-05;     // Wavelength of light, cm
d = 0.3/2;     // Separation between the slits, cm
omega = D*Lambda/(2*d);     // Fringe width, cm
printf("\nThe value of fringe width = %6.4f cm",
    omega);

// Result
// The value of fringe width = 0.0118 cm
```

Scilab code Exa 4.7 Wavelength of light

```
// Scilab code Ex4.7 : Pg:146 (2008)
clc;clear;
D = 80;    // Distance between the source and the
    screen , cm
d = 0.018/2;    // Separation between two coherent
    sources , cm
n = 4;    // Number of the fringe
x_n = 1.08;    // Distance of nth bright fringe from
     the center of central fringe , cm
// As x_n = n*Lambda*D/(2*d), solving for Lambda
Lambda = x_n*2*d/(n*D);    // wavelength of light ,
    Angstorm
printf("\nThe wavelength of light used = %4.0f
    angstrom", Lambda/1e-008);

// Result
// The wavelength of light used = 6075 angstrom
```

Scilab code Exa 4.8 Double slit separation

```
// Scilab code Ex4.8 : Pg:146 (2008)
clc;clear;
D = 200;    // Distance between the source and the
    screen , cm
Lambda = 5100e-08;    // Wavelength of light , cm
x = 2;    // Separation of fringes , cm
n = 10;    // number of fringes
omega = x/n;    // Fringe width , cm
d = D*Lambda/(2*omega);    // Double slit separation
    , mm
printf("\nThe double slit separation = %4.2f mm", 2*
    d*10);

```

```
// Result
// The double slit separation = 0.51 mm
```

Scilab code Exa 4.9 Wavelength of light used in double slit experiment

```
// Scilab code Ex4.9: Pg:147 (2008)
clc;clear;
D = 1000;     // Distance between the source and the
    screen , mm
omega = 1;     // For simplicity assume fringe width
    to be unity , mm
x9 = 9*omega;     // Position of 9th bright fringe ,
   mm
x2_prime = 3/2*omega;     // Position of 9th bright
    fringe , mm
d = 0.5/2;     // Separation between the slits , mm
l = 8.835;     // Distance between 9th bright fringe
    and second dark fringe
// As x9 − x2_prime = 9∗omega−3/2∗omega = l , solving
     for omega
omega = l/(x9 - x2_prime);     // Fringe width , mm
lambda = omega*2*d/D;     // Wavelength of light used
    , mm
printf("\nThe wavelength of light used = %4d
    angstrom", lambda/1e-007);

// Result
// The wavelength of light used = 5890 angstrom
```

Scilab code Exa 4.10 Wavelength of light in two slit experiment

```
// Scilab code Ex4.10: Pg:147 (2008)
clc;clear;
```

```
delta_D = 5e-002;    // Distance through which the
    screen is moved, m
delta_omega = 3e-005;    // Change in fringe width
    as a result of motion of screen, m
d = 1e-003/2;    // Half of the separation distance
    between the slits, m
// As delta_omega = lambda*delta_D/(2*d), solving
    for lambda
lambda = delta_omega*(2*d)/delta_D;    // Wavelength
     of light used, m
printf("\nThe wavelength of light used = %4d
    angstrom", lambda/1e-010);

// Result
// The wavelength of light used = 6000 angstrom
```

Scilab code Exa 4.11 Position of twentieth order fringes relative to zero order fringe in two slit interference pattern

```
// Scilab code Ex4.11: Pg:148 (2008)
clc;clear;
x0 = 12.34;    // Position of zero order fringe, mm
Lambda = 6000;    // Wavelength of light, angstrom
Lambda_prime = 5000;    // New wavelength of light,
    angstrom
omega = 0.239;    // Fringe width, mm
omega_prime = Lambda_prime/Lambda*omega;    // New
    fringe width, mm
d_20 = 20*omega_prime;    // Separation of 20th
    fringe, mm
x_20 = [d_20, -d_20];    // Position of 20th order
    fringe, mm
x = x0 + x_20;    // Positions of 20th order fringe
     relative to zero order fringe, mm
printf("\nThe positions of 20th order fringe
```

```
relative to zero order fringe are %5.2f mm or %4.2f mm", x(1), x(2));

// Result
// The positions of 20th order fringe relative to zero order fringe are 16.32 mm or 8.36 mm
```

Scilab code Exa 4.12 Brightt fringes in Young double slit experiment

```
// Scilab code Ex4.12: Pg:149 (2008)
clc;clear;
Lambda = 6500e-007;     // Wavelength of light, mm
Lambda_prime = 5200e-007;     // New wavelength of light, mm
n = 3;     // Order of bright fringe
D = 1200;     // Distance between the source and the slits, mm
d = 2/2;     // Separation between teh slits, mm
x3 = n*Lambda*D/(2*d);     // The distance of the third bright fringe from the central maximum, mm
n = 5;          // Minimum value of n
m = Lambda_prime/Lambda*n;     // Minimum value of m
x4 = m*Lambda*D/(2*d);     // The least distance from the central maximum at which bright fringes duw to both the wavelengths coincide, mm
printf("\nThe distance of the third bright fringe from the central maximum = %4.2f mm", x3);
printf("\nThe least distance from the central maximum at which bright fringes duw to both the wavelengths coincide = %5.3f cm", x4/10);

// Result
// The distance of the third bright fringe from the central maximum = 1.17 mm
// The least distance from the central maximum at
```

```
   which bright fringes duw to both the wavelengths
   coincide = 0.156 cm
```

Scilab code Exa 4.13 Width of the fringes observed with the biprism

```
// Scilab code Ex4.13 : Pg:155 (2008)
clc;clear;
D = 80;     // Distance between the biprism and
   narrow slit, cm
Lambda = 5890e-08;     // Wavelength of light, cm
d = 0.05/2;     // Half of the distance between the
   sources, cm
omega = D*Lambda/(2*d);     // Fringe width, cm
printf("\nThe width of the fringes observed with the
    biprism = %5.3e cm", omega);

// Result
// The width of the fringes observed with the
   biprism = 9.424e-002 cm
```

Scilab code Exa 4.14 Fringe width at a distance of one meter from biprism

```
// Scilab code Ex4.14 : Pg:155 (2008)
clc;clear;
D = 110;     // Distance between the biprism and
   narrow slit, cm
Lambda = 5500e-08;     // Wavelength of light, cm
mu = 1.5;     // refractive index of glass biprism
a = 10;     // Distance of slit from biprism, cm
alpha = 2*%pi/180;     // Angle between the inclined
   faces and base of prism, degree
d = a*(mu-1)*alpha;     // Separation between two
   virtual sources, cm
```

```
omega = D*Lambda/(2*d);     // Fringe width at a
    distance of one meter from biprism , cm
printf("\nThe width of the fringes in the eye-piece
    from the biprism = %6.4f cm", omega);

// Result
// The width of the fringes in the eye-piece from
    the biprism = 0.0173 cm
```

Scilab code Exa 4.15 Wavelength of light used with the interference fringes produced by Fresnel biprism

```
// Scilab code Ex4.15 : Pg:156 (2008)
clc;clear;
d1 = 0.45;     // Position of the first lens placed
    between the biprism and the eye-piece , cm
d2 = 0.29;     // Position of the second lens placed
    between the biprism and the eye-piece , cm
omega = 0.0326;     // Fringe width , cm
D = 200;     // Distance between the biprism and
    narrow slit , cm
d = sqrt(d1*d2)/2;     // Separation between two
    virtual sources , cm
Lambda = 2*d*omega/D;     // Wavelength of light used
    , cm
printf("\nThe wavelength of light used = %4.2e cm",
    Lambda);

// Result
// The wavelength of light used = 5.89e-005 cm
```

Scilab code Exa 4.16 Wavelength of sodium light from Fresnel biprism experiment

```
// Scilab code Ex4.16 : Pg:156 (2008)
clc;clear;
omega = 0.0196;     // Fringe width, cm
D = 100;     // Distance between the biprism and
    narrow slit, cm
I = 0.70;     // Separation of the two coherent
    sources, cm
u = 30;     // Distance of the lens from the slit, cm
v = D - u;     // Distance of image from the lens, cm
// As magnification, M = I/O = v/u and O = 2*d,
    solving for d
d = I*u/(2*v);     // Half the distance between two
    coherent sources, cm
Lambda = 2*d*omega/D;     // Wavelength of light used
    , cm
printf("\nThe wavelength of light used = %4.2e cm",
    Lambda);

// Result
// The wavelength of light used = 5.88e-005 cm
```

Scilab code Exa 4.17 Wavelength of the light of the source in the biprism experiment

```
// Scilab code Ex4.17 : Pg:156 (2008)
clc;clear;
omega = 1.888/20;     // Fringe width, cm
D = 120;     // Distance between the biprism and
    narrow slit, cm
d = 0.075/2;     // Half the distance between two
    coherent sources, cm
Lambda = 2*d*omega/D;     // Wavelength of light used
    , cm
printf("\nThe wavelength of the light of the source
    = %4d angstrom", Lambda/1e-008);
```

```

// Result
// The wavelength of the light of the source = 5900
   angstrom
```

Scilab code Exa 4.18 Number of fringes in the biprism experiment with diiferent filters of mercury lamp

```
// Scilab code Ex4.18 : Pg:157 (2008)
clc;clear;
D = 1;     // For simplicity assume the distance
   between the biprism and narrow slit to be unity,
   unit
d = 1;     // Assume half the distance between two
   coherent sourcesto be unity, unit
lambda = 5893;     // Mean wavelength of sodium light
   , angstrom
lambda1 = 5461     // Wavelength of green color,
   angstrom
lambda2 = 4358;     // Wavelength of violet color,
   angstrom
omega = lambda*D/(2*d);     // Fringe width with
   yellow color, unit
omega1 = lambda1*D/(2*d);     // Fringe width with
   green color, unit
omega2 = lambda2*D/(2*d);     // Fringe width with
   violet color, unit
n = 62;     // Number of fringes obtained with light
   from sodium lamp
// As n1*omega1 = n*omega, solving for n1
n1 = n*omega/omega1;     // Number of fringes
   obtained with green color
// As n2*omega2 = n*omega, solving for n2
n2 = n*omega/omega2;     // Number of fringes
   obtained with violet color
```

```
printf("\nThe number of fringes with green filter =
   %2d", ceil(n1));
printf("\nThe number of fringes with violet filter =
    %2d", ceil(n2));

// Result
// The number of fringes with green filter = 67
// The number of fringes with violet filter = 84
// The second answer is given wrong in the textbook
```

Scilab code Exa 4.19 Distance between biprism and eye piece and wavelength of light

```
// Scilab code Ex4.19 : Pg:158 (2008)
clc;clear;
x1 = 100;     // Position of eye-piece, cm
x2 = 67;      // Position of first lens, cm
x3 = 34;      // Position of second lens, cm
v1 = x1 - x2;     // Distance between eye-piece and
   the second position of the lens, cm
u = v1;
x = x3 - u;     // The reading of the slit on the
   bench, cm
D = x1 - x;     // The distance between the focal
   plane of the eye-piece and the plane of the
   interfering sources, cm
d1 = 0.12;     // Position of the first lens placed
   between the biprism and the eye-piece, cm
d2 = 0.03;     // Position of the second lens placed
   between the biprism and the eye-piece, cm
omega = 0.972/10;     // Fringe width, cm
d = sqrt(d1*d2)/2;     // Separation between two
   virtual sources, cm
Lambda = 2*d*omega/D;     // Wavelength of light used
   , cm
```

```
printf("\nThe distance between the focal plane of the eye-piece and the plane of the interfering sources = %2d cm", D);
printf("\nThe wavelength of light used = %5.3e cm", Lambda);

// Result
// The distance between the focal plane of the eye-piece and the plane of the interfering sources = 99 cm
// The wavelength of light used = 5.891e-005 cm
```

Scilab code Exa 4.20 Refractive index of transparent plate in the two slit young interference experiment

```
// Scilab code Ex4.20 : Pg:159 (2008)
clc;clear;
D = 10;    // The distance between the slits and the screen, cm
d = 0.2/2;    // Half the separation between two slits, cm
lambda = 6000e-008;    // Wavelength of light used, cm
t = 0.05;    // Thickness of transparent plate, cm
x0 = 0.5;    // The shift of interference pattern, cm
// As x0 = D/(2*d)*(mu - 1)*t, solving for mu
mu = 2*d*x0/(D*t)+1;    // The refractive index of transparent plate
printf("\nThe refractive index of transparent plate = %3.1f", mu);

// Result
// The refractive index of transparent plate = 1.2
```

Scilab code Exa 4.21 Thickness of mica sheet in the double slit interference experiment

```
// Scilab code Ex4.21 : Pg:159 (2008)
clc;clear;
D = 50;    // The distance between the slits and the
    screen , cm
d = 0.1/2;    // Half the separation between two
    slits , cm
mu = 1.58;    // The refractive index of mica sheet
x0 = 0.2;    // The shift of interference pattern ,
    cm
// As x0 = D/(2*d)*(mu − 1)*t , solving for t
t = 2*d*x0/(D*(mu-1));    // Thickness of mica sheet
    , cm
printf("\nThe thickness of mica sheet = %3.1e cm", t
    );

// Result
// The thickness of mica sheet = 6.9e−004 cm
```

Scilab code Exa 4.22 Thickness of transparent material in two slit experiment

```
// Scilab code Ex4.22 : Pg:159 (2008)
clc;clear;
lambda = 5890e-008;    // Wavelength of light used ,
    cm
n = 12;    // Number of bright fringe to which the
    central fringe shifts
mu = 1.60;    // The refractive index of transparent
    material
```

```
t = n*lambda/(mu-1);    // Thickness of transparent
    material , cm
printf("\nThe thickness of the transparent material
    = %5.3e cm", t);

// Result
// The thickness of the transparent material = 1.178
    e-003 cm
// The answer is given wrong in the textbook
```

Scilab code Exa 4.23 Intensity and lateral shift of the central fringe

```
// Scilab code Ex4.23 : Pg:159 (2008)
clc;clear;
a = 1;    // Assume amplitude of the wave from
    coherent sources to be unity
D = 1;    // The distance between the slits and the
    screen , m
d = 5e-004/2;    // Half the separation between two
    slits , m
mu = 1.5;    // The refractive index of glass plate
t = 1.5e-006;    // Thickness of glass plate , m
lambda = 5000e-010;    // Wavelength of light used ,
    m
x0 = D/(2*d)*(mu - 1)*t;    // The lateral shift of
    central fringe , m
delta = (mu - 1)*t;    // Path difference created
    due to the introduction of the thin glass plate ,
    m
kro_delta = 2*%pi/lambda*delta;    // Phase
    difference , rad
a1 = a, a2 = a;    // Amplitude of waves from
    coherent sources
I = a1^2 + a2^2 + 2*a1*a2*cos(kro_delta);    //
    Intensity of central fringe
```

```
printf("\nThe lateral shift of central fringe = %4.2f cm", x0*100);
printf("\nThe intensity of central fringe = %d", I);

// Result
// The lateral shift of central fringe = 0.15 cm
// The intensity of central fringe = 0
// The first answer is given wrong in the textbook
```

Scilab code Exa 4.24 Shift in fringe position due to changed wavelength of path length

```
// Scilab code Ex4.24 : Pg:160 (2008)
clc;clear;
lambda = 5.9e-005;    // Wavelength of light, cm
lambda_prime = 7.5e-005;    // Chamged wavelength of light, cm
t = 0.002;    // Thickness of mica sheet, cm
mu = 1.5;    // Refractive index of mica
x0 = 0.237;    // Position of zeroth order fringe, cm
x10 = 0.355;    // Position of tenthth order fringe, cm
omega = (x10-x0)/10;    // Fringe width with original pattern, cm
// As omega = lambda*D/(2*d), so
omega_prime = omega*lambda_prime/lambda;    // New fringe width with changed wavelength, cm
x10_prime = x0+10*omega_prime;    // Position of tenth order fringe due to changed wavelength, cm
x_0 = omega/lambda*(mu - 1)*t;    // Shift in the zeroth fringe, cm
dx0 = [x_0 -x_0];
x0_prime = x0+dx0;    // Position of the zeroth order fringe due to changed path length, cm
```

```
printf("\nThe position of tenth order fringe due to changed wavelength = %4.2f mm", x10_prime*10);
printf("\nThe position of the zeroth order fringe due to changed path length = %4.2f mm or %4.2f mm", x0_prime(1)*10, x0_prime(2)*10);

// Result
// The position of tenth order fringe due to changed wavelength = 3.87 mm
// The position of the zeroth order fringe due to changed path length = 4.37 mm or 0.37 mm
```

Scilab code Exa 4.25 The smallest thickness of the plate which makes the glass plate dark by reflection

```
// Scilab code Ex4.25 : Pg:167 (2008)
clc;clear;
lambda = 5880e-008;    // Wavelength of light, cm
mu = 1.5;    // Refractive index of mica
r = 60;    // Angle of reflection in the plate, degree
n = 1;    // Order of fringes for the smallest thickness
t = n*lambda/(2*mu*cosd(r));    // The smallest thickness of the glass plate, cm
printf("\nThe smallest thickness of the glass plate = %4.0f angstrom", t/1e-008);

// Result
// The smallest thickness of the glass plate = 3920 angstrom
```

Scilab code Exa 4.26 Thickness of the film for which interference by reflection for violet component takes place

```
// Scilab code Ex4.26 : Pg:167 (2008)
clc;clear;
lambda = 4000e-008;    // Wavelength of light , cm
mu = 1.4;    // Refractive index of the film
r = 0;    // Angle of reflection in the plate ,
    degree
n = 1;    // Order of firnges for the smallest
    thickness
t = n*lambda/(4*mu*cosd(r));    // The thickness of
    the thinnest film , cm
printf("\nThe thickness of the thinnest film for
    reflection from violet component = %4.1f angstrom
    ", t/1e-008);

// Result
// The thickness of the thinnest film for reflection
     from violet component = 714.3 angstrom
```

Scilab code Exa 4.27 Thickness of the oil film

```
// Scilab code Ex4.27 : Pg:167 (2008)
clc;clear;
lambda = 5890e-008;    // Wavelength of light , cm
mu = 1.5;    // Refractive index of oil
i = 30;    // Angle of incidence , degree
n = 8;    // Order of dark band
sin_r = sind(i)/mu;    // Sine of angle of
    reflection from Snell's Law, degree
cos_r = sqrt(1-sin_r^2);    // Cosine of angle of
    reflection from the trigonometric identity ,
    degree
t = n*lambda/(2*mu*cos_r);    // The thickness of
```

```
the oil film, cm
printf("\nThe thickness of the oil film = %5.3e cm", t);

// Result
// The thickness of the oil film = 1.666e-004 cm
```

Scilab code Exa 4.28 Thickness of the soap film from interference by reflection

```
// Scilab code Ex4.28 : Pg:168 (2008)
clc;clear;
lambda1 = 6.1e-005;     // Wavelength corresponding to the first dark band, cm
lambda2 = 6.0e-005;     // Wavelength corresponding to the second dark band, cm
n = lambda2/(lambda1 - lambda2);     // Order of dark band
mu = 4/3;     // Refractive index of the film
sin_i = 4/5;     // Sine of ngle of incidence
sin_r = sin_i/mu;     // Sine of angle of reflection from Snell's Law, degree
cos_r = sqrt(1-sin_r^2);     // Cosine of angle of reflection from the trigonometric identity, degree
t = n*lambda1/(2*mu*cos_r);     // The thickness of the oil film, cm
printf("\nThe thickness of the soap film = %6.4f cm", t);

// Result
// The thickness of the soap film = 0.0017 cm
```

Scilab code Exa 4.29 Number of dark bands seen in the interference pattern between the given wavelength range

```
// Scilab code Ex4.29 : Pg:168 (2008)
clc;clear;
lambda1 = 4e-005;    // First wavelength, cm
lambda2 = 7e-005;    // Second wavelength, cm
t = 0.001;    // The thickness of the air film, cm
mu = 1;    // Refractive index of the air film
i = 30;    // Angle of incidence, degree
// As mu = sin_i/sin_r = 1, so that sin_i = sin_r
sin_r = sind(30);    // Sine of angle of reflection from Snell's Law, degree
cos_r = sqrt(1-sin_r^2);    // Cosine of angle of reflection from the trigonometric identity, degree
n1 = 2*mu*t*cos_r/lambda1;    // Number of dark bands seen at first wavelength
n2 = 2*mu*t*cos_r/lambda2;    // Number of dark bands seen at second wavelength
n = n1 - n2;    // Number of dark bands observed within the given spectral range
printf("\nThe number of dark bands observed within the given spectral range = %2d", ceil(n));

// Result
// The number of dark bands observed within the given spectral range = 19
```

Scilab code Exa 4.30 Fringe width in air wedge for normal incidence

```
// Scilab code Ex4.30 : Pg:180 (2008)
clc;clear;
Lambda = 6000e-08;    // Wavelength of light, cm
d = 0.005;    // Diameter of wire, mm
```

```
x = 15;     // Distance between the glass plates, cm
theta = d/x;     // Angle of the wedge, degree
omega = Lambda/(2*theta);     // Fringe width in air
   wedge for normal incidence, cm
printf("\nThe fringe width in air-wedge for normal
   incidence = %4.2f cm", omega);

// Result
// The fringe width in air-wedge for normal
   incidence = 0.09 cm
```

Scilab code Exa 4.31 Angle of the wedge

```
// Scilab code Ex4.31: : Pg:181 (2008)
clc;clear;
Lambda = 6000e-08;     // Wavelength of light, cm
mu = 1.35;     // Refractive index of thin wedge
   shaped film
omega = 0.20;     // Fringe width, cm
// As omega = Lambda/(2*mu*theta), solving for theta
theta = Lambda/(2*mu*omega)*180/%pi;     // Angle of
   the wedge, degree
printf("\nThe angle of the wedge = %6.4f degree",
   theta);

// Result
// The angle of the wedge = 0.0064 degree
```

Scilab code Exa 4.32 Thickness of the wire

```
// Scilab code Ex4.32: : Pg:181 (2008)
clc;clear;
Lambda = 5890e-08;     // Wavelength of light, cm
```

```
n = 20;      // Number of fringes
// Since omega = Lambda*x/2*t and x = n*omega, solving for t
t = n*Lambda/2;      // Thickness of the wire, cm
printf("\nThe thickness of the wire = %4.2e cm", t);

// Result
// The thickness of the wire = 5.89e-004 cm
```

Scilab code Exa 4.33 Wedge shaped air film between two optically plane glass plates

```
// Scilab code Ex4.33: : Pg:182 (2008)
clc;clear;
Lambda = 5.46e-05;      // Wavelength of light, cm
n = 12;      // Number of fringes
d = 0.40;      // Spacing between 12 fringes, cm
omega = d/n;      // Fringe width, cm
// Since fringe width in air wedge for normal incidence is given by omega = Lambda/2*theta. On solving for theta, we have
// As omega = Lambda/(2*theta), solving for theta
theta = Lambda/(2*omega);      // Angle of the wedge, radian
l = 3;      // Length of the plate, cm
t = theta*l;      // Thickness of the foil, cm
mu = 1.33;      // Refractive index of water
omega_prime = Lambda/(2*mu*theta);      // Fringe width if water is introduced in the wedge space in Newton's ring experiment, cm
printf("\nThe angle of the wedge = %3.1e radian", theta);
printf("\nThe thickness of the foil = %4.2e cm", t);
printf("\nThe fringe width if water is introduced in the wedge space = %5.3f cm", omega_prime);
```

```
// Result
// The angle of the wedge = 8.2e-004 radian
// The thickness of the foil = 2.46e-003 cm
// The fringe width if water is introduced in the
   wedge space = 0.025 cm
```

Scilab code Exa 4.34 Angular diameter of bright fringe

```
// Scilab code Ex4.34: : Pg:188 (2008)
clc;clear;
Lambda = 5896e-08;    // Wavelength of light, cm
d = 0.3;    // Path difference between the M1 and M2
    mirrors, cm
r = 0;    // For central bright fringe
// Since 2*d*cos(r) = n*Lambda and for r = 0 which
   gives 2*d = n*Lambda
// 2*d*cos_theta = (n-6)*Lambda, solving for theta
theta = acosd(1-6*Lambda/(2*d));    // Angular
   radius of the seventh bright fringe, degree
D = 2*theta;    // Angular diameter of the seventh
   bright fringe, degree
printf("\nThe angular diameter of 7th bright fringe
   = %1.0f degree", D);

// Result
// The angular diameter of 7th bright fringe = 4
   degree
```

Scilab code Exa 4.35 Wavelength of light

```
// Scilab code Ex4.35: : Pg:188 (2008)
clc;clear;
```

```
N = 500;     // Number of fringes
x = 0.01474;     // Distance traversed by the mirror when N fringes cross the field of view, cm
//Since x = N*Lambda/2, solving for Lambda
Lambda = 2*x/(N*1e-08);     // wavelngth of light, angstrom
printf("\nThe wavelength of light = %4.0f angstrom", Lambda);

// Result
// The wavelength of light = 5896 angstrom
```

Scilab code Exa 4.36 Difference in the wavelengths of the D1 and D2 lines of the sodium lamp

```
// Scilab code Ex4.36: : Pg:188 (2008)
clc;clear;
x = 0.0289;     // Distance traversed by the mirror between two successive disappearances, cm
Lambda = 5890e-08;     // Wavelength of light, cm
delta_Lambda = Lambda^2/(2*x);     // Difference in the wavelengths of the D1 and D2 lines of the sodium lamp, cm
printf("\nThe difference in the wavelengths of the D1 and D2 lines of the sodium lamp = %1.0e cm", delta_Lambda);

// Result
// The difference in the wavelengths of the D1 and D2 lines of the sodium lamp = 6e-008 cm
```

Chapter 5

Diffraction of Light

Scilab code Exa 5.1 Distance between the first and fourth band

```
// Scilab code Ex5.1: Pg:200 (2008)
clc;clear;
a = 300;     // Distance between narrow slit and
   straight edge, cm
b = 600;     // Distance between straight edge and
   screen, cm
Lambda = 4900e-08;     // Wavelength of light, cm
// For n = 1
n = 1;
x_1 = sqrt(b*(a + b)*Lambda/a)*sqrt(2*n);     //
   Distance of Ist minimum outside the geometrical
   shadow
// For n = 4
n = 4;
x_4 = sqrt(b*(a + b)*Lambda/a)*sqrt(2*n);     //
   Distance of fourth minimum outside the
   geometrical shadow
x = x_4 - x_1;     // Distance between first and
   fourth band, cm
printf("\nThe distance between the first and fourth
   band = %4.2f cm", x);
```

```

// Result
// The distance between the first and fourth band =
   0.42 cm
```

Scilab code Exa 5.2 Angular position of first two minima on either side of the central maxima

```
// Scilab code Ex5.2: : Pg:207 (2008)
clc;clear;
// Define function to convert degrees to degree and
   minute
function [deg, minute] = deg2degmin(theta)
    deg = floor(theta);
    minute = ceil((theta-deg)*60);
endfunction
a = 22e-05;    // Width of slit , cm
Lambda = 5500e-08;    // Wavelength of light , cm
// Since a*sin(theta) = n*Lambda, solving for sin(
   theta_1)
n = 1;    // First order minimum
theta_1 = asind(n*Lambda/a);    // Angular position
   of first order minimum, degree
[d1, m1] = deg2degmin(theta_1);    // Transformtion
    function
n = 2;    // Second order minimum
theta_2 = asind(n*Lambda/a);    // Angular position
   of second order minimum, degree
[d2, m2] = deg2degmin(theta_2);    // Transformtion
    function
printf("\nThe angular position of first order minima
    = %d degree %d minute", d1, m1);
printf("\nThe angular position of second order
   minima = %d degree %d minute", d2, m2);

```

```
// Result
// The angular position of first order minima = 14 degree 29 minute
// The angular position of second order minima = 30 degree 1 minute
```

Scilab code Exa 5.3 The wavelengths of incident light in diffraction pattern

```
// Scilab code Ex5.3: Pg:207 (2008)
clc;clear;
a = 0.04;    // Width of slit, cm
Lambda = 5500e-08;    // Wavelength of light, cm
x = 0.5;    // Distance from the central maximum at which both fourth and fifth minimum occur, cm
f = 100;    // Focal length of lens, cm
theta = x/f;    // Angle of diffraction, radian
// As a*sin(theta) = 4*Lambda_1 = 5*Lambda_2, solving for Lambdas
Lambda_1 = a*sin(theta)/4;    // First wavelength, cm
Lambda_2 = 4*Lambda_1/5;    // Second wavelength, cm
printf("\nThe two wavelengths of incident lights are :\nLambda_1 = %1.0e cm; Lambda_2 = %1.0e cm", Lambda_1, Lambda_2);

// Result
// The two wavelengths of incident lights are:
// Lambda_1 = 5e-005 cm; Lambda_2 = 4e-005 cm
```

Scilab code Exa 5.4 Wavelength of spectral line

```
// Scilab code Ex5.4: : Pg:216 (2008)
```

```
clc;clear;
aplusb = 1/1250;     // Grating element where a is
    the width of slit and b is the width of opaque
    region in a grating , cm
theta = 30;     // Direction of principal maxima ,
    degree
n = 2;     // Second order principal maxima
Lambda = aplusb*sind(theta)/n;     // Wavelength of
    spectral line , angstrom
printf("\nThe wavelength of spectral line = %d
    angstrom", ceil(Lambda/1e-008));

// Result
// The wavelength of spectral line = 20000 angstrom
```

Scilab code Exa 5.5 Number of lines on the grating surface

```
// Scilab code Ex5.5: Pg:217 (2008)
clc;clear;
Lambda = 5e-05;     // Wavelength of spectral line ,
    cm
n = 2;     // Second order principal maxima
theta = 30;     // Direction of principal maxima ,
    degree
aplusb_inv = sind(theta)/(n*Lambda);     // Number of
     lines in one cm of grating where a is the width
    of slit and b is the width of opaque region in a
    grating , cm
printf("\nThe number of lines on the grating surface
     = %d ", ceil(aplusb_inv));

// Result
// The number of lines on the grating surface = 5000
```

Scilab code Exa 5.6 Direction of principal maxima

```
// Scilab code Ex5.6: Pg:217 (2008)
clc;clear;
Lambda = 6e-05;     // Wavelength of spectral line, cm
n = 1;     // First order principal maxima
aplusb = 1/160;     // Grating element where a is the width of slit and b is the width of opaque region in a grating, cm
// since the grating equation is given by (a +b)*sint_theta = n*Lambda. On solving fot theta, we have
theta = asind(n*Lambda/aplusb);     // Direction of principal maxima, minutes
printf("\nThe direction of principal maxima = %2d minutes", theta*60);

// Result
// The direction of principal maxima = 33 minutes
```

Scilab code Exa 5.7 Angle of diffraction in first order

```
// Scilab code Ex5.7:  Pg:217 (2008)
clc;clear;
// Define function to convert degrees to degree and minute
function [deg, minute] = deg2degmin(theta)
    deg = floor(theta);
    minute = ceil((theta-deg)*60);
endfunction
```

```
Lambda = 5e-05;     // Wavelength of spectral line, cm
n = 1;     // First order principal maxima
aplusb = 3/15000;     // Grating element where a is the width of slit and b is the width of opaque region in a grating, cm
// Since (a +b)*sint_theta = n*Lambda, solving fot theta
theta = asind((n*Lambda/aplusb));     // Angle of diffraction in first order, minutes
[d, m] = deg2degmin(theta);
printf("\nThe angle of diffraction in first order = %2d degree %2d minutes", d, m);

// Result
// The angle of diffraction in first order = 14 degree 29 minutes
```

Scilab code Exa 5.8 Dispersive powers of first and third order spectra of diffraction grating

```
// Scilab code Ex5.8: Pg:218 (2008)
clc;clear;
Lambda = 5000;     // Wavelength of spectral line, Angstorm
n = 1;     // First order principal maxima
n = 3;     // Third order principal maxima
aplusb = 18000;     // Grating element where a is the width of slit and b is the width of opaque region in a grating, cm
n = 1;      // First order diffraction
tl_ratio_1 = 1/sqrt((aplusb/n)^2-Lambda^2);     // Angular dispersion produced by a grating around a mean wavelength lambda, radian per angstorm
n = 3;      // Second order diffraction
```

```
tl_ratio_3 = 1/sqrt((aplusb/n)^2-Lambda^2);    // Angular dispersion produced by a grating around a mean wavelength lambda, radian per angstorm
printf("\nThe dispersive powers of first and third order spectra of diffraction grating are %4.2e rad/angstrom and %3.1e rad/angstrom", tl_ratio_1, tl_ratio_3);

// Result
// The dispersive powers of first and third order spectra of diffraction grating are 5.78e-005 rad/angstrom and 3.0e-004 rad/angstrom
```

Scilab code Exa 5.9 Difference in two wavelengths

```
// Scilab code Ex5.9: Pg:218 (2008)
clc;clear;
Lambda = 5000;    // Wavelength of spectral line, Angstorm
theta = 30;    // Direction of principal maxima, degree
d_theta = 0.01;    // Angular separation between two wavelengths, radians
d_Lambda = Lambda*cotd(theta)*d_theta;    // Difference in two wavelengths, angstrom
printf("\nThe difference in two wavelengths = %4.1f angstrom", d_Lambda);

// Result
// The difference in two wavelengths = 86.6 angstroms
```

Scilab code Exa 5.10 Dispersion in the spectrograph and separation between the spectral lines

```
// Scilab code Ex5.10:  Pg:219 (2008)
clc;clear;
Lambda = 5.9e-05;    // Wavelength of spectral line, Angstorm
n = 2;    // Second order principal maxima
f = 25;    // focal length of the convex lens, cm
aplusb = 2.54/15000;    // Grating element where a is the width of slit and b is the width of opaque region in a grating, cm
sin_theta = n*Lambda/aplusb;
// Since (a +b)*sin_theta = n*Lambda, solving for cos_theta
cos_theta = sqrt(1-sin_theta^2);
tl_ratio = n/(aplusb*cos_theta);    // Angular dispersion produced by grating, radians per Angstorm
xl_ratio = f*(tl_ratio);    // Linear dispersion in the spectrograph, radian per Angstorm
d_Lambda = 6;    // Separation between two wavelengths, Angstorm
d_x = xl_ratio*1e-008*d_Lambda;    // Separation between spectral lines, cm
printf("\nThe angular dispersion produced by the grating = %3.1e rad/angstrom", tl_ratio*1e-008);
printf("\nThe linear dispersion in the spectrograph = %1.0e cm/Angstorm", xl_ratio*1e-008);
printf("\nThe separation between spectral lines = %3.1e cm", d_x);

// Result
// The angular dispersion produced by the grating = 1.6e-004 rad/angstrom
// The linear dispersion in the spectrograph = 4e-003 cm/Angstorm
// The separation between spectral lines = 2.5e-002
```

cm

Scilab code Exa 5.11 Separation between two spectral lines in the first order spectrum

```
// Scilab code Ex5.11: Pg:219 (2008)
clc;clear;
Lambda_1 = 5000e-08;    // First wavelength of spectral line, cm
Lambda_2 = 5200e-08;    // Second wavelength of spectral line, cm
aplusb = 1/10000;    // Grating element where a is the width of slit and b is the width of opaque region in a grating, cm
f = 150;    // Focal length of the lens, cm
n = 1;    // Order of diffractions
// Since (a +b)*sin_theta = n*Lambda
theta_1 = asind(n*Lambda_1/aplusb);    // Angle of diffraction for the first order with first wavelength, degree
theta_2 = asind(n*Lambda_2/aplusb);    // Angle of diffraction for the first order with second wavelength, degree
x_1 = tand(theta_1)*f;    // Position of first spectral line in the first order spectrum, cm
x_2 = tand(theta_2)*f;    // Position of second spectral line in the first order spectrum, cm
d_x = x_2 - x_1;    // Separation between two spectral lines in the first order spectrum, cm
printf("\nThe separation between two spectral lines in the first order spectrum = %4.2f cm", d_x);

// Result
// The separation between two spectral lines in the first order spectrum = 4.71 cm
```

Scilab code Exa 5.12 Resolving power of a grating in the second order

```
// Scilab code Ex5.12: Pg:224 (2008)
clc;clear;
n = 2;    // Second order diffraction
N = 40000;    // Number of lines per inch on the
    diffraction grating
lambda_ratio = n*N;    // Resolving power of grating
     in second order where d_Lambda is the smallest
    wavelength difference between neighbouring lines
printf("\nThe resolving power of a grating in the
    second order = %d ", lambda_ratio);

// Result
// The resolving power of a grating in the second
    order = 80000
```

Scilab code Exa 5.13 Minimum number of lines in the plane diffraction grating in the first and second order spectra

```
// Scilab code Ex5.13: Pg:224 (2008)
clc;clear;
n_1 = 1;    // First order diffraction
n_2 = 2;    // Second order diffraction
Lambda_1 = 5890;    // First wavelength of sodium
    light , Angstorm
Lambda_2 = 5896;    // Second wavelength of sodium
    light , Angstorm
Lambda = (Lambda_1 + Lambda_2)/2;    // Mean
    wavelength , angstorm
d_Lambda = Lambda_2 - Lambda_1;    // Difference in
    wavelength , Angstorm
```

```
N1 = Lambda_1/(n_1*d_Lambda);     // Number of lines in a plane diffraction grating required to just resolve the sodium doublet in the first order
N2 = Lambda_2/(n_2*d_Lambda);     // Number of lines in a plane diffraction grating required to just resolve the sodium doublet in the second order
printf("\nThe minimum number of lines in the plane diffraction grating in the first and second order spectra respectively are %d and %d", ceil(N1), N2);

// Result
// The minimum number of lines in the plane diffraction grating in the first and second order spectra respectively are 982 and 491
```

Scilab code Exa 5.14 Wavelength difference in the first order spectrum

```
// Scilab code Ex5.14: Pg:225 (2008)
clc;clear;
n = 1;     // First order diffraction
N = 1000;     // Number of lines on the grating
Lambda = 6e-05;     // Wavelength of light, cm
// Let Lambda and d_Lambda be the two wavelengths in the first order spectrum. Since the resolving power of a grating is given by Lambda/d_Lambda = n*N. On solving for d_lambda, we have
d_Lambda = Lambda/(n*N);     // Difference between two wavelength in the first order spectrum, Angstorm
printf("\nThe wavelength difference in the first order spectrum = %d angstrom", d_Lambda/1e-008);

// Result
// The wavelength difference in the first order
```

spectrum = 6 angstrom

Scilab code Exa 5.15 Maximum resolving power for normal incidence

```
// Scilab code Ex5.15: Pg:225 (2008)
clc;clear;
Lambda = 5080e-08;    // Wavelength of light on the
   grating, cm
theta = 90;    // Angle of incidence of light on
   grating, degree
d = 2.54;    // Total ruled width of grating, cm
frac_lambda_max = d/Lambda;
printf("\nThe maximum resolving power = %1.0e ",
   frac_lambda_max);

// Result
// The maximum resolving power = 5e+004
```

Scilab code Exa 5.16 Resolving power of the grating in the second order

```
// Scilab code Ex5.16: Pg:225 (2008)
clc;clear;
Lambda_1 = 5140.34;    // First wavelength of light
   on the grating in the first order, angstrom
Lambda_2 = 5140.85;    // Second wavelength of light
    on the grating in the first order, angstrom
Lambda_3 = 8037.20;    // First wavelength of light
   on the grating in the second order, angstrom
Lambda_4 = 8037.50;    // Second wavelength of light
    on the grating in the second order, angstrom
Lambda = (Lambda_1 + Lambda_2)/2;    //Mean
   wavelength for the first order diffraction,
   angstrom
```

```
d_Lambda = Lambda_2 - Lambda_1;     // Smallest wavelength difference at the mean wavelength Lambda for the first order diffraction, angstrom
n = 1;     // First order diffraction
// As RP_1 = Lambda/d_Lambda = n*N, solving for N
N = 1/n*Lambda/d_Lambda;     // Number of lines on the diffraction grating for the first order diffraction
n = 2;     // Second order diffraction
RP2 = n*N;     // Expected resolving power of grating in the second order
Lambda = (Lambda_3 + Lambda_4)/2;     // Mean wavelength for the second order diffraction, angstrom
d_Lambda = Lambda_4 - Lambda_3;     // Smallest wavelength difference at the mean wavelength Lambda for the second order diffraction, angstrom
RP = Lambda/d_Lambda;     // Calculated resolving power of grating in the second order
if (RP > RP2) then
    printf("The grating will not be able to resolve the lines %7.2f angstrom and %7.2f angstrom", Lambda_3, Lambda_4);
else
    printf("The grating will be able to resolve the lines %7.2f angstrom and %7.2f angstrom", Lambda_3, Lambda_4);
end

// Result
// The grating will not be able to resolve the lines 8037.20 angstrom and 8037.50 angstrom
```

Scilab code Exa 5.17 Wavelength of spctral lines and minimum grating width in the second order spectrum of diffraction grating

```
// Scilab code Ex5.17: Pg:226 (2008)
clc;clear;
n = 2;    // Second order diffraction
theta = 10;    // Angle of diffraction , degree
d_Lambda = 5e-009;    // Wavelength of second spectral line of light on the grating in the second order , cm
d_theta = (3/3600)*(%pi/180);    // Differential angle of diffraction , rad
Lambda = sind(theta)*d_Lambda/(cosd(theta)*d_theta);    // Wavelength of spectral line , cm
N = (Lambda/d_Lambda)*1/n;    // Number of lines on the grating
w_min = N*n*Lambda/sind(theta);    // Minimum grating width of diffraction grating required to resolve the spectral lines , cm
printf("\nThe wavelength of first spectral line = %4.0f angstrom", Lambda/1e-008);
printf("\nThe wavelength of Second spectral line = %6.1f angstrom", (Lambda+d_Lambda)/1e-008);
printf("\nThe minimum grating width of diffraction grating required to resolve the spectral lines = %3.1f cm", w_min);

// Result
// The wavelength of first spectral line = 6062 angstrom
// The wavelength of Second spectral line = 6062.2 angstrom
// The minimum grating width of diffraction grating required to resolve the spectral lines = 4.2 cm
// The answer is given wrong in the textbook
```

Scilab code Exa 5.18 Smallest wavelength difference in the second order

```
// Scilab code Ex5.18: Pg:227 (2008)
clc;clear;
n = 2;    // Order of diffraction
Lambda = 6000e-08;    // Wavelength of light on the
    grating, cm
m = 16000;    // Number of lines per inch on grating
L = 5;    // Length of the ruled grating, inches
N = L*m;    // Total number of lines on the grating
// Since the resolving power, Lambda/d_Lambda = n*N,
    solving for d_Lambda
d_Lambda = Lambda/(n*N);    // The smallest
    wavelength difference, Angstorm
printf("\nThe smallest wavelength difference in the
    second order = %6.4f angstrom",d_Lambda/1e-008);

// Result
// The smallest wavelength difference in the second
    order = 0.0375 angstrom
```

Scilab code Exa 5.19 Resolution of smallest difference of wavelengths by a spectrometer

```
// Scilab code Ex5.19: Pg:229 (2008)
clc;clear;
t = 5;    // width of the base of the prism, cm
Lambda = 5000;    // wavelength, angstrom
D = 200;    // Rate of change of refractive index
    with wavelength, per cm
RP = t*D;    // Resolving power of a prism
d_Lambda = Lambda/(D*t);    // Smallest difference
    in wavelengths by a spectrometer, angstrom
printf("\nThe resolution of smallest difference of
    wavelengths by a spectrometer = %d angstrom",
    d_Lambda);

```

```
// Result
// The resolution of smallest difference of
   wavelengths by a spectrometer = 5 angstrom
```

Scilab code Exa 5.20 Length of base of a flint glass prism

```
// Scilab code Ex5.20: Pg:229 (2008)
clc;clear;
Lambda_1= 5896;     // Wavelength of D1 Sodium light ,
    Angstorm
Lambda_2= 5890;     // Wavelength of D2 Sodium light ,
    Angstorm
Lambda = (Lambda_1 + Lambda_2)/2;     // Mean
   wavelength of sodium light , Angstorm
d_Lambda = Lambda_1 - Lambda_2;     // Difference in
   wavelengths of sodium , Angstorm
RP = Lambda/d_Lambda;     // Resolving power of prism
D = 982;     // Rate of change of refractive index
   with wavelength , per cm
// As RP = t*D, solving for t
t =1/D*RP;     // Length of base of a flint glass
   prism , cm
printf("\nThe length of base of a flint glass prism
   = %3.1f cm", t);

// Result
// The length of base of a flint glass prism = 1.0
   cm
```

Scilab code Exa 5.21 Smallest difference of wavelengths resolved by a prism of flint glass

```
// Scilab code Ex5.21: Pg:229 (2008)
```

```
clc;clear;
mu_C = 1.6389;    // Refractive index index of material
mu_F = 1.7168;    // Refractive index index of material
Lambda_C = 6563e-008;    // Wavelength of C Sodium light, Angstorm
Lambda_F = 4861e-008;    // Wavelength of F Sodium light, Angstorm
Lambda = 5e-05;    // Wavelength of light, cm
t = 3;    // Length of base of a flint glass prism, cm
// Since the resolving power of a spectrometer is given by Lambda/d_Lambda. Thus
D = (mu_F - mu_C)/(Lambda_C - Lambda_F);    // Dispersion of material of the prism
d_Lambda = Lambda/(t*D);    // Resolving power of a prism
printf("\nThe smallest difference of wavelengths resolved by the flint glass prism = %4.2f angstrom", d_Lambda/1e-008);

// Result
// The smallest difference of wavelengths resolved by the flint glass prism = 0.36 angstrom
// The answer is given wrong in the textbook
```

Scilab code Exa 5.22 Size of the grating interval

```
// Scilab code Ex5.22: Pg:230 (2008)
clc;clear;
Lambda_1 = 6708e-008;    // Wavelength, Angstorm
Lambda_2 = 6438e-008;    // wavelength, Angstorm
n = 2;    // Order of diffraction
mu_1 = 1.5400;    // Refractive index index of
```

```
    material
mu_2 = 1.5412;     // Refractive index index of
    material
D = (mu_2 - mu_1)/(Lambda_1 - Lambda_2);     //
    Dispersion of the material of the grating, per cm
aplusb = n/D;     // Size of the grating interval, cm
printf("\nThe size of the grating interval = %3.1e
    cm", aplusb);

// Result
// The size of the grating interval = 4.5e-003 cm
// The answer is given wrong in the textbook
```

Scilab code Exa 5.23 Smallest angular separation of two stars resolved by a telescope

```
// Scilab code Ex5.23: Pg:232 (2008)
clc;clear;
Lambda = 5600e-08;     //Mean wavelength of light, cm
a = 101.6;     // Diameter of the objective of a
    telescope, cm
theta_1 = 1.22*Lambda/a;     // The smallest angular
    separation of two stars in seconds resolved by a
    telescope, radian
theta = theta_1*(180/%pi)*60*60;     // Smallest
    angular separation of two stars in seconds
    resolved by a telescope, second
printf("\nThe smallest angular separation of two
    stars in seconds resolved by a telescope = %4.2f
    second", theta);

// Result
// The smallest angular separation of two stars in
    seconds resolved by a telescope = 0.14 second
```

Scilab code Exa 5.24 Diameter of an objective of a telescope

```
// Scilab code Ex5.24: Pg:232 (2008)
clc;clear;
Lambda = 5000e-08;    //Mean wavelength of light, cm
theta = 10e-03;    // Smallest angular separation resolvable by a telescope objective, degree
theta = %pi/180*(1/1000);    // The smallest angular separation resolvable by a telescope objective, radian
// As theta = (1.22*Lambda)/a, solving for a
a = 1.22*Lambda/theta;    // Diameter of an objective of the telescope, cm
printf("\nThe diameter of an objective of the telescope = %3.1f cm", a);

// Result
// The diameter of an objective of the telescope = 3.5 cm
```

Scilab code Exa 5.25 The distance between two objects on the moon and the magnifying power of a telescope

```
// Scilab code Ex5.25: Pg:232 (2008)
clc;clear;
Lambda = 5800e-08;    // Mean wavelength of light, cm
a = 20;    // Diameter of the objective of a telescope, cm
theta = 1.22*Lambda/a;    // The smallest angular separation resolvable by a telescope objective of diameter a, radian
```

```
l = 4e+05;     // Distance of moon from the earth, km
x = theta * l;     // Distance between two objects on the moon, km
theta = 1.22*Lambda/a;     // Angular resolution of the eye
theta_prime = 1.5*%pi/180*1/60;     // Angular resolution of the telescope, degree
MP = theta_prime/theta;     // Magnifying power of a telescope
printf("\nThe distance between two objects on the moon = %3.1f km", x);
printf("\nThe magnifying power of the telescope = %3d ", MP);

// Result
// The distance between two objects on the moon = 1.4 km
// The magnifying power of the telescope = 123
```

Scilab code Exa 5.26 Minimum linear resolvable distance between two person

```
// Scilab code Ex5.26: Pg:233 (2008)
clc;clear;
Lambda = 5.5e-05;     //Mean wavelength of light, cm
a = 0.2;     // Diameter of the pupil of an eye, cm
theta = 1.22*Lambda/a;     // The smallest angular separation resolvable by a human eye of pupil diameter a, radian
l = 5000;     // Distance of person from the man, cm
x = theta * l;     // Minimum linear resolvable distance between two person, cm
printf("\nThe minimum linear resolvable distance between two persons = %4.3f cm",x);

```

```
// Result
// The minimum linear resolvable distance between
   two persons = 1.678 cm
```

Scilab code Exa 5.27 Minimum focal length of the objective if the full resolving power of the telescope is to be utilized

```
// Scilab code Ex5.27: Pg:233 (2008)
clc;clear;
Lambda = 6000e-08;    //Mean wavelength of light, cm
a = 200;    // Diameter of the objective of a
   telescope, cm
a_prime = 0.2;    // Aperture of the eye lens, cm
f = 2.54;    // Focal length of eye-piece, cm
theta = 1.22*Lambda/a;    // The smallest angular
   separation resolvable by a telescope objective of
    diameter a, radian
theta_prime = 1.22*Lambda/a_prime;    // The
   smallest angle that can be resolved by the eye
   where a^' is the aperture of the eye, radian
MP = theta_prime/theta;    // Magnifying power of
   the telescope
// As MP = F/f, solving for F
F = MP*f;    // The minimum focal length of the
   objective, cm
printf("\nThe minimum focal length of the objective
   if the full resolving power of the telescope is
   to be utilized = %4d cm", F);

// Result
// The minimum focal length of the objective if the
   full resolving power of the telescope is to be
   utilized = 2540 cm
```

Scilab code Exa 5.28 Resolving limit of a microscope

```
// Scilab code Ex5.28: Pg:236 (2008)
clc;clear;
Lambda = 5500e-08;    // Wavelength of the visible
    light, cm
theta = 30;    // Semi-angle of the cone of light,
    degree
x = 1.22*Lambda/(2*sind(theta));    // Distance
    between the two nearby objects just resolved by
    the microscope, cm
printf("\nThe resolving limit of the microscope = %3
    .1e cm", x);

// Result
// The resolving limit of the microscope = 6.7e-005
    cm
```

Scilab code Exa 5.29 Resolving power of a microscope

```
// Scilab code Ex5.29: Pg:236 (2008)
clc;clear;
Lambda = 6e-05;    // Wavelength of the light, cm
NA = 0.12;    // numerical aperture
x = Lambda/(2*NA);    // Minimum resolvable distance
     between two nearby objects
RP = 1/x;    // Resolving power of a microscope
printf("\nThe resolving power of the microscope =
    %4d ", RP);

// Result
// The resolving power of the microscope = 4000
```

Scilab code Exa 5.30 Magnifying power of a microscope

```
// Scilab code Ex5.30: Pg:236 (2008)
clc;clear;
L_1 = 5e-05;     // Limit of resolution of microscope
    , cm
l = 25;     // Least distance of distiinct vision, cm
theta_1 = 1.5;     // Angular limit of resolution of
    eye, minute
theta_2 = theta_1/60*%pi/180;     // Angular limit of
    resolution of eye, radian
L_2 = l*theta_2;     // Linear limit of the
    resolution of eye, cm
M = L_2/L_1;     // Magnifying power of the
    microscope
printf("\nThe magnifying power of the microscope =
    %3d ", M);

// Result
// The magnifying power of the microscope = 218
```

Chapter 6

Polarization of Light

Scilab code Exa 6.1 Refractive index of the material and angle of refraction

```
// Scilab code Ex6.1: Pg:247 (2008)
clc;clear;
i_p = 60;    // Angle of polarization, degree
mu = tand(i_p);    // Refractive index of the
   material
r = 90-i_p;    // Angle of refraction, degree
printf("\nThe refractive index of the material = %5
   .3f ", mu);
printf("\nThe angle of refraction = %2d degree", r);

// Result
// The refractive index of the material = 1.732
// The angle of refraction = 30 degree
```

Scilab code Exa 6.2 Angle of refraction in benzene

```
// Scilab code Ex6.2: Pg:247 (2008)
```

```
clc;clear;
mu = 1.50;     // Refractive index of the material
// Since mu = tan i_p , solving for i_p
i_p = atand(mu);     // Angle of polarization , degree
r = 90-i_p;     // Angle of refraction , degree
printf("\nThe angle of polarization = %4.1f degree",
    i_p);
printf("\nThe angle of refraction = %4.1f degree", r
   );

// Result
// The angle of polarization = 56.3 degree
// The angle of refraction = 33.7 degree
```

Scilab code Exa 6.3 Comparison of polarizing angle from two different media

```
// Scilab code Ex6.3: Pg:248 (2008)
clc;clear;
mu_glass = 1.54;     // Refractive index of the glass
mu_water = 1.33;     // Refractive index of the water
mu_1 = mu_glass/mu_water;     // Refractive index for
    a water to glass interface
mu_2 = mu_water/mu_glass;     // Refractive index for
    a glass to water interface
// Since mu = tan i_p , solving for i_p
i_p_1 = atand(mu_1);     // Angle of polarization for
    water to glass interface , degree
i_p_2 = atand(mu_2);     // Angle of polarization for
    glass to water interface , degree
printf("\nThe polarizing angle for the water to
   glass interface is larger than that of glass to
   water inteface by %3.1f degree", i_p_1 - i_p_2);

// Result
```

```
// The polarizing angle for the water to glass
   interface is larger than that of glass to water
   inteface by 8.4 degree
```

Scilab code Exa 6.4 Angle of minimum deviation

```
// Scilab code Ex6.4: Pg:248 (2008)
clc;clear;
A = 60;     // Angle of prism, degree
i_p = 60;     // Polarizing angle, degree
mu = tand(i_p);     // Refractive index of glass
// Since mu = sind((A + d_m)/2)/sind(A/2), solving
   for d_m
d_m = 2*asind(mu*sind(A/2)) - A;     // Angle of
   minimum deviation, degree
printf("\nThe angle of minimum deviation = %2d
   degree", ceil(d_m));

// Result
// The angle of minimum deviation = 60 degree
// The answer is given wrongly in the textbook
```

Scilab code Exa 6.5 Angle between two polarizing sheets

```
// Scilab code Ex6.5: Pg:249 (2008)
clc;clear;
// Define function to convert degrees to degree and
   minute
function [deg, minute] = deg2degmin(theta)
    deg = floor(theta);
    minute = ceil((theta-deg)*60);
endfunction
```

```
I_m = 1;     // For simplicity assume maximum intensity to be unity, unit
I0 = I_m;     // Initial intensity, unit
I = I_m/3;     // Final intensity, unit
// From Malus' Law. I = I0*cosd(theta)^2, solving for theta
theta = acosd(sqrt(I/I0));     // The angle between two polarizing sheets, degree
[d1, m1] = deg2degmin(theta);     // Call conversion function
[d2, m2] = deg2degmin(180-theta);     // Call conversion function for supplement
printf("\nThe angle between two polarizing sheets = %2d degree %2d minute = %2d degree %2d minute", d1, m1, d2, m2);

// Result
// The angle between two polarizing sheets = 54 degree 45 minute = 125 degree 16 minute
// The answer is given wrongly in the textbook
```

Scilab code Exa 6.6 Intensity of the transmitted light

```
// Scilab code Ex6.6: Pg:249 (2008)
clc;clear;
I_m = 1;     // For simplicity assume maximum intensity to be unity, unit
I = I_m/3;     // Final intensity, unit
for theta = 30:15:60
I = I_m*cosd(theta)^2;     // Intensity of the emerging light
printf("\nThe fractional intensity of light transmitted for theta = %2d degree is %3.2f ", theta, I/I_m);
end
```

```
// Result
// The fractional intensity of light transmitted for
     theta = 30 degree is 0.75
// The fractional intensity of light transmitted for
     theta = 45 degree is 0.50
// The fractional intensity of light transmitted for
     theta = 60 degree is 0.25
```

Scilab code Exa 6.7 Intensity ratio of two emerging beams

```
// Scilab code Ex6.7: Pg:249 (2008)
clc;clear;
I_0 = 1;     // For simplicity assume maximum
    intensity to be unity, unit
theta_A = 60;     // Angle between the plane of
    polarizer and plane of the analyzer for beam A,
    degree
theta_B = 30;     // Angle between the plane of
    polarizer and plane of the analyzer for beam B,
    degree
I_A = I_0*cosd(theta_A)^2;     // Malus' Law for beam
     A
I_B = I_0*cosd(theta_B)^2;     // Malus' Law for beam
     A
printf("\nThe intensity ratio of two emerging beams
    = %4.2f ", I_A/I_B);

// Result
// The intensity ratio of two emerging beams = 0.33
```

Scilab code Exa 6.8 Polarizing angle and the angle of refraction for light incident on water

```
// Scilab code Ex6.8: Pg:250 (2008)
clc;clear;
// Define function to convert degrees to degree and
   minute
function [deg, minute] = deg2degmin(theta)
    deg = floor(theta);
    minute = ceil((theta-deg)*60);
endfunction
C = 48;    // Critical angle of incidence, degree
mu = 1/sind(C);    // Index of refraction
// From Brewester's law mu = tan i_p, solving for
   i_p
i_p = atand(mu);    // Polarizing angle, degree
// Since i_p + r = %pi/2, solving for r
r = 90 - i_p;    // Angle of refraction, degree
[d1, m1] = deg2degmin(i_p);
[d2, m2] = deg2degmin(r);
printf("\nThe polarizing angle = %2d degree %2d
   minute", d1, m1);
printf("\nThe angle of refraction = %2d degree %2d
   minute", d2, m2);

// Result
// The polarizing angle = 53 degree 23 minute
// The angle of refraction = 36 degree 38 minute
```

Scilab code Exa 6.9 Thickness of a quarter wave plate for a crystal

```
// Scilab code Ex6.9: Pg:261 (2008)
clc;clear;
mu_O = 1.55;    // Refractive index for an ordinary
   beam
mu_E = 1.54;    // Refractive index for an extra-
   ordinary beam
lambda = 5890e-08;    // Wavelength of light, cm
```

```
t = lambda/(4*(mu_O-mu_E));     // Thickness of quarter wave plate, cm
printf("\nThe thickness of a quarter wave plate for the crystal = %4.2e cm", t);

// Result
// The thickness of a quarter wave plate for the crystal = 1.47e-003 cm
```

Scilab code Exa 6.10 Thickness of a quarter wave plate of quartz

```
// Scilab code Ex6.10: Pg:261 (2008)
clc;clear;
mu_O = 1.55336;     // Refractive index for an ordinary beam
mu_E = 1.54425;     // Refractive index for an extra-ordinary beam
lambda = 5.893e-05;     // Wavelength of sodium light, cm
t = lambda/(4*(mu_O-mu_E));     // Thickness of quarter wave plate, cm
printf("\nThe thickness of the quarter wave plate for quartz = %4.2e cm", t);

// Result
// The thickness of the quarter wave plate for quartz = 1.62e-003 cm
```

Scilab code Exa 6.11 Phase retardation in quarter wave plate for given wavelength

```
// Scilab code Ex6.11: Pg:261 (2008)
clc;clear;
```

```
mu_O = 1.55336;      // Refractive index for an
   ordinary beam
mu_E = 1.54425;      // Refractive index for an extra-
   ordinary beam
lambda_O = 5.893e-05;      // Wavelength of ordinary
   light , cm
lambda = 4.358e-005;      // Given wavelength of light
   , cm
PR = 2*%pi/lambda*lambda_O/4;      // The phase
   retardation in quarter wave plate for given
   wavelength
printf("\nThe phase retardation in quarter wave
   plate for given wavelength = %4.2f pi-radian", PR
   /%pi);

// Result
// The phase retardation in quarter wave plate for
   given wavelength = 0.68 pi-radian
```

Scilab code Exa 6.12 Difference in the refractive indices of two rays

```
// Scilab code Ex6.12: Pg:262 (2008)
clc;clear;
t = 0.003;      // Thickness of the crystal slice , cm
Lambda = 6e-005;      // Wavelength of linearly
   polarized light , cm
d_mu = Lambda/(4*t);      // Difference in the
   refractive indices of two rays
printf("\nThe difference in the refractive indices
   of two rays = %1.0e ", d_mu );

// Result
// The difference in the refractive indices of two
   rays = 5e-003
```

Scilab code Exa 6.13 Thickness of the doubly refracting crystal

```
// Scilab code Ex6.13: Pg:262 (2008)
clc;clear;
mu_O = 1.65;     // Refractive index for an ordinary
    beam
mu_E = 1.48;     // Refractive index for an extra-
    ordinary beam
lambda = 6000e-08;     // Wavelength of light, cm
t = lambda/(2*(mu_O - mu_E));     // Thickness of
    doubly refracting crystal, cm
printf("\nThe thickness of the doubly refracting
    crystal = %4.2e cm", t);

// Result
// The thickness of the doubly refracting crystal =
    1.76e-004 cm
```

Scilab code Exa 6.14 Thinnest possible quartz plate

```
// Scilab code Ex6.14: Pg:262 (2008)
clc;clear;
mu_O = 1.544;     // Refractive index for an ordinary
     beam
mu_E = 1.553;     // Refractive index for an extra-
    ordinary beam
lambda = 6000e-08;     // Wavelength of light, cm
t = lambda/(2*(mu_E - mu_O));     // Thickness of
    doubly refracting crystal, cm
printf("\nThe thinnest possible quartz = %4.2e cm",
    t);
```

```
printf("\nThe thicknesses which would give the same result are %4.2e cm, %4.2e cm, %4.2e cm,...", t, 3*t, 5*t);

// Result
// The thinnest possible quartz = 3.33e-003 cm
// The thicknesses which would give the same result are 3.33e-003 cm, 1.00e-002 cm, 1.67e-002 cm,...
```

Scilab code Exa 6.15 Wavelength for a quarter and a half wave plate in the visible region

```
// Scilab code Ex6.15: Pg:263 (2008)
clc;clear;
mu_O = 1.5443;     // Refractive index for an ordinary beam
mu_E = 1.5533;     // Refractive index for an extra-ordinary beam
t = 0.01436;     // Thickness of the quartz plate, cm
lambda = zeros(6);     // Initialize lambda
// As t = (2*n + 1)*lambda/(4*(mu_O - mu_E)) for quarter wave plate, solving for lambda
printf("\nFor quarter wave in visible region the wavelengths are:\n");
for n = 1:1:6
lambda(n) = 4*(mu_E - mu_O)*t/(2*(n-1) + 1)*1e+008;     // Wavelength for a quarter wave plate, cm
if lambda(n) >= 3500 & lambda(n) <= 8000 then
    printf("%d ansgtrom; ", ceil(lambda(n)));
end
end // for loop
// As t = (2*n + 1)*lambda/(2*(mu_O - mu_E)) for half wave plate, solving for lambda
printf("\n\nFor half wave in visible region the wavelengths are:\n");
```

```
for n = 1:1:6
lambda(n) = 2*(mu_E - mu_O)*t/(2*(n-1) + 1)*1e+008;    // Wavelength for a half wave plate, cm
if lambda(n) >= 3500 & lambda(n) <= 8000 then
    printf("%d ansgtrom; ", ceil(lambda(n)));
end
end    // for loop

// Result
// For quarter wave in visible region the wavelengths are:
// 7386 ansgtrom; 5744 ansgtrom; 4700 ansgtrom;

// For half wave in visible region the wavelengths are:
// 5170 ansgtrom; 3693 ansgtrom;
```

Chapter 7

Nuclear Structure and Nuclear Forces

Scilab code Exa 7.1 Binding energy of an alpha particle

```
// Scilab code Ex7.1: Pg:275 (2008)
clc;clear;
M_He = 4.001265;     // Mass of helium nucleus, amu
M_P = 1.007277;     // Mass of proton, amu
M_N = 1.008666;     // Mass of neutron, amu
amu = 931.4812;     // One amu
M = 2*M_P+2*M_N;     // Total initial mass of two
    protons and two neutrons, amu
delta_m = M-M_He;     // Mass defect, amu
BE = delta_m * amu;     // Binding energy of alpha
    particle, MeV
printf("\nThe binding energy of an alpha particle =
    %7.4f Mev", BE);
printf("\nThe binding energy per nucleon = %8.6f Mev
    ", BE/4);

// Result
// The binding energy of an alpha particle = 28.5229
    Mev
```

```
// The binding energy per nucleon = 7.130721 Mev
```

Scilab code Exa 7.2 Energy in joule and electrical energy in kilowatt hours in a thermonuclear reaction

```
// Scilab code Ex7.2: Pg:275 (2008)
clc;clear;
M_H = 1e-03;     // Mass of hydrogen, kg
M_He = 0.993e-03;     // Mass of helium, kg
delta_m = M_H-M_He;     // Mass defect, amu
c = 3e+08;     // Velocity of light, m/s
E = delta_m*c^2;     // Energy released, joules
EL = (5/100)*E/36e+05;     // Electrical energy, kilowatt hour
printf("\nThe energy released in joule in a thermonuclear reaction = %4.1e joule", E);
printf("\nThe electrical energy in kilowatt hours in a thermonuclear reaction = %4.2e kilowatt hour", EL);

// Result
// The energy released in joule in a thermonuclear reaction = 6.3e+011 joule
// The electrical energy in kilowatt hours in a thermonuclear reaction = 8.75e+003 kilowatt hour
```

Scilab code Exa 7.3 Energy produced when a neutron breaks into a proton and electron

```
// Scilab code Ex7.3: Pg:276 (2008)
clc;clear;
M_n = 1.6747e-027;     // Mass of neutron, kg
M_p = 1.6725e-027;     // Mass of proton, kg
```

```
M_e = 9e-031;     // Mass of electron, kg
c = 3e+08;     // Velocity of light, m/s
delta_m = M_n-(M_p + M_e);     // Mass defect, kg
E = delta_m*c^2/1.6e-013;     // Energy released, MeV
printf("\nThe energy produced when a neutron breaks into a proton and an electron = %4.2f MeV", E);

// Result
// The energy produced when a neutron breaks into a proton and an electron = 0.73 MeV
```

Scilab code Exa 7.4 Magnetic field to accelerate protons

```
// Scilab code Ex7.4: Pg:288 (2008)
clc;clear;
f0 = 8e+06;     // Cyclotron frequency, c/s
c = 3e+010;     // Speed of light, cm/s
m = 1.67e-024;     // Mass of proton, gm
q = 4.8e-010/c;     // Charge on a proton, esu
// Since the cyclotron frequency is given by fo = q*B/2*%pi*m. On solving it for B, we have
B = 2*%pi*m*f0/q;     // Magnetic field, Weber per meter square
printf("\nThe magnetic field to accelerate protons = %5.3f Wb per Sq. m", B/1e+04);

// Result
// The magnetic field to accelerate protons = 0.525 Wb per Sq. m
```

Scilab code Exa 7.5 Velocity and energy of deutron

```
// Scilab code Ex7.5: Pg:288 (2008)
```

```
clc;clear;
m = 3.34e-027;     // Mass of deutron , gm
q = 1.6e-019;     // Charge , coulomb
r = 0.2;     // Radius of the path of deutron , meter
B = 1.5;     // Magnetic field , weber per meter square
v = q*B*r/m;     // velocity of the deutron , m/s
E = 1/2*m*v^2/1.6e-013;     // Energy of the deutron , MeV
printf("\nThe velocity of deutron = %5.3e m/s ", v);
printf("\nThe energy of deutron = %5.3f MeV ", E);

// Result
// The velocity of deutron = 1.437e+007 m/s
// The energy of deutron = 2.156 MeV
```

Scilab code Exa 7.6 Energy of an electron undergoing revolutions in a betatron

```
// Scilab code Ex7.6: Pg:293 (2008)
clc;clear;
dE = 15/1e+006;     // Increase in energy per revolution , MeV
n = 1e+006;     // Number of revolutions
E = dE*n;     // Final energy of an electron after 10e+06 revolutions , MeV
printf("\nThe energy of an electron undergoing revolutions = %2.0f MeV ", E);

// Result
// The energy of an electron undergoing revolutions = 15 MeV
```

Scilab code Exa 7.7 Final energy and average energy gained per revolution by electron

```
// Scilab code Ex7.7: Pg:294 (2008)
clc;clear;
c = 3e+08;     // Velocity of light , m/s
e = 1.6e-019;     // Charge of an electron , coulomb
B = 0.5;     // Maximum magnetic field at the
    electron orbit , Weber per meter square
R = 0.75;     // Radius of the orbit , meter
omega = 50;     // frequency of alternating current
    through electromagnetic coils , Hz
N = c/(4*2*%pi*omega*R);     // Number of revolutions
E = B*e*R*c/(e*1e+006);     // Final energy of the
    electrons , MeV
E_av = E*1e+06/N;     // Average energy per
    revolution , eV
printf("\nThe final energy of electron = %5.1f MeV "
    , E);
printf("\nThe average energy of electron = %3.0f eV
    ", E_av);

// Result
// The final energy of electron = 112.5 MeV
// The average energy of electron = 353 eV
// The answer is wrong in the textbook
```

Scilab code Exa 7.8 Energy per revolution of an electron

```
// Scilab code Ex7.8: Pg:295 (2008)
clc;clear;
c = 3e+08;     // Velocity of light , m/s
e = 1.6e-019;     // Charge of an electron , coulomb
B = 0.5;     // Maximum magnetic field at the
    electron orbit , Weber per meter square
```

```
D = 1.5;     // Diameter of the orbit, meter
R = D/2;     // Radius of the orbit, meter
omega = 50;     // frequency of alternating current through electromagnetic coils, Hz
N = c/(4*2*%pi*omega*R);     // Number of revolutions
E = B*e*R*c/1.6e-013;     // Final energy of the electrons, MeV
E_av = (E*1e+06)/N;     // Average energy per revolution, eV
printf("\nThe energy per revolution of the electron = %4.1f eV ", E);
printf("\nThe average energy of electron = %3.0f eV ", E_av);

// Result
// The energy per revolution of the electron = 112.5 eV
// The average energy of electron = 353 eV
// The answer is given wrong in the textbook
```

Scilab code Exa 7.9 Thermal neutrons capture

```
// Scilab code Ex7.9: Pg:298 (2008)
clc;clear;
sigma = 2e+04*1e-028;     // Nuclear reaction cross-section, Sq.m
x = 1e-04;     // Thickness of the sheet, meter
m = 112;     // Mean atomic mass of cadmium, amu
rho = 8.64e+03;     // Density of cadmium sheet, kg per cubic meter
amu = 1.66e-027;     // Mass equivalent of 1 amu, kg
// Since cadmium 113 contains 12 percent of natural cadmium. Thus
n = 12/100*rho/(m*amu);     // Number of nuclei per unit volume, atoms per cubic meter
```

```
n_sigma = n*sigma;    // Microscopic cross-section, per length
// As N = N0*exp(-n*sigma*x), so that (N - N0)/N0 = 1-exp(-n_sigma*x)
frac_N = 1-exp(-n_sigma*x);
N0 = 1;    // For simlicity assume number of incident neutrons be unity
N = 1/100*N0;    // Given number of neutrons which pass through cadmium sheet
x = -log(N/N0)/n_sigma*1e+003;    // Thickness of the cadmium sheet when one percent of the incident neutrons pass through the cadmium sheet, mm
printf("\nThe fraction of the incident thermal neutrons absorbed by the cadmium sheet = %4.2f ", frac_N);
printf("\nThe thickness of the cadmium sheet when one percent of the incident neutrons pass through the cadmium sheet = %4.2f mm", x);

// Result
// The fraction of the incident thermal neutrons absorbed by the cadmium sheet = 0.67
// The thickness of the cadmium sheet when one percent of the incident neutrons pass through the cadmium sheet = 0.41 mm
```

Scilab code Exa 7.10 Total energy in fission of uranium reaction in MeV and kilowatt hours

```
// Scilab code Ex7.10: Pg:306 (2008)
clc;clear;
m_u = 235.0439;    // Mass of uranium, amu
m_n = 1.0087;    // Mass of neutron, amu
m_Ba = 140.9139;    // Mass of Barium, amu
```

```
m_Kr = 91.8937;      // Mass of Krypton, amu
M_1 = m_u + m_n;     // Sum of masses before reaction, amu
M_2 = m_Ba + m_Kr + 3*m_n;     // Sum of masses after reaction, amu
delta_m = M_1 -M_2;     // Mass lost in the fission, amu
// Since the number of atoms in 235 g of Uranium is 6.02e+023
N = 6.02e+023/235;     // Number of atoms in one gm of U-235
// Since energy equivalent of 1 amu is 931.5MeV
E_MeV = delta_m*N*931.5;     // Energy released in fission of Uranium 235, MeV
printf("\nTotal energy in fission of uranium reaction in MeV = %4.2e MeV ", E_MeV);
E_kWh = E_MeV*1.6e-013/3.6e+06;     // Energy released in fission of Uranium 235, kWh
printf("\nTotal energy in fission of uranium reaction in kiloWatt hour = %4.2e kWh", E_kWh);

// Result
// Total energy in fission of uranium reaction in MeV = 5.22e+023 MeV
// Total energy in fission of uranium reaction in kiloWatt hour = 2.32e+004 kWh
```

Scilab code Exa 7.11 Uranium undergoing fission in a nuclear reactor

```
// Scilab code Ex7.11: Pg:307 (2008)
clc;clear;
P = 3.2e+07/1.6e-013;     // Power developed by the reactor, MeV
E = 200;     // Energy released by the reactor per fission, MeV
```

```
n = P/E;    // Number of fissions occuring in the reactor per second, per sec
N = n*1000*3600;    // Number of atoms or nuclei of Uranium 235 consumed in 1000 hours
// Since the number of atoms in 235 g of Uranium is 6e+023
M = N/6e+023*235/1000;    // Mass of Uranium 235 consumed in 1000 hours, kg
printf("\nThe number of atoms of Uranium 235 undergoing fission per second = %4.1e ", N);
printf("\nThe mass of Uranium 235 consumed in 1000 hours = %4.2f kg ", M);

// Result
// The number of atoms of Uranium 235 undergoing fission per second = 3.6e+024
// The mass of Uranium 235 consumed in 1000 hours = 1.41 kg
```

Scilab code Exa 7.12 Energy liberated by the fission of one kg of substance

```
// Scilab code Ex7.12: Pg:307 (2008)
clc;clear;
c = 3e+08;    // Velocity of light, m/s
delta_m =0.1/100*1;    // Mass lost in one kg of substance, kg
delta_E = delta_m*c^2;    // Energy liberated by the fission of one kg of substance, joule
// Since 1kWh = 1000 watt*3600 sec = 3.6e+06 joule
delta_E = delta_m*c^2/3.6e+06;    // Energy liberated by the fission of one kg of substance, kWh
printf("\nThe energy liberated by the fission of one kg of substance = %3.2e kWh", delta_E);
```

```

// Result
// The energy liberated by the fission of one kg of
   substance = 2.50e+007 kWh
```

Scilab code Exa 7.13 Total energy released in the fission of uranium 235

```
// Scilab code Ex7.13: Pg:308 (2008)
clc;clear;
P = 2/1.6e-013;     // Power to be produced, MeV/sec
E_bar = 200;    // Energy released per fission, MeV
n = P/E_bar;    // Required number of fissions per
   second
// Since the number of atoms in 235gm of Uranium is
   6.02e+023
N = (6.02e+023/235)*500;    // Number of atoms in
   500 gm of U-235
E = E_bar*N;    // Total energy released in the
   complete fission of 500gm of uranium 235, MeV
printf("\nThe total energy released in the complete
   fission of 500gm of uranium 235 = %4.2e MeV", E);

// Result
// The total energy released in the complete fission
    of 500gm of uranium 235 = 2.56e+026 MeV
```

Scilab code Exa 7.14 Energy source in stars

```
// Scilab code Ex7.14: Pg:309 (2008)
clc;clear;
amu = 931.5;    // Energy equivalent of 1 amu, MeV
M_He = 4.00260;     // Mass of helium, amu
m_e = 0.00055;     // Mass of electron, amu
```

```
M_C = 12.000;    // Mass of carbon, amu
m_He = M_He - 2*m_e;    // Mass of helium nucleus,
    amu
m_C = M_C - 6*m_e;    // Mass of carbon nucleus, amu
d_m = 3*m_He - m_C;    // Mass defect, amu
E = d_m*amu;    // Equivalent energy of mass defect,
    MeV
printf("\nThe energy invloved in each fusion
    reaction inside the star = %4.2f MeV", E);

// Result
// The energy invloved in each fusion reaction
    inside the star = 7.27 MeV
```

Scilab code Exa 7.15 Average current in the Geiger Muller circuit

```
// Scilab code Ex7.15: Pg:311 (2008)
clc;clear;
r = 500;    // Counting rate of Geiger-Muller
    counter, counts/minute
n = r*1e+08;    // Number of electrons collected per
    minute
q = n*1.6e-019;    // Charge per minute, coulomb per
    minute
I = q/60;    // Charge per second, coulomb per
    second
printf("\nThe average current in the Geiger-Muller
    counter circuit = %4.2e ampere ", I);

// Result
// The average current in the Geiger-Muller counter
    circuit = 1.33e-010 ampere
```

Scilab code Exa 7.16 Mass of the particle in an Aston mass spectrograph

```
// Scilab code Ex7.16: Pg 315 (2008)
clc;clear;
m1 = 12;     // Mass of first trace, unit
m2 = 16;     // Mass of second trace, unit
d = 4.8;     // Distance between the traces, cm
D = [8.4, -8.4];     // Distance of the mark from the trace of mass 16
x = poly(0, 'x');
x = roots(m1*x-m2*(x-d));     // The distance of the mark from the trace of mass 16
M = m2*(x+D)/x;     // Mass of the particle whose trace is at a distance of 8.4 cm from the trace of mass 16
printf("\nThe mass of the particle whose trace is at a distance of 8.4 cm from the trace of mass 16 = %d or %d", M(1), M(2));

// Result
// The mass of the particle whose trace is at a distance of 8.4 cm from the trace of mass 16 = 23 or 9
```

Chapter 8

Number Systems Used in Digital Electronics

Scilab code Exa 8.1 Conversion of binary number to decimal number

```
// Scilab code Ex8.1 : Pg:327(2008)
clc;clear;
function [dec]= binary_decimal(n) // Function to convert binary to decimal
    dec = 0;
    i = 0;
    while (n <> 0)
      rem = n-fix(n./10).*10;
      n = int(n/10);
      dec = dec + rem*2.^i;
      i = i + 1;
    end
endfunction

num = 11001;     // Initialize the binary number
printf("%d in binary = %d in decimal", num, binary_decimal(num));

// Result
```

```
//  11001 in binary = 25 in decimal
```

Scilab code Exa 8.2 Conversion of binary fraction to its decimal equivalent

```
// Scilab code Ex8.2 : Pg:328(2008)
clc;clear;
function [dec]= binfrac_decifrac(n) // Function to
    convert binary fraction to decimal fraction
    dec = 0;
    i = -1;
    while (i >= -3)
      n = n*10;
      rem = round(n);
      n = n-rem;
      dec = dec + rem*2.^i;
      i = i - 1;
    end
endfunction

n = 0.101;     // Initialize the binary number
printf("Binary fraction %5.3f = Decimal frac = %5.3f
    ", n, binfrac_decifrac(n));

// Result
//  Binary fraction 0.101 = Decimal frac = 0.625
```

Scilab code Exa 8.3 Decimal equivalent of 6 bit binary number

```
// Scilab code Ex8.3 : Pg:328(2008)
clc;clear;
function [deci]= binary_decimal(ni) // Function to
    convert binary to decimal
```

```
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

function [decf]= binfrac_decifrac(nf) // Function to convert binary fraction to decimal fraction
    decf = 0;
    i = -1;
    while (i >= -3)
      nf = nf*10;
      rem = round(nf);
      nf = nf-rem;
      decf = decf + rem*2.^i;
      i = i - 1;
    end
endfunction

n = 101.101;     // Initialize the binary number
n_int = int(n);      // Extract the integral part
n_frac = n-n_int;    // Extract the fractional part
printf("Decimal equivalent of %7.3f = %5.3f", n, binary_decimal(n_int)+binfrac_decifrac(n_frac));

// Result
// Decimal equivalent of 101.101 = 5.625
```

Scilab code Exa 8.4 Binary equivalent of decimal number

```
// Scilab code Ex8.4 : Pg:330(2008)
```

```
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [binf]= decifrac_binfrac(nf) // Function to convert binary fraction to decimal fraction
    binf = 0; i = 0.1;
    while (nf <> 0)
      nf = nf*2;
      rem = int(nf);
      nf = nf-rem;
      binf = binf + rem*i;
      i = i/10;
    end
endfunction

n = 25.625;     // Initialize the decimal number
n_int = int(n);      // Extract the integral part
n_frac = n-n_int;    // Extract the fractional part
printf("Binary equivalent of %6.3f = %9.3f", n, decimal_binary(n_int)+decifrac_binfrac(n_frac));

// Result
// Binary equivalent of 25.625 = 11001.101
```

Scilab code Exa 8.5 Addition of two binary numbers

```
// Scilab code Ex8.5 : Pg:332(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to
    convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to
    convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

num1 = 110011;    // Initialize the first binary
    number
num2 = 101101;    // Initialize the second binary
    number

printf("%6d + %6d = %7d", num1, num2, decimal_binary
    (binary_decimal(num1)+binary_decimal(num2)));

// Result
// 110011 + 101101 = 1100000
```

Scilab code Exa 8.6 Subtraction of two binary number

```
// Scilab code Ex8.6 : Pg:333(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

sub = 1110;    // Initialize the first binary number
men = 0101;    // Initialize the second binary number

printf("%4d - 0%3d = %4d", sub, men, decimal_binary(binary_decimal(sub)-binary_decimal(men)));
```

```

// Result
// 1110 - 0101 = 1001
```

Scilab code Exa 8.7 Binary Subtraction

```
// Scilab code Ex8.7 : Pg:333(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

sub = 1000;     // Initialize the first binary number
men = 0001;     // Initialize the second binary number
```

```
printf("%4d - 000%1d = 0%3d", sub, men, decimal_binary(binary_decimal(sub)-binary_decimal(men)));

// Result
// 1000 - 0001 = 0111
```

Scilab code Exa 8.8 Binary subtraction of two numbers

```
// Scilab code Ex8.8 : Pg:334(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction
```

```

sub = 1001;     // Initialize the first binary number
men = 0111;     // Initialize the second binary number

printf("%4d - 0%1d = 00%2d", sub, men, decimal_binary(binary_decimal(sub)-binary_decimal(men)));

// Result
// 1001 - 0111 = 0010
```

Scilab code Exa 8.9 Five digit binary subtraction

```
// Scilab code Ex8.9 : Pg:334(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
```

```
        deci = deci + rem*2.^i;
        i = i + 1;
      end
endfunction

sub = 10110;     // Initialize the first binary number
men = 01011;     // Initialize the second binary number

printf("%5d - 0%4d = 0%4d", sub, men, decimal_binary(binary_decimal(sub)-binary_decimal(men)));

// Result
// 10110 - 01011 = 01011
```

Scilab code Exa 8.10 Ones complement method to subtract two binary numbers

```
// Scilab code Ex8.10 : Pg:335(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
```

```
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

// Function to convert a vector with binary elements to a binary number
function vtob = vector_to_bin(vector)
    cnt = 1; vtob = 0;
for i = 1:1:length(vector)
    vtob = vtob + vector(i)*cnt;
    cnt = cnt*10;
end
endfunction

function bin_cmp = ones_cmp(bin)       // Function to perform ones complement
    binc = zeros(5);
    i = 1;
    while(i <= 5)
        rem = bin-fix(bin./10).*10;
        if rem == 1 then
            rem = 0;
        else
           rem = 1;
        end
         bin = int(bin/10);
         binc(i)=rem;
         i = i+1;
    end
bin_cmp = vector_to_bin(binc);
endfunction

```

```
function plus_one_res = twos_cmp(r)     // Function to perform twos complement
    onec = zeros(5);
    i = 1;
    while(i <= 5)
        rem = r-fix(r./10).*10;
        r = int(r/10);
        onec(i)=rem;
        i = i+1;
    end
plus_one_res = vector_to_bin(onec);
    plus_one_res = binary_decimal(plus_one_res)+1;
endfunction

function fr = check_result(res)     // Function to check the occurence of end-around carry
    max_result = 11111;
    if binary_decimal(res) > binary_decimal(max_result) then
        fr = decimal_binary(twos_cmp(res));
    else
        fr = ones_cmp(res);
    end
endfunction

sub = 11011;    // Initialize the first binary number
men = 01101;    // Initialize the second binary number
result = decimal_binary(binary_decimal(sub)+binary_decimal(ones_cmp(men)));
final_result = check_result(result);
printf("%5d - 0%4d = 0%4d", sub, men, final_result);

// Result
//  11011 - 01101 = 01110
```

Scilab code Exa 8.11 Binary subtraction using ones complement method

```
// Scilab code Ex8.11 : Pg:336(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

// Function to convert a vector with binary elements to a binary number
function vtob = vector_to_bin(vector)
    cnt = 1; vtob = 0;
for i = 1:1:length(vector)
    vtob = vtob + vector(i)*cnt;
```

```
    cnt = cnt*10;
end
endfunction

function bin_cmp = ones_cmp(bin)       // Function to perform ones complement
    binc = zeros(5);
    i = 1;
    while(i <= 5)
        rem = bin-fix(bin./10).*10;
        if rem == 1 then
            rem = 0;
        else
           rem = 1;
        end
         bin = int(bin/10);
         binc(i)=rem;
         i = i+1;
    end
bin_cmp = vector_to_bin(binc);
endfunction

function plus_one_res = twos_cmp(r)     // Function to perform twos complement
    onec = zeros(5);
    i = 1;
    while(i <= 5)
        rem = r-fix(r./10).*10;
        r = int(r/10);
        onec(i)=rem;
        i = i+1;
    end
plus_one_res = vector_to_bin(binc);
    plus_one_res = binary_decimal(plus_one_res)+1;
endfunction

function fr = check_result(res)     // Function to check the occurence of end-around carry
```

```
    max_result = 11111;
    if binary_decimal(res) > binary_decimal(max_result) then
        fr = decimal_binary(twos_cmp(res));
    else
        fr = ones_cmp(res);
    end
endfunction

sub = 01101;    // Initialize the first binary number
men = 11011;    // Initialize the second binary number
result = decimal_binary(binary_decimal(sub)+binary_decimal(ones_cmp(men)));
final_result = check_result(result);
printf("0%4d - %5d = -0%4d", sub, men, final_result);

// Result
//  01101 - 11011 = -01110
```

Scilab code Exa 8.12 Binary subtraction using twos complement method

```
// Scilab code Ex8.12 : Pg:336(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
```

```
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

// Function to convert a vector with binary elements to a binary number
function vtob = vector_to_bin(vector)
    cnt = 1; vtob = 0;
for i = 1:1:length(vector)
    vtob = vtob + vector(i)*cnt;
    cnt = cnt*10;
end
endfunction

function bcmp_plus_one = twos_cmp(bin)        // Function to perform twos complement
    binc = zeros(4);
    i = 1;
    while(i <= 4)
        rem = bin-fix(bin./10).*10;
        if rem == 1 then
            rem = 0;
        else
           rem = 1;
        end
         bin = int(bin/10);
         binc(i)=rem;
```

```
            i = i+1;
        end
bcmp_plus_one = vector_to_bin(binc);
        bcmp_plus_one = binary_decimal(bcmp_plus_one)+1;
endfunction

function fr = refine_result(res)      // Function to refine the resut
        binc = zeros(4);
        i = 1;
        while(i <= 4)
            rem = res-fix(res./10).*10;
            res = int(res/10);
            binc(i)=rem;
            i = i+1;
        end
fr = vector_to_bin(binc);
endfunction

sub = 1101;     // Initialize the first binary number
men = 1010;     // Initialize the second binary number
result = decimal_binary(binary_decimal(sub)+binary_decimal(twos_cmp(men)));
final_result = refine_result(result);
printf("%4d - %4d = 00%2d", sub, men, final_result);

// Result
//    1101 - 1010 = 0011
```

Scilab code Exa 8.13 Twos complement method of binary subtraction

```
// Scilab code Ex8.13 : Pg:336(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to
```

```
convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

// Function to convert a vector with binary elements to a binary number
function vtob = vector_to_bin(vector)
    cnt = 1; vtob = 0;
for i = 1:1:length(vector)
    vtob = vtob + vector(i)*cnt;
    cnt = cnt*10;
end
endfunction

function bin_cmp = ones_cmp(bin)        // Function to perform ones complement
    binc = zeros(4);
    i = 1;
    while(i <= 4)
```

```
        rem = bin-fix(bin./10).*10;
        if rem == 1 then
            rem = 0;
        else
           rem = 1;
        end
         bin = int(bin/10);
         binc(i)=rem;
         i = i+1;
    end
bin_cmp = vector_to_bin(binc);
endfunction

function bcmp_plus_one = twos_cmp(bin)        // Function to perform twos complement
    binc = zeros(4);
    i = 1;
    while(i <= 4)
        rem = bin-fix(bin./10).*10;
        if rem == 1 then
            rem = 0;
        else
           rem = 1;
        end
         bin = int(bin/10);
         binc(i)=rem;
         i = i+1;
    end
bcmp_plus_one = vector_to_bin(binc);
    bcmp_plus_one = binary_decimal(bcmp_plus_one)+1;
endfunction

function fr = check_result(res)      // Function to check the occurence of end-around carry
    max_result = 11111;
    if binary_decimal(res) < binary_decimal(max_result) then
        fr = ones_cmp(res-1);
```

```
    else
        fr = res;
    end
endfunction

sub = 1010;     // Initialize the first binary number
men = 1101;     // Initialize the second binary number
result = decimal_binary(binary_decimal(sub)+binary_decimal(twos_cmp(men)));
final_result = check_result(result);
printf("%4d - %4d = -00%2d", sub, men, final_result);

// Result
//  1010 - 1101 = -0011
```

Scilab code Exa 8.14 Binary multiplication of two numbers

```
// Scilab code Ex8.14 : Pg:337(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
```

```
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

function binp = bin_product(op1, op2)
    binp = decimal_binary(binary_decimal(op1)*binary_decimal(op2));
endfunction

mul1 = 111;    // Initialize the first binary multiplicand
mul2 = 101;    // Initialize the second binary multiplicand
product = bin_product(mul1, mul2);

printf("%3d X %3d = %6d", mul1, mul2, product);

// Result
//  111 X 101 = 100011
```

Scilab code Exa 8.15 Multiplication of two binary numbers

```
// Scilab code Ex8.15 : Pg:337(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
```

```
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

function binp = bin_product(op1, op2)
    binp = decimal_binary(binary_decimal(op1)*binary_decimal(op2));
endfunction

mul1 = 1101;    // Initialize the first binary multiplicand
mul2 = 1100;    // Initialize the second binary multiplicand
product = bin_product(mul1, mul2);

printf("%4d X %4d = %8d", mul1, mul2, product);

// Result
//  1101 X 1100 = 10011100
```

Scilab code Exa 8.16 Product of two binary numbers

```
// Scilab code Ex8.16 : Pg:337(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

function binp = bin_product(op1, op2)
    binp = decimal_binary(binary_decimal(op1)*binary_decimal(op2));
endfunction

mul1 = 1111 ;     // Initialize the first binary multiplicand
mul2 = 0111;     // Initialize the second binary multiplicand
product = bin_product(mul1, mul2);
```

```
printf("%4d X 0%3d = %7d", mul1, mul2, product);

// Result
//    1111 X 0111 = 1101001
```

Scilab code Exa 8.17 Binary division of two numbers

```
// Scilab code Ex8.17 : Pg:338(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

function binp = bin_division(op1, op2)
```

```
    binp = decimal_binary(binary_decimal(op1)/binary_decimal(op2));
endfunction

dividend = 11001 ;     // Initialize the first binary multiplicand
divisor = 101;     // Initialize the second binary multiplicand
product = bin_division(dividend, divisor);

printf("%5d divided by %3d gives %3d", dividend, divisor, product);

// Result
// 11001 divided by 101 gives 101
```

Scilab code Exa 8.18 Division of two binary numbers

```
// Scilab code Ex8.18 : Pg:339(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
    bini = 0;
    i = 1;
    while (ni <> 0)
      rem = ni-fix(ni./2).*2;
      ni = int(ni/2);
      bini = bini + rem*i;
      i = i * 10;
    end
endfunction

function [binf]= decifrac_binfrac(nf) // Function to convert binary fraction to decimal fraction
    binf = 0; i = 0.1;
```

```
    while (nf <> 0)
      nf = nf*2;
      rem = int(nf);
      nf = nf-rem;
      binf = binf + rem*i;
      i = i/10;
    end
endfunction

function [deci]= binary_decimal(ni) // Function to convert binary to decimal
    deci = 0;
    i = 0;
    while (ni <> 0)
      rem = ni-fix(ni./10).*10;
      ni = int(ni/10);
      deci = deci + rem*2.^i;
      i = i + 1;
    end
endfunction

function binp = bin_division(op1, op2)
int_Q = int(binary_decimal(op1)/binary_decimal(op2));
frac_Q = binary_decimal(op1)/binary_decimal(op2) - int_Q;
    binp = decimal_binary(int_Q)+decifrac_binfrac(frac_Q);
endfunction

dividend = 11011 ;    // Initialize the first binary multiplicand
divisor = 100;    // Initialize the second binary multiplicand

product = bin_division(dividend, divisor);

printf("%5d divided by %3d gives %6.2f", dividend,
```

```
divisor, product);

// Result
//  11011 divided by 100 gives 110.11
```

Scilab code Exa 8.19 Conversion between number systems

```
// Scilab code Ex8.19 : Pg:346(2008)
clc;clear;
function [bini]= decimal_binary(ni) // Function to convert decimal to binary
   bini = 0;
   i = 1;
   while (ni <> 0)
     rem = ni-fix(ni./2).*2;
     ni = int(ni/2);
     bini = bini + rem*i;
     i = i * 10;
   end
endfunction

function octal = decimal_octal(n)  // Function to convert decimal to octal
   i=1; octal = 0;
   while (n<>0)
       rem = n-fix(n./8).*8;
       octal = octal + rem*i;
       n = int(n/8);
       i = i*10;
   end
endfunction

function hex = decimal_hex(n) // Function to convert decimal to hexadecimal
   hex = emptystr();
```

```
    while (n <>0)
        rem = n-fix(n./16).*16;
        if rem == 10 then
            hex(i)=hex+'A';
        elseif rem == 11 then
            hex=hex+'B';
        elseif rem == 12 then
            hex=hex+'C';
        elseif rem == 13 then
            hex=hex+'D';
        elseif rem == 14 then
            hex=hex+'E';
        elseif rem == 15 then
            hex=hex+'F';
        else
            hex=hex+string(rem);
        end
        n = int(n/16);
    end
    hex = strrev(hex);   // Reverse string
endfunction

n = [32, 256, 51];     // Initialize a vector to the given decimals
printf("\n--------------------------------------------------");
printf("\nDecimal       Binary         Octal          Hexadecimal ");
printf("\n--------------------------------------------------");
for i = 1:1:3
printf("\n%d        %10d       %5d            %4s", n(i), decimal_binary(n(i)), decimal_octal(n(i)), decimal_hex(n(i)));
end
printf("\
```

```
   n---------------------------------------------------
   ");

// Result
//
   ---------------------------------------------------
// Decimal       Binary          Octal
   Hexadecimal
//
   ---------------------------------------------------
// 32            100000          40              20
// 256          100000000        400             100
// 51            110011          63              33
//
   ---------------------------------------------------
```

Scilab code Exa 8.20 Conversion of various number systems to decimal number system

```
// Scilab code Ex8.20 : Pg:346(2008)
clc;clear;
n1 = '11010', n2 = 'AB60', n3 = "777";
printf("\nThe %s of binary = %d of decimal", n1,
   bin2dec(n1)); // Convert from binary to decimal
printf("\nThe %s of hex = %d of decimal", n2,
   hex2dec(n2));  // Convert from hex to decimal
printf("\nThe %s of octal = %d of decimal", n3,
   oct2dec(n3));  // Convert from octal to decimal

// Result
// The 11010 of binary = 26 of decimal
// The AB60 of hex = 43872 of decimal
```

```
// The 777 of octal = 511 of decimal
```

Scilab code Exa 8.21 Octal and hexadecimal equivalent of groups of bytes

```
// Scilab code Ex8.21 : Pg:347(2008)
clc;clear;
bin = ['10001100', '00101110', '01011111', '01111011
    ', '00111010', '10010101', '10110110', '01011011'
    ];
printf("\n__________________________________");
printf("\nBinary        Octal         Hexadecimal");
printf("\n__________________________________");
for i=1:1:8
printf("\n%8s     %4s           %4s", bin(i), dec2oct(
    bin2dec(bin(i))), dec2hex(bin2dec(bin(i))));
end
printf("\n__________________________________");


// Result
// -----------------------------------
// Binary        Octal        Hexadecimal
// -----------------------------------
// 10001100       214              8C
// 00101110        56              2E
// 01011111       137              5F
// 01111011       173              7B
// 00111010        72              3A
// 10010101       225              95
// 10110110       266              B6
// 01011011       133              5B
// -----------------------------------
```

Chapter 10

Dielectrics

Scilab code Exa 10.1 Relative permittivity of sodium chloride

```
// Scilab code Ex10.1 : Pg:405 (2008)
clc;clear;
E = 1000;    // Electric field applied to sodium
   chloride crystal , V/m
P = 4.3e-008;    // Polarization , Coulomb per meter
   square
epsilon_0 = 8.85e-012;    // Permittivity of free
   space , force per meter
// Since P = epsilon_0*(epsilon_r -1)*E, solving for
   epsilon_r
epsilon_r = 1 + P/(epsilon_0*E);    // Relative
   permittivity of sodium chloride
printf("\nThe relative permittivity of sodium
   chloride = %4.2f ", epsilon_r);

// Result
// The relative permittivity of sodium chloride =
   5.86
```

Scilab code Exa 10.2 Electronic polarizability of an argon atom

```
// Scilab code Ex10.2: Pg:411 (2008)
clc;clear;
N = 2.7e+025;     // Number of molecules per unit volume
epsilon_r = 1.0024;     // Dielectric constant due to electronic polarization
epsilon_0 = 8.85e-012;     // Permittivity of free space, force per meter
// P = epsilon_0*(epsilon_r-1)*E and P = N*alpha_e*E, solving for alpha_e
alpha_e = epsilon_0*(epsilon_r-1)/N;     // Electronic polarizability of an argon atom, farad Sq.m
printf("\nThe electronic polarizability of an argon atom = %3.1e farad Sq.m", alpha_e);

// Result
// The electronic polarizability of an argon atom = 7.9e-040 farad Sq.m
```

Scilab code Exa 10.3 Polarizability and relative permittivity of one cubic meter of hydrogen gas

```
// Scilab code Ex10.3 : Pg:414 (2008)
clc;clear;
N = 9.8e+026;     // Number of atoms in one cubic meter of hydrogen gas
R = 0.53e-010;     // Radius of hydrogen atom, meter
epsilon_0 = 8.85e-012;     // Permittivity of free space, force per meter
alpha_e = 4*%pi*epsilon_0*R^3;     // Electronic polarizability of an argon atom, farad Sq.m
epsilon_r = 1 + 4*%pi*N*R^3;     // Relative
```

```
permittivity of one cubic meter of hydrogen gas
printf("\nThe polarizability of one cubic meter of hydrogen gas = %4.2e farad Sq.m", alpha_e);
printf("\nThe relative permittivity of one cubic meter of hydrogen gas = %6.4f", epsilon_r);

// Result
// The polarizability of one cubic meter of hydrogen gas = 1.66e-041 farad Sq.m
// The relative permittivity of one cubic meter of hydrogen gas = 1.0018
```

Scilab code Exa 10.4 Relative dielectric constant for sulphur

```
// Scilab code Ex10.4: Pg:417 (2008)
clc;clear;
alpha_e = 3.28e-040;    // Electronic polarizability of sulphur atom, Force meter square
eps_0 = 8.85e-012;    // Permittivity of free space, farad per metre
N_A = 6.023e+026;    // Avagadro's number
M = 32;    // Atomic weight of sulphur
rho = 2.08e+003;    // Density of sulphur atom, kg per cubic meter
// Since (eps_r - 1)/(eps_r + 2) = N*alphe_e/(3*eps_0), solvinf for  eps_r
ep_r = poly(0, 'ep_r');
ep_r = roots((ep_r - 1)*3*M*eps_0-(ep_r + 2)*N_A*rho*alpha_e);    // Relative permittivity of the medium
printf("\nThe relative dielectric constant for sulphur = %3.1f", ep_r);

// Result
// The relative dielectric constant for sulphur =
```

```
3.8
```

Scilab code Exa 10.5 Ionic polarizability for glass

```
// Scilab code Ex10.5: Pg:419 (2008)
clc;clear;
n = 1.5;    // Refractive index of glass
E = 1;    // For simplicity assume electric field strength to be unity, N/C
epsilon_0 = 8.85e-012;    // Permittivity of free space, farad per metre
epsilon_r = 6.75;    // Relative permittivity of free space at optical frequencies
mu = 1.5;    // Refractive index for glass
P_e = epsilon_0*(n^2 - 1)*E;        // Electronic polarizability, farad Sq.m
P_i = epsilon_0*(epsilon_r - n^2)*E;    // Ionic polarizability, farad Sq.m
percent_P_i = P_i/(P_e+P_i)*100;    // Percentage ionic polarizability
printf("\nPercent ionic polarizability for glass = %3.1f percent", percent_P_i);

// Result
// Percent ionic polarizability for glass = 78.3 percent
```

Scilab code Exa 10.6 Frequency and phase difference in the presence of dielectric

```
// Scilab code Ex10.6: Pg:422 (2008)
clc;clear;
```

```
eps_r_prime = 1;     // For simplicity assume real part of dielectric constant to be unity
eps_r_dprime = eps_r_prime;     // Imaginary part of dielectric constant is the same as that of real part
tau = 18e-06;     // Relaxation time of ice, s
f = 1/(2*%pi*tau*1e+003);     // Frequency when the real and imaginary parts of the complex dielectric constant will become equal, kHz
delta = atand(eps_r_dprime/eps_r_prime);     // Loss angle, degree
phi = 90 - delta;     // Phase difference between the current and voltage, degree
printf("\nThe frequency when the real and imaginary parts of the complex dielectric constant will become equal = %3.1f kHz", f);
printf("\nThe phase difference between the current and voltage = %2.0f degree", phi);

// Result
// The frequency when the real and imaginary parts of the complex dielectric constant will become equal = 8.8 kHz
// The phase difference between the current and voltage = 45 degree
```

Chapter 12

Fiber Optics

Scilab code Exa 12.1 Specifications of an optical fibre

```
// Scilab code Ex12.1: Pg:463 (2008)
clc;clear;
n1 = 1.5;     // Core index of an optical fibre
n0 = 1;     // Refractive index of air
delta = 0.0005;     // Intermodal dispersion factor for the fibre
// Since delta = (n1-n2)/n1, solving for n2
n2 = n1 - n1*delta;     // Refractive index of cladding
//As sind(phi_c) = n2/n1, solving for phi_c, we have
phi_c = asind(n2/n1);     // Critical internal reflection angle, degree
// As sind(theta_0) = sqrt(n1^2-n2^2)/n0, solving for theta_0
theta_0 = asind(sqrt(n1^2-n2^2)/n0);     // External critical acceptance angle, degree
NA = n1*sqrt(2*delta);     // Numerical aperture
printf("\nThe refractive index of cladding = %7.5f ", n2);
printf("\nThe critical internal reflection angle = %4.1f degree", phi_c);
```

```
printf("\nThe external critical acceptance angle = %4.2f degree", theta_0);
printf("\nThe numerical aperture = %6.4f ", NA);

// Result
// The refractive index of cladding = 1.49925
// The critical internal reflection angle = 88.2 degree
// The external critical acceptance angle = 2.72 degree
// The numerical aperture = 0.0474
```

Scilab code Exa 12.2 Acceptance angle for fiber in water

```
// Scilab code Ex12.2: Pg:464 (2008)
clc;clear;
n2 = 1.59;    // Cladding refractive index of an optical fibre
n0 = 1;    // Refractive index when the fiber is in air
NA = 0.20;    // Numerical aperture of fiber
// Since NA = sqrt(n_1^2-n_2^2)/n0, solving for n1
n1 = sqrt(NA^2 + n2^2)/n0;    // Core refractive index of fiber
// In water, n0 = 1.33
n0 = 1.33;    // Refractive index of water
NA = sqrt(n1^2-n2^2)/n0;    // Numerical aperture when the fiber is in water
theta_max = asind(NA);    // Acceptance angle  for the fiber in water, degree
printf("\nThe acceptance angle for the fibre = %3.1f degree", theta_max);

// Result
// The acceptance angle for the fibre = 8.6 degree
```

Scilab code Exa 12.3 Normalized frequency for the fiber

```
// Scilab code Ex12.3: Pg:467 (2008)
clc;clear;
n1 = 1.45;    // Core refractive index of an fibre
d = 0.6;    // Core  diameter of fiber , m
NA = 0.16;    // Numerical aperture of fiber
lambda_0 = 9e-007;    // Wavelength of light , m
V = %pi*d*NA/lambda_0;    // Normalized frequency (V
    -number)for the fiber
printf("\nThe normalized frequency for fiber = %4.2e
    ", V);

// Result
// The normalized frequency for fiber = 3.35e+005
```

Scilab code Exa 12.4 Normalized frequency and number of modes for the fiber

```
// Scilab code Ex12.4: Pg:468 (2008)
clc;clear;
n1 = 1.52;    // Core refractive index of an fibre
d = 29e-06;    // Core  diameter of fiber , m
delta = 0.0007;    // Fractional difference index
lambda_0 = 1.3e-06;    // Wavelength of light , m
// Since delta = (n1-n2)/n1, solving for n2
n2 = n1-n1*delta;    // Cladding refractive index of
    fiber
V = %pi*d*sqrt(n1^2 - n2^2)/lambda_0;    //
    Normalized frequency for the fiber
N = 1/2*V^2;    // Number of modes the fiber will
    support
```

```
printf("\nThe normalized frequency for fiber = %5.3f ", V);
printf("\nThe number of modes supported by the fiber = %1.0f ", N);

// Result
// The normalized frequency for fiber = 3.985
// The number of modes supported by the fiber = 8
```

Scilab code Exa 12.5 Single mode operation in step index fiber

```
// Scilab code Ex12.5: Pg:468 (2008)
clc;clear;
// Define function to convert degrees to degree, minute and second
function [deg, minute, second] = deg2dms(theta)
    deg = floor(theta);
    minute = floor((theta-deg)*60);
    second = floor(((theta-deg)*60-minute)*60);
endfunction
n1 = 1.480;    // Core refractive index of an optical fibre
n2 = 1.47;    // Cladding refractive index of an optical fibre
lambda_0 = 850e-09;    // wavelength of light, m
V = 2.405;    // Normalized frequency for single mode propagation of the fibre
// As V = %pi*d*sqrt(n1^2-n2^2)/lambda_0, solving for d
d = V*lambda_0/(%pi*sqrt(n1^2-n2^2)*1e-006);    // Core radius, micro-metre
NA = sqrt(n1^2-n2^2);    // Numerical aperture of the fiber
// Since sind(theta_0) = NA, solving for theta_0
theta_0 = asind(NA);    // The maximum acceptance
```

```
   angle of fiber, degree
[deg, m, s] = deg2dms(theta_0);    // Call
   conversion function
printf("\nThe core radius of the fiber = %4.2f micro
   -meter", d);
printf("\nThe numerical aperture of fiber = %6.4f ",
    NA);
printf("\nThe maximum acceptance angle = %d deg %d
   min %d sec", deg, m, s);

// Result
// The core radius of the fiber = 3.79 micro-meter
// The numerical aperture of fiber = 0.1718
// The maximum acceptance angle = 9 deg 53 min 23
   sec
```

Scilab code Exa 12.6 Output power level in optical fiber

```
// Scilab code Ex12.6: Pg:473 (2008)
clc;clear;
alpha = 3.5;    // Attenuation of optical signal, dB
   /km
Pi = 0.5e-003;    // Initial Power level of optical
   fibre, mW
L = 4;    // Lenght of optical fibre, km
// As alpha = (10/L)*log(Pi/Po), solving for Po
Po = Pi/10^(alpha*L/10);    // Output power level of
    optical fibre, micro-W
printf("\nThe output power level in optical fiber =
   %4.1f micro-W", Po/1e-006);

// Result
// The output power level in optical fiber = 19.9
   micro-W
```

Scilab code Exa 12.7 Attenuation of optical signal

```
// Scilab code Ex12.7: Pg:473 (2008)
clc;clear;
Pi = 1;      // Initial Power level of optical fibre , mW
Po = 0.85;      // Output Power level of optical fibre , mW
L = 0.5;      // Lenght of optical fibre , km
alpha = (10/L)*log10(Pi/Po);      // Attenuation of optical signal , dB/km
printf("\nThe attenuation of optical signal = %4.2f dB/km", alpha);

// Result
// The attenuation of optical signal = 1.41 dB/km
```

Scilab code Exa 12.8 Intermodal dispersion factor total dispersion and maximum bit rate of an optical fibre

```
// Scilab code Ex12.8: Pg:477 (2008)
clc;clear;
c = 3e+008;      // Speed of light , m/s
n1 = 1.5;      // Core index of an optical fibre
n2 = 1.498;      // Cladding index of an optical fibre
l = 18;      // Length of an optical fibre , km
D = (n1-n2)/n1;      // Intermodal dispersion factor for the fibre
// For a 1 km length fibre
delta = n1*1000/c*D/(1-D)*1e+009;      // intermodal dispersion factor for 1 km length fibre , ns/km
```

```
delta_t_total = delta*l;    // Total dispersion in
    18 km length , ns
B_max = 1/(5*delta_t_total*1e-009);    // Maximum
    bit rate , bits/sec
printf("\nThe intermodal dispersion factor for 1 km
    length fibre = %4.2f ns/km", delta );
printf("\nThe total dispersion in 18 km length fibre
     = %5.1f ns", delta_t_total);
printf("\nThe maximum bit rate allowed asuuming
    dispersion limiting = %4.2f M bits/s",B_max/1e
    +006);

// Result
// The intermodal dispersion factor for 1 km length
    fibre = 6.68 ns/km
// The total dispersion in 18 km length fibre =
    120.2 ns
// The maximum bit rate allowed asuuming dispersion
    limiting = 1.66 M bits/s
```

Scilab code Exa 12.9 Initial power level of an optical fibre

```
// Scilab code Ex12.9:Pg:478 (2008)
clc;clear;
P2 = 0.3e-006;    // Optical power level at the
    detector , W
dB_1 = 0.8*15;    // Connector loss , dB
dB_2 = 1.5*15;    // Fibre loss , dB
dB = dB_1 + dB_2;    // Total Loss , dB
// As dB = 10*log10(P1/P2) , solving for P1
P1 = P2*10^(dB/10)/1e-003;    // Initial power level
     of an optical fibre , mw
printf("\nThe initial power level of an optical
    fibre = %4.2f mW",P1 );

```

```
// Result
// The initial power level of an optical fibre =
   0.85 mW
```