Scilab Textbook Companion for Engineering Physics by D. C. Ghosh, N. C. Ghosh and P. K. Haldar[1]

Created by
Pankaj Biswas
Physics
Physics
Shri Mata Vaishno Devi University
College Teacher
Dr. Kamni
Cross-Checked by
Chaya

June 2, 2016

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

Book Description

Title: Engineering Physics

Author: D. C. Ghosh, N. C. Ghosh and P. K. Haldar

Publisher: University Science, New Delhi

Edition: 1

Year: 2008

ISBN: 978-81-318-0366-0

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

Classical Mechanics

Scilab code Exa 1.5 Force of contact between two masses

```
1 // Scilab Code Ex1.5: Page-11 (2008)
2 clc; clear;
3 m1 = 2;     // Mass of first body, kg
4 m2 = 1;     // Mass of second body, kg
5 F = 3;     // The horizontal force applied to the
     mass m1, N
6 F_prime = m2/(m1 + m2)*F;     // Force of contact
     between m1 and m2, N
7 printf("\nThe force of contact between m1 and m2 =
     %3.1f N", F_prime);
8 F_prime = m1/(m1 + m2)*F;     // Force of contact
     when F is applied to m2, N
9 printf("\nThe force of contact when F is applied to
     m2 = %3.1f N", F_prime);
10
11 // Result
12 // The force of contact between m1 and m2 = 1.0 N
13 // The force of contact when F is applied to m2 =
     2.0 N
```

Scilab code Exa 1.6 Direction of motion of a ball after momentum conservation during collision

```
1 // Scilab Code Ex1.6: Page-12 (2008)
2 clc; clear;
3 v = 1;    // Let the speed of the ball B be unity,
     unit
4 v_prime = v/2;    // Speed of the ball after the
     collision, unit
5 theta = atand(v_prime/v);    // The direction of
     motion of the ball A after collision, degree
6 printf("\nThe direction of motion of the ball after
     collision = %2.0f degree", theta);
7
8 // Result
9 // The direction of motion of the ball after
     collision = 27 degree
```

Scilab code Exa 1.9 Angular velocity of the combination of two wheels

```
1 // Scilab Code Ex1.9: Page-14 (2008)
2 clc; clear;
3 omega1 = 500;    // Angular speed of rotating shaft,
     r.p.m.
4 omega2 = 0;    // Initial angular speed of the
     second wheel, r.p.m.
5 I = 1;    // For simplicity assume moment of ineria
     of the wheels to be unity
6 I1 = I, I2 = I;    // Moment of inertia of wheels A
     and B, kg-Sq.m
7 // As I1*omega1 + I2*omega2 = (I1 + I2)*omega,
     solving for omega
```

```
8  omega = (I1*omega1 + I2*omega2)/(I1 + I2);     //
     Angular speed of the combination of two wheels, r
     .p.m.
9  printf("\nThe angular speed of the combination of
     two wheels = %3.0f r.p.m.", omega);
10
11 // Result
12 // The angular speed of the combination of two
     wheels = 250 r.p.m.
```

Scilab code Exa 1.10 Common velocity of a car truck system

```
1  // Scilab Code Ex1.10: : Page-14 (2008)
2  clc; clear;
3  m1 = 1200;     // Mass of the car, kg
4  m2 = 3600;     // Mass of the truck, kg
5  u1 = 30;     // Speed of the car, m/s
6  u2 = 20;     // Speed of the truck, m/s
7  theta = 60;     // Direction of motion of the truck w
     .r.t. that of car, degree
8  // As m1*u1 + m2*u2 = (m1 + m2)*v, solving for v
     along x and y directions
9  v_x = (m1*u1 + m2*u2*cosd(theta))/(m1 + m2);     //
     Common speed along x-direction, m/s
10 u1 = 0;     // The speed of the car after
     interlocking with the truck, m/s
11 v_y = (m1*u1 + m2*u2*sind(theta))/(m1 + m2);     //
     Common speed along y-direction, m/s
12 v = sqrt(v_x^2 + v_y^2);     // Common speed of the
     car-truck system, m/s
13 theta = atand(v_y/v_x);     // Direction of common
     velocity w.r.t. that of car, degree
14 printf("\nThe common speed of the car-truck system =
     %4.1f m/s", v);
15 printf("\nThe direction of common velocity = %4.1f
```

```
    degree north of east", theta);
16
17 // Result
18 // The common speed of the car-truck system = 19.8 m
    /s
```

Scilab code Exa 1.11 Velocity of third piece of the exploded object

```
1  // Scilab Code Ex1.11: Page-14 (2008)
2  clc; clear;
3  v1 = 20;     // Velocity of first piece, m/s
4  v2 = 30;     // Velocity of second piece, m/s
5  // From conservation of momentum, in x-direction
6  // m*v1*cosd(0)+m*v2*cosd(45)+m*v3*cosd(theta) = 0,
    solving for v3*cosd(theta)
7  v3_cos_theta = -(v1*cosd(0)+v2*cosd(45));     // x-
    component of v3 along theta, m/s
8  // From conservation of momentum, in y-direction
9  // m*v1*sind(0)-m*v2*sind(45)+m*v3*sind(theta) = 0,
    solving for v3*sind(theta)
10 v3_sin_theta = -(v1*sind(0)-v2*sind(45));     // y-
    component of v3 along theta, m/s
11 theta = atand(v3_sin_theta/v3_cos_theta);     //
    Direction of velocity of third piece, degree
12 v3 = -(v1*cosd(0)+v2*cosd(45))/cosd(theta+180);
    // Velocity of third piece, m/s
13 printf("\nThe velocity of third piece is %4.1f m/s
    towards %d degree north of west", v3, ceil(theta
    +180));
14
15 // Result
16 // The velocity of third piece is 46.4 m/s towards
    153 degree north of west
```

Chapter 2

Electricity and Magnetism

Scilab code Exa 2.12 Work done in moving a particle in force field

```
1 // Scilab Code Ex2.12: Page-80 (2008)
2 clc; clear;
3 t = poly(0, 't');
4 x = t^2 + 1;
5 y = 2*t^2;
6 z = t^3;
7 F = [3*x*y -5*z 10*x];     // Force acting on the
    particle, N
8 t1 = 1;     // lower limit
9 t2 = 2;     // upper limit
10 dr = [derivat(x); derivat(y); derivat(z)];     //
    Infinitesimal displacement, m
11 dW = F*dr;     // Work done or infinitesimally small
    displcement, J
12 work_exp = sci2exp(dW);     // Convert the polynomial
    to the expression
13 W = integrate(work_exp, 't', t1, t2);     // Total
    work done in moving the particle in a force field
    , J
14 printf("\nThe total work done in moving the particle
    in a force field = %d J", W);
```

```
15
16 // Result
```

Scilab code Exa 2.13 Evaluation of force integral

```
1 // Scilab Code Ex2.13: Page-80 (2008)
2 clc; clear;
3 x = poly(0, 'x');
4 y = x^2-4;
5 F = [x*y (x^2 + y^2)];     // Force acting on the
    particle, N
6 x1 = 2;     // lower limit
7 x2 = 4;     // upper limit
8 dr = [derivat(x); derivat(y);];     // Infinitesimal
    displacement, m
9 dW = F*dr;     // Work done or infinitesimally small
    displcement, J
10 work_exp = sci2exp(dW);     // Convert the polynomial
    to the expression
11 W = integrate(work_exp, 'x', x1, x2);     // Total
    work done in moving the particle in a force field
    , J
12 printf("\nThe total work done in moving the particle
    in the x-y plane = %d J", W);
13
14 // Result
15 // The total work done in moving the particle in the
    x-y plane = 732 J
```

Scilab code Exa 2.31 Electric flux through a surface area

```
1 // Scilab Code Ex2.31: Page-93 (2008)
2 clc; clear;
```

```
3 E = [3 4 8];     // Coefficients of i, j and k in the
      electric field, N/C
4 S = [0; 0; 100];     // Coefficients of i, j and k in
      the area vector, Sq. m
5 phi_E = E*S;     // Electric flux through the surface
    , N-Sq.m/C
6 printf("\nThe electric flux through the surface = %d
      N-Sq.m/C", phi_E);
7
8 // Result
9 // The electric flux through the surface = 800 N-Sq.
    m/C
```

Scilab code Exa 2.32 Electric flux through an area in XY plane

```
1 // Scilab Code Ex2.32: Page-93 (2008)
2 clc; clear;
3 E = [8 4 3];     // Coefficients of i, j and k in the
      electric field, N/C
4 S = [0; 0; 100];     // Coefficients of i, j and k in
      the area vector, Sq. m
5 phi_E = E*S;     // Electric flux through the surface
    , N-Sq.m/C
6 printf("\nThe electric flux through the area in XY
    plane = %d N-Sq.m/C", phi_E);
7
8 // Result
9 // The electric flux through the area in XY plane =
    300 N-Sq.m/C
```

Scilab code Exa 2.33 Electric flux through a surface in YZ plane

```
1 // Scilab Code Ex2.33: Page-93 (2008)
```

```
2 clc; clear;
3 E = [2 3 4];     // Coefficients of i, j and k in the
      electric field, N/C
4 S = [10; 0; 0];     // Coefficients of i, j and k in
     the area vector, Sq. m
5 phi_E = E*S;     // Electric flux through the surface
     , N-Sq.m/C
6 printf("\nThe electric flux through the surface in
     YZ plane = %d N-Sq.m/C", phi_E);
7
8 // Result
9 // The electric flux through the surface in YZ plane
      = 20 N-Sq.m/C
```

Scilab code Exa 2.39 Magnetic field due to a straight conductor carrying current

```
1 // Scilab Code Ex2.39: Page-96 (2008)
2 clc; clear;
3 mu_0 = 4*%pi*1e-007;     // Absolute magnetic
     permeability of free space, N/ampere-square
4 I = 15;     // Current through the wire, A
5 x = 1e-002;     // Distance of observation point from
      the wire, m
6 B = mu_0/(4*%pi)*2*I/x;     // Magnetic field at 1 cm
      distance, T
7 printf("\nThe magnetic field due to the current
     carrying wire at %d cm distance = %1.0e tesla", x
     /1e-002, B);
8 x = 5;     // Distance of observation point from the
     infinite straight conductor, m
9 I = 100;     // Current through the straight
     conductor, A
10 B = mu_0/(4*%pi)*2*I/x;     // Magnetic field at 1 cm
      distance, T
```

```
11 printf("\nThe magnetic field due to the current
      carrying infinite straight conductor at %d m
      distance = %1.0e tesla", x, B);
12
13 // Result
14 // The magnetic field due to the current carrying
      wire at 1 cm distance = 3e-004 tesla
15 // The magnetic field due to the current carrying
      infinite straight conductor at 5 m distance = 4e
      -006 tesla
```

Scilab code Exa 2.40 Force between two current carrying straight wires

```
1 // Scilab Code Ex2.40: Page-96 (2008)
2 clc; clear;
3 mu_0 = 4*%pi*1e-007;     // Absolute magnetic
     permeability of free space, N/ampere-square
4 I1 = 30;     // Current through the first wire, A
5 I2 = 40;     // Current through the second wire, A
6 x = 2;     // Separation distance between two wires,
     m
7 F = mu_0/(4*%pi)*2*I1*I2/x;     // Force between two
     current carrying straight wires, N
8 printf("\nThe force between two current carrying
     straight wires = %3.1e N", F);
9
10 // Result
11 // The force between two current carrying straight
     wires = 1.2e-004 N
```

Chapter 3

Vibration Waves and Light

Scilab code Exa 3.a.02 Frequency of particle executing SHM

```
1 // Scilab Code Ex3a.a.2: Page-132 (2008)
2 clc; clear;
3 m = 10;    // Mass of the particle, g
4 x = poly(0, 'x');
5 V = 50*x^2 + 100;    // Potential field surrounding
    the particle, erg/g
6 U = m*V;    // Potential energy of the particle
    field system, erg
7 F = -derivat(U);    // Force acting on the particle,
    dyne
8 // As F = -m*a = -m*omega^2*x = -m*(2%pi*f)^2*x,
    solving for f
9 f = sqrt(eval(pol2str(-pdiv(F,x)/m)))/(2*%pi);    //
    Frequency of oscillations of the particle
    executing SHM, Hz
10 printf("\nThe frequency of oscillations of the
    particle executing SHM = %4.2f Hz", f);
11
12 // Result
13 // The frequency of oscillations of the particle
    executing SHM = 1.59 Hz
```

Scilab code Exa 3.a.03 A body executing SHM

```
1 // Scilab Code Ex3a.a.3:Page-133 (2008)
2 clc; clear;
3 v1 = 80;     // Velocity of the body at 3 cm
     displacement, cm/s
4 v2 = 60;     // Velocity of the body at 4 cm
     displacement, cm/s
5 x1 = 3;     // Displacement of the body at velocity
     of 80 cm/s
6 x2 = 4;     // Displacement of the body at velocity
     of 60 cm/s
7 // As v = omega*sqrt(a^2 - x^2), solving for a
8 a = poly(0, 'a');
9 a = roots(v1^2*(a^2-16) - v2^2*(a^2 - 9));
10 omega = v1/sqrt(a(1)^2 - x1^2);     // Angular
     ferquency of the oscillations, rad/s
11 x = a(1);     // Maximum displacement, cm
12 // As x = a*sin(omega*t), solving for t
13 t_ex = asin(x/a(1))/omega;     // Time taken to reach
      the +ve extremity, s
14 d = a(1) - 2.5;     // Distance of the point from the
      mean position, cm
15 t = asin(d/a(1))/omega;     // Time taken to travel
     from mean position to positive extremity, s
16 printf("\nThe time taken to travel from 2.5 cm from
     +ve extremity = %5.3f s", t_ex - t);
17
18 // Result
19 // The time taken to travel from 2.5 cm from +ve
     extremity = 0.052 s
```

Scilab code Exa 3.a.04 Time period of a body oscillating in the tunnel across the earth

```
1 // Scilab Code Ex3a.a.4: Page-134 (2008)
2 clc; clear;
3 R = 6.4e+006;     // Radius of the earth, m
4 g = 10;     // Acceleration due to gravity, m/sec-
    square
5 T = 2*%pi*sqrt(R/g);     // Time period of
    oscillations of the body, s
6 printf("\nThe time period of oscillations of the
    body = %4.1f min", T/60);
7
8 // Result
9 // The time period of oscillations of the body =
    83.8 min
```

Scilab code Exa 3.a.05 Resultant amplitude and phase angle relative to the first SHM

```
1 // Scilab Code Ex3a.a.5: Page-135 (2008)
2 clc; clear;
3 phi1 = 0;     // Phase of the first SHM, degree
4 phi2 = 60;     // Phase of the second SHM, degree
5 phi3 = 90;     // Phase of the third SHM, degree
6 a1 = 1.0;     // Amplitude of the first SHM, cm
7 a2 = 1.5;     // Amplitude of the second SHM, cm
8 a3 = 2.0;     // Amplitude of the third SHM, cm
9 A = sqrt((a1 + a2*cosd(phi2)+a3*cosd(phi3))^2 + (a2*
    sind(phi2)+a3*sind(phi3))^2);     // Resultant
    amplitude relative to the first SHM, cm
10 phi = atand((a2*sind(phi2)+a3*sind(phi3))/(a1 + a2*
    cosd(phi2)+a3*cosd(phi3)));     // Resultant phase
     angle relative to the first SHM, degree
11 printf("\nThe resultant amplitude and phase angle
```

```
    relative to the first SHM = %4.2f cm and %2d
    degrees respectively", A, phi);
12
13 // Result
14 // The resultant amplitude and phase angle relative
    to the first SHM are 3.73 cm and 62 degrees
    respectively
```

Scilab code Exa 3.a.07 Two SHMs acting in the same direction

```
1  // Scilab Code Ex3a.a.7:Page-136 (2008)
2  clc; clear;
3  phi1 = 0;     // Phase of the first SHM, degree
4  phi2 = 45;     // Phase of the second SHM, degree
5  a1 = 0.005;     // Amplitude of the first SHM, m
6  a2 = 0.002;     // Amplitude of the second SHM, m
7  A = sqrt((a1 + a2*cosd(phi2))^2 + (a2*sind(phi2))^2)
    ;     // Resultant amplitude relative to the first
     SHM, m
8  phi = atand(a2*sind(phi2)/(a1 + a2*cosd(phi2)));
    // Resultant phase angle relative to the first
    SHM, degree
9  printf("\nThe amplitude of the resultant
    displacement and phase angle relative to the
    first SHM are %7.5f m and %5.2f degrees
    respectively", A, phi);
10
11 // Result
12 // The amplitude of the resultant displacement and
    phase angle relative to the first SHM are 0.00657
     m and 12.43 degrees respectively
```

Scilab code Exa 3.a.11 A spring disc system undergoing damped oscillation

```
1 // Scilab Code Ex3a.b.1: Page-138 (2008)
2 clc; clear;
3 m = 100;     // Mass of the horizontal disc, g
4 t = 60;     // Time during which the amplitude
     reduces to half of its undamped value, s
5 f = 10;     // Frequency of oscillations of the
     system, Hz
6 omega_prime = 2*%pi*f;     // Angular frequency of
     the oscillations, rad/s
7 A0 = 1;     // Assume the amplitude of undamped
     oscillations to be unity, cm
8 // As A = A0*exp(-k*t), solving for k
9 A = A0/2;     // Amplitude of damped oscillations
     after 1 min, cm
10 k = log(A0/A)/t;     // Resisting force per unit mass
     per unit velocity, nepers/sec
11 r = 2*k*m;     // Resistive force constant, sec/cm
12 tau = 1/k;     // Relaxation time, sec
13 Q = m*omega_prime/r;     // Quality factor
14 s = m*(omega_prime^2 + k^2);     // Force constant of
     the spring, dynes/Sq.cm
15 printf("\nThe resistive force constant = %4.2f dyne-
     sec/cm", r);
16 printf("\nThe relaxation time of the system = %4.2f
     sec", tau);
17 printf("\nThe quality factor, Q = %4.2f", Q);
18 printf("\nThe force constant of the spring = %4.2e
     dyne/Sq.cm", s);
19
20 // Result
21 // The resistive force constant = 2.31 dyne-sec/cm
22 // The relaxation time of the system = 86.56 sec
23 // The quality factor, Q = 2719.42
24 // The force constant of the spring = 3.95e+005 dyne
     /Sq.cm
```

Scilab code Exa 3.a.12 A mass executing damped oscillations in one dimension

```
1  // Scilab Code Ex3a.b.2: Page-139 (2008)
2  clc; clear;
3  function m = check_motion_type(k, omega0)
4      if k > omega0 then
5          m = 'aperiodic';
6      else if  k == omega0 then
7              m = 'criticallydamped';
8          else if  k < omega0 then
9                  m = 'oscillatory';
10             end
11         end
12     end
13 endfunction
14 m = 10;     // Mass of the body, g
15 s = 10;     // Restoring force, dyne/cm
16 r = 2;     // Resistive force constant, dyne.sec/cm
17 k = r/(2*m);     // Resisting force, nepers/sec
18 // As omega0^2 = s/m, solving for omega0
19 omega0 = sqrt(s/m);     // Angular frequency, rad/s
20 motion = check_motion_type(k, omega0);     // Check
       for the type of motion
21 r_new = 2*sqrt(m*s);     // Resistive force constant,
       dyne-sec/cm
22 m = r^2/(4*s);     // Mass for which the given forces
       makes the motion critically damped, g
23 printf("\nThe motion is %s in nature", motion);
24 printf("\nThe resistive force constant = %d dyne-sec
       /cm", r_new);
25 printf("\nThe mass for which the given forces makes
       the motion critically damped = %3.1f g", m);
26
```

```
27 // Result
28 // The motion is oscillatory in nature
29 // The resistive force constant = 20 dyne-sec/cm
30 // The mass for which the given forces makes the
   motion critically damped = 0.1 g
```

Scilab code Exa 3.a.14 A mass executing damped oscillations in one dimension

```
1 // Scilab Code Ex3a.b.4: Page-140 (2008)
2 clc; clear;
3 m = 1;    // Mass of the suspended body, kg
4 s = 25;    // Stifness constant of the spring, N/m
5 r = poly(0, 'r');
6 // As f0/f_prime = 2/sqrt(3), solving for r
7 r = roots(4*(s/m-r^2/(4*m^2))-3*s/m);    // Damping
   factor, kg/sec
8 printf("\nThe damping factor of damped oscillations
   = %d kg/sec", r(1));
9
10 // Result
11 // The damping factor of damped oscillations = 5 kg/
   sec
```

Scilab code Exa 3.a.15 Resisting force for critically damped motion

```
1
2 // Scilab Code Ex3a.b.5: Page-141 (2008)
3 clc; clear;
4 function m = check_motion_type(k, omega0)
5     if k > omega0 then
6         m = 'aperiodic';
7     else if  k == omega0 then
```

```
8               m = 'criticallydamped';
9           else if  k < omega0 then
10                  m = 'oscillatory';
11              end
12          end
13      end
14  endfunction
15  m = 10;     // Mass of the oscillating body, g
16  r = 2;     // Resisting force, dyne-sec/cm
17  s = 5;     // Restoring force, dyne/cm
18  k = r/(2*m);     // Resisting force, nepers/sec
19  // As omega0^2 = s/m, solving for omega0
20  omega0 = sqrt(s/m);     // Angular frequency, rad/s
21  motion = check_motion_type(k, omega0);     // Check
    for the type of motion
22  r = 2*sqrt(m*s);     // Resistive force constant for
    critical damping, dyne-sec/cm
23  printf("\nThe motion is %s in nature", motion);
24  printf("\nThe resistive force constant for critical
    damping = %4.1f dyne-sec/cm", r);
25
26  // Result
27  // The motion is oscillatory in nature
28  // The resistive force constant for critical damping
     = 14.1 dyne-sec/cm
```

Scilab code Exa 3.a.16 Damped oscillatory motion

```
1  // Scilab Code Ex3a.b.6: Page-141 (2008)
2  clc; clear;
3  m = 0.1;     // Mass of the oscillating body, kg
4  t = 50;     // Time during which the energy of system
     decays to 1/e of its undamped value, s
5  s = 10;     // Spring constant, N/m
6  E0 = 1;     // Assume the energy of undamped
```

```
    oscillations to be unity, erg
// As E = E0*exp(-k*t) and E/E0 = 1/e,  solving for
    k
E = E0/%e;     // Energy of damped oscillations after
    50 sec, erg
k = log(E0/E)/t;     // Resisting force per unit mass
    per unit velocity, nepers/sec
p = m*k;     // A resistive force constant, N-s/m
omega0 = sqrt(s/m);     // Angular frequency in the
    absence of damping, rad/sec
omega_prime = sqrt(omega0^2 - k^2/4);     // Angular
    frequency when damping takes place, rad/sec
Q = omega_prime/k;     // Quality factor
printf("\nThe resistive force constant, p = %1.0e N-
    s/m", p);
printf("\nThe quality factor, Q = %d", ceil(Q));

// Result
// The resistive force constant, p = 2e-003 N-s/m
// The quality factor, Q = 500
```

Scilab code Exa 3.a.17 Damped simple harmonic motion

```
// Scilab Code Ex3a.b.7: Page-142 (2008)
clc; clear;
t = 10;     // Time during which the amplitude
    reduces to 1/10th of its undamped value, s
f = 200;     // Frequency of oscillations of the
    system, Hz
omega0 = 2*%pi*f;     // Angular frequency of the
    oscillations, rad/s
A0 = 1;     // Assume the amplitude of undamped
    oscillations to be unity, cm
// As A = A0*exp(-k*t), solving for k
A = A0/10;     // Amplitude of damped oscillations
```

```
    after 10 sec, cm
k = log(A0/A)/t;    // Resisting force per unit mass
    per unit velocity, nepers/sec
tau = 1/(2*k);    // Relaxation time, sec
Q = omega0*tau;    // Quality factor
E0 = 1;    // Assume energy of undamped oscillations
    to be unity, erg
E = E0/10;    // Energy of damped oscillations after
    t sec, erg
// As E = E0*exp(-2*k*t), solving for t
t = 1/(2*k)*log(E0/E);    // Time during which the
   energy falls to 1/10 of its initial value, sec
printf("\nThe relaxation time = %4.2f sec", tau);
printf("\nThe quality factor, Q = %d", Q);
printf("\nThe time during which the energy falls to
   1/10 of its initial value = %d sec", t);
printf("\nThe damping constant, k = %4.2f", k);

// Result
// The relaxation time = 2.17 sec
// The quality factor, Q = 2728
// The time during which the energy falls to 1/10 of
    its initial value = 5 sec
// The damping constant, k = 0.23
// The answer for Q is given wrongly in the textbook
```

Scilab code Exa 3.a.21 Characteristics of progressive waves

```
// Scilab Code Ex3a.c.1: Page-143 (2008)
clc; clear;
// Comparing with the standard progressive wave
   equation, we have
a = 0.5;    // Amplitude of the wave, m
lambda = 2*%pi/12.56;    // Wavelength of the wave,
   m
```

```
v = 314/12.56;      // Wave velocity , m/s
nu = v/lambda;      // Frequency of the wave, Hz
printf("\nThe amplitude of the wave = %3.1f m", a);
printf("\nThe wavelength of the wave = %3.1f m",
   lambda);
printf("\nThe velocity of the wave = %d m/s", v);
printf("\nThe frequency of the wave = %d Hz", ceil(
   nu));

// Result
// The amplitude of the wave = 0.5 m
// The wavelength of the wave = 0.5 m
// The velocity of the wave = 25 m/s
// The frequency of the wave = 50 Hz
```

Scilab code Exa 3.a.22 A simple harmonic wave travelling along X axis

```
// Scilab Code Ex3a.c.2: Page-144 (2008)
clc; clear;
// Comparing with the standard progressive wave
   equation , we have
a = 5;      // Amplitude of the wave, m
nu = 0.2;      // Frequency of the wave, Hz
lambda = 1/0.5;      // Wavelength of the wave, m
v = nu*lambda;      // Wave velocity , m/s
printf("\nThe amplitude of the wave = %3.1f m", a);
printf("\nThe wavelength of the wave = %3.1f m",
   lambda);
printf("\nThe velocity of the wave = %3.1f m/s", v);
printf("\nThe frequency of the wave = %3.1f Hz", nu)
   ;

// Result
// The amplitude of the wave = 5.0 m
// The wavelength of the wave = 2.0 m
```

```
16 // The velocity of the wave = 0.4 m/s
17 // The frequency of the wave = 0.2 Hz
```

Scilab code Exa 3.a.23 Travelling wave characteristics and phase difference

```
1 // Scilab Code Ex3a.c.3: Page-144 (2008)
2 clc; clear;
3 // Comparing with the standard progressive wave
     equation, we have
4 a = 8;     // Amplitude of the wave, cm
5 nu = 4/2;     // Frequency of the wave, Hz
6 lambda = 2/0.02;     // Wavelength of the wave, cm
7 v = nu*lambda;     // Wave velocity, cm/s
8 delta_x = 20;     // Path difference between two
     particles, cm
9 delta_phi = delta_x*2*%pi/lambda*180/%pi;     //
     Phase difference between two particles, degree
10 printf("\nThe amplitude of the wave = %3.1f cm", a);
11 printf("\nThe wavelength of the wave = %3.1f cm",
     lambda);
12 printf("\nThe velocity of the wave = %3.1f cm/s", v)
     ;
13 printf("\nThe frequency of the wave = %d Hz", nu);
14 printf("\nThe phase difference between two particles
     = %d degree", delta_phi);
15
16 // Result
17 // The amplitude of the wave = 8.0 cm
18 // The wavelength of the wave = 100.0 cm
19 // The velocity of the wave = 200.0 cm/s
20 // The frequency of the wave = 2 Hz
21 // The phase difference between two particles = 72
     degree
```

Scilab code Exa 3.b.101 Brewster angle and angle of refraction for glass

```
1 // Scilab Code Ex3b.1: Page-163 (2008)
2 clc; clear;
3 mu = 1.5;     // Refractive indexof glass
4 i_p = atand(mu);     // Angle of polarization from
     Brewster's law, degree
5 r = 90 - i_p;     // Angale of refraction, degree
6 printf("\nThe Brewster angle for glass = %4.1f
     degree", i_p);
7 printf("\nThe angle of refraction for glass = %4.1f
     degree", r);
8
9 // Result
10 // The Brewster angle for glass = 56.3 degree
11 // The angle of refraction for glass = 33.7 degree
```

Scilab code Exa 3.b.102 Polarizing angles for various pair of media

```
1 // Scilab Code Ex3b.2: Page-163 (2008)
2 clc; clear;
3 // Function to convert degree to degree-minute
4 function [d,m]= deg2deg_min(deg)
5     d = int(deg);
6     m = (deg - d)*60;
7 endfunction
8 mu_air = 1;     // Refractive index fo air
9 mu_glass = 1.54;     // Refractive index of glass
10 mu_water = 1.33;     // Refractive index of water
11 // Air to glass incidence
12 i_p = atand(mu_glass/mu_air);     // Angle of
     polarization for air to glass incidence, degree
```

```
13 printf("\nFor air to glass, i_p = %d degree", i_p);
14 // glass to air incidence
15 i_p = atand(mu_air/mu_glass);    // Angle of
      polarization for glass to air incidence, degree
16 printf("\nFor glass to air, i_p = %d degree", ceil(
      i_p));
17 // Water to glass incidence
18 i_p = atand(mu_glass/mu_water);    // Angle of
      polarization for water to glass incidence, degree
19 [d,m] = deg2deg_min(i_p);    // Call function to
      convert to deg-min
20 printf("\nFor water to glass, i_p = %d degree %d min
      ", d, m);
21 // Glass to water incidence
22 i_p = atand(mu_water/mu_glass);    // Angle of
      polarization for glass to water incidence, degree
23 [d,m] = deg2deg_min(i_p);    // Call function to
      convert to deg-min
24 printf("\nFor glass to water, i_p = %d degree %d min
      ", d, m);
25 // Air to water incidence
26 i_p = atand(mu_water/mu_air);    // Angle of
      polarization for air to water incidence, degree
27 [d,m] = deg2deg_min(i_p);    // Call function to
      convert to deg-min
28 printf("\nFor air to water, i_p = %d degree %d min",
       d, m);
29 // Water to air incidence
30 i_p = atand(mu_air/mu_water);    // Angle of
      polarization for water to airincidence, degree
31 [d,m] = deg2deg_min(i_p);    // Call function to
      convert to deg-min
32 printf("\nFor water to air, i_p = %d degree %d min",
       d, m);
33
34 // Result
35 // For air to glass, i_p = 57 degree
36 // For glass to air, i_p = 33 degree
```

```
37 // For water to glass, i_p = 49 degree 11 min
38 // For glass to water, i_p = 40 degree 48 min
39 // For air to water, i_p = 53 degree 3 min
40 // For water to air, i_p = 36 degree 56 min
```

Scilab code Exa 3.b.103 Polarizing angle for glass

```
1 // Scilab Code Ex3b.3: Page-163 (2008)
2 clc; clear;
3 C = 40;    // Critical angle for glass to air
4 mu = 1/sind(C);    // Refractive index of glass w.r.
     t. air
5 i_p = atand(mu);    // Polarizing angle for glass,
     degree
6 printf("\nThe polarizing angle for glass = %4.1f
     degree", i_p);
7
8 // Result
9 // The polarizing angle for glass = 57.3 degree
```

Scilab code Exa 3.b.104 Polarization by reflection

```
1 // Scilab Code Ex3b.4: Page-164 (2008)
2 clc; clear;
3 i = 60;    // Angle of incidence, degree
4 i_p = i;    // Angle of polarization, degree
5 mu = tand(i_p);    // Refractive index of the medium
6 r = 90 - i;    // Angle of refraction, degree
7 printf("\nThe refractive index of transparent medium
     = %5.3f", mu);
8 printf("\nThe angle of refraction, r = %d degree", r
     );
```

```
9 printf("\nThe reflected and transmitted components
    are at right angles to each other.");
10
11 // Result
12 // The refractive index of transparent medium =
    1.732
13 // The angle of refraction , r = 30 degree
14 // The reflected and transmitted components are at
    right angles to each other.
```

Scilab code Exa 3.b.105 Intensity ratio of two beams through analyser

```
1 // Scilab Code Ex3b.5: Page-164 (2008)
2 clc; clear;
3 theta_A = 30;    // Angle between principal sections
    of polariser and  analyser for beam A, degree
4 theta_B = 60;    // Angle between principal sections
    of polariser and  analyser for beam B, degree
5 // As I_A*cosd(theta_A)^2 = I_B*cosd(theta_B)^2,
    solving for I ratio
6 I_ratio = cosd(theta_B)^2/cosd(theta_A)^2;    // The
    intensity ratio of the two beams
7 printf("\nThe intensity ratio of the two beams = %4
    .2f", I_ratio);
8
9 // Result
10 // The intensity ratio of the two beams = 0.33
```

Scilab code Exa 3.b.106 Percentage reduction in the intensity of the inident light

```
1 // Scilab Code Ex3b.6: Page-165 (2008)
2 clc; clear;
```

```
3 theta = [30 45 60 90];     // Angles between
    principal sections of polariser and analyser,
    degree
4 for i = 1:1:4
5     P_red = (1-cosd(theta(i))^2)*100;     //
        Percentage reduction in intensity of incident
         light
6     printf("\nFor theta = %d degree, percentage
        reduction = %1.0f percent", theta(i), P_red);
7 end
8
9 // Result
10 // For theta = 30 degree, percentage reduction = 25
    percent
11 // For theta = 45 degree, percentage reduction = 50
    percent
12 // For theta = 60 degree, percentage reduction = 75
    percent
13 // For theta = 90 degree, percentage reduction = 100
     percent
```

Scilab code Exa 3.b.107 Angle of rotation of polaroid to reduce the intensity

```
1 // Scilab Code Ex3b.7: Page-165 (2008)
2 clc; clear;
3 // For half reduction in intensity
4 I_ratio = 1/2;     // Intensity ratio
5 theta = acosd(sqrt(I_ratio));     // Angle of
    rotation of polaroid, degree
6 printf("\nFor half reduction in intensity, the angle
     of rotation = %d degree", theta);
7 // For one-fourth reduction in intensity
8 I_ratio = 1/4;     // Intensity ratio
9 theta = acosd(sqrt(I_ratio));     // Angle of
```

```
    rotation of polaroid, degree
printf("\nFor one-fourth reduction in intensity, the
    angle of rotation = %d degree", theta);

// Result
// For half reduction in intensity, the angle of
    rotation = 45 degree
// For one-fourth reduction in intensity, the angle
    of rotation = 60 degree
```

Scilab code Exa 3.c.202 Ratio of maximum to minimum intensity in the interference fringe system

```
// Scilab Code Ex3c.2: Page-184 (2008)
clc; clear;
I2 = 1;     // Assume intensity of light beam from
    the second source to be unity
I1 = 81*I2;     // Intensity of light beam from the
    first source
a = sqrt(I1);     // Width of the first slit, mm
b = sqrt(I2);     // Width of the second slit, mm
I_max = (1+a/b)^2;     // Maximum intensity in the
    fringe pattern
I_min = (1-a/b)^2;     // Minimum intensity in the
    fringe pattern
fact = gcd([I_max,I_min]); // Find l.c.m. of I_max
    and I_min
printf("\nThe ratio of maximum to minimum intensity
    in the fringe system, I_max:I_min = %d:%d", I_max
    /4, I_min/4);

// Result
// The ratio of maximum to minimum intensity in the
    fringe system, I_max:I_min = 25:16
```

Scilab code Exa 3.c.203 Wavelength of light from interference of fringes

```
1 // Scilab Code Ex3c.3: Page-184 (2008)
2 clc; clear;
3 d = 0.1;     // Separation between the two slits, cm
4 D = 100;     // Distance between the source and the
     slit, cm
5 bita = 0.05;     // Fringe width, cm
6 lambda = bita*d/D;     // Wavelength of light, cm
7 printf("\nThe wavelength of light used = %4d
     angstrom", lambda/1e-008);
8
9 // Result
10 // The wavelength of light used = 5000 angstrom
```

Scilab code Exa 3.c.204 Fringe width from interference pattern

```
1 // Scilab Code Ex3c.4: Page-184 (2008)
2 clc; clear;
3 d = 0.3;     // Separation between the two slits, cm
4 D = 60;     // Distance between the source and the
     slit, cm
5 lambda = 59e-006;     // Wavelength of light, cm
6 bita = lambda*D/d;     // Fringe width, cm
7 printf("\nThe fringe width = %4.2e cm", bita);
8
9 // Result
10 // The fringe width = 1.18e-002 cm
```

Scilab code Exa 3.c.205 Distance between the two coherent sources

```
1 // Scilab Code Ex3c.5: Page-185 (2008)
2 clc; clear;
3 D = 80;     // Distance between the source and the
     slit, cm
4 lambda = 5890e-008;     // Wavelength of light, cm
5 bita = 9.424e-002;     // Fringe width, cm
6 d = lambda*D/bita;     // Separation between the two
     slits, cm
7 printf("\nThe distance between the two coherent
     sources = %4.2f cm", d);
8
9 // Result
10 // The distance between the two coherent sources =
     0.05 cm
```

Scilab code Exa 3.c.206 Distance between consecutive interference bands

```
1 // Scilab Code Ex3c.6: Page-185 (2008)
2 clc; clear;
3 D = 100;     // Distance between the source and the
     slit, cm
4 lambda = 5893e-008;     // Wavelength of light, cm
5 d1 = 4.05e-001;     // Distance between the images of
      the two slits in one position, cm
6 d2 = 2.90e-001;     // Distance between the images of
      the two slits in second position, cm
7 d = sqrt(d1*d2);     // Separation between the two
     slits, cm
8 bita = lambda*D/d;     // Fringe width, cm
9 printf("\nThe distance between consecutive
     interference bands = %6.4f cm", bita);
10
11 // Result
```

```
12 // The distance between consecutive interference
      bands = 0.0172 cm
```

Scilab code Exa 3.c.207 Wavelength of the light used in biprism experiment

```
1 // Scilab Code Ex3c.7: Page-185 (2008)
2 clc; clear;
3 D = 1.2;     // Distance between the source and the
     slit, m
4 d = 7.5e-004;    // Separation between the two slits,
     cm
5 n = 20;     // Number of fringes crossed in the field
     of view
6 bita = 1.888e-002/n;     // Fringe width, cm
7 lambda = bita*d/D;     // Wavelength of light, cm
8 printf("\nThe wavelength of the light used in
     biprism experiment = %4d angstrom", lambda/1e
     -010);
9
10 // Result
11 // The wavelength of the light used in biprism
     experiment = 5900 angstrom
```

Scilab code Exa 3.c.208 Number of fringes obtained with the given wavelength

```
1 // Scilab Code Ex3c.8: Page-186 (2008)
2 clc; clear;
3 lambda1 = 5893;     // First wavelength of light,
     angstrom
4 lambda2 = 4358;     // Second wavelength of light,
     angstrom
```

```
5 n = 40;     // Number of fringes obtained with first
    wavelength
6 // As bita1/bita2 = lambda1/lambda2, so
7 x = n*lambda1/lambda2;     // Number of fringes
    obtained with the seocond wavelength
8 printf("\nThe number of fringes obtained with the
    given wavelength = %d", x);
9
10 // Result
11 // The number of fringes obtained with the given
    wavelength = 54
```

Scilab code Exa 3.c.209 Wavelength of light from biprism interference pattern

```
1 // Scilab Code Ex3c.9: Page-186 (2008)
2 clc; clear;
3 D = 100;     // Distance between the source and the
    slit, cm
4 bita = 0.0135;     // Fringe width, cm
5 alpha = %pi/360;     // Angle of refracting face with
    the base of biprism, radian
6 mu = 1.5;     // Refractive index of the material of
    biprism
7 x = 50;     // Distance between slit and the biprism,
    cm
8 d = 2*(mu-1)*x*alpha;     // Separation between the
    two virtual slits, cm
9 lambda = bita*d/D;     // Wavelength of light, cm
10 printf("\nThe wavelength of light from biprism
    interference pattern = %4d angstrom", lambda/1e
    -008);
11
12 // Result
13 // The wavelength of light from biprism interference
```

```
pattern = 5890 angstrom
```

Scilab code Exa 3.c.210 Fringe width observed at one metre distance from biprism

```
1 // Scilab Code Ex3c.10: Page-187 (2008)
2 clc; clear;
3 mu = 1.5;     // Refractive index of the material of
    biprism
4 alpha = %pi/180;     // Base angle of biprism, radian
5 D = 110;     // Distance between the source and the
    slit, cm
6 x = 10;     // Distance between slit and the biprism,
    cm
7 d = 2*(mu-1)*x*alpha;    // Separation between the
    two virtual slits, cm
8 lambda = 5900e-008;     // Wavelength of light, cm
9 bita = lambda*D/d;     // Fringe width, cm
10 printf("\nThe fringe width observed at one metre
    distance from biprism = %6.4f cm", bita);
11
12 // Result
13 // The fringe width observed at one metre distance
    from biprism = 0.0372 cm
```

Scilab code Exa 3.c.211 Wavelength of light in Newton ring experiment

```
1 // Scilab Code Ex3c.11: Page-187 (2008)
2 clc; clear;
3 D_n = 0.42;     // Diameter of nth ring, cm
4 D_mplusn = 0.7;     // Diameter of (m+n)th ring, cm
5 m = 14;     // Difference between (m+n)th and nth
    rings
```

```
6 R = 100;     // Radius of curvature of the plano-
    convex lens, m
7 lambda = (D_mplusn^2 - D_n^2)/(4*m*R);     //
    Wavelength of the light, cm
8 printf("\nThe wavelength of the light used = %4d
    angstrom", lambda/1e-008);
9
10 // Result
11 // The wavelength of the light used = 5600 angstrom
```

Scilab code Exa 3.c.212 Radius of plano convex lens

```
1 // Scilab Code Ex3c.12: Page-187 (2008)
2 clc; clear;
3 D5 = 0.336;     // Diameter of 5th ring, cm
4 D10plus5 = 0.590;     // Diameter of (10+5)th ring,
    cm
5 m = 10;     // Difference between (10+5)th and 5th
    rings
6 lambda = 5890e-008;     // Wavelength of the light,
    cm
7 R = (D10plus5^2 - D5^2)/(4*m*lambda);     // Radius
    of curvature of the plano-convex lens, m
8 printf("\nThe radius of plano convex lens = %5.2f cm
    ", R);
9
10 // Result
11 // The radius of plano convex lens = 99.83 cm
```

Scilab code Exa 3.c.213 Wavelenght of light used in obtaining Newton rings

```
1 // Scilab Code Ex3c.13: Page-187 (2008)
```

```
clc; clear;
D3 = 0.181;     // Diameter of 3rd ring, cm
D23 = 0.501;     // Diameter of 23rd ring, cm
m = 23-3;     // Difference between (m+n)th and nth rings
R = 50;     // Radius of curvature of the plano-convex lens, m
lambda = (D23^2 - D3^2)/(4*m*R);     // Wavelength of the light, cm
printf("\nThe wavelength of the light used = %4d angstrom", lambda/1e-008);

// Result
// The wavelength of the light used = 5456 angstrom
```

Scilab code Exa 3.c.214 Diameter of the 20th dark ring

```
// Scilab Code Ex3c.14: Page-188 (2008)
clc; clear;
D4 = 0.4;     // Diameter of 4th ring, cm
D12 = 0.7;     // Diameter of 12th ring, cm
m = 12-4;     // Difference between (m+n)th and nth rings
lambda_R = (D12^2 - D4^2)/(4*m);     // Wavelength-Radius product, Sq.cm
D20 = sqrt(80*lambda_R);     // Diameter of the 20th dark ring, cm
printf("\nThe diameter of the 20th dark ring = %5.3f cm", D20);

// Result
// The diameter of the 20th dark ring = 0.908 cm
```

Scilab code Exa 3.c.215 Radius of curvature of the lens and the thickness of the air film

```
1 // Scilab Code Ex3c.15: Page-188 (2008)
2 clc; clear;
3 D10 = 0.50;     // Diameter of 10th ring, cm
4 n = 10;     // Number of dark fringe
5 lambda = 6250e-008;     // Wavelength of light used,
    cm
6 R = D10^2/(4*n*lambda);     // Radius of curvature of
    the lens, cm
7 t = D10^2/(8*R);     // Thickness of the air film, cm
8 printf("\nThe radius of curvature of the lens = %3d
    cm", R);
9 printf("\nThe thickness of the air film = %9.7f cm",
    t);
10
11 // Result
12 // The radius of curvature of the lens = 100 cm
13 // The thickness of the air film = 0.0003125 cm
```

Scilab code Exa 3.c.216 Newton rings observed in reflected light

```
1 // Scilab Code Ex3c.16: Page-188 (2008)
2 clc; clear;
3 D10 = 5e-003;     // Diameter of 10th ring, cm
4 n = 10;     // Number of dark fringe
5 lambda = 5.9e-007;     // Wavelength of reflected
    light, m
6 R = D10^2/(4*n*lambda);     // Radius of curvature of
    the lens, cm
7 t = D10^2/(8*R);     // Thickness of the air film, cm
8 printf("\nThe radius of curvature of the lens = %5.3
    f m", R);
9 printf("\nThe thickness of the air film = %4.2e m",
```

```
    t);
10
11 // Result
12 // The radius of curvature of the lens = 1.059 m
13 // The thickness of the air film = 2.95e-006 m
```

Scilab code Exa 3.c.217 Smallest thickness of the glass film in which it appears dark

```
1 // Scilab Code Ex3c.17: Page-189 (2008)
2 clc; clear;
3 lambda = 5893e-010;     // Wavelength of light used,
    m
4 mu = 1.5;     // Refractive index of glass film
5 r = 60;     // Angle of reflection in the film,
    degree
6 t = lambda/(2*mu*cosd(r));     // Smallest thickness
    of the
7 printf("\nThe smallest thickness of the glass film
    when it appears dark = %6.1f angstrom", t/1e-010)
    ;
8
9 // Result
10 // The smallest thickness of the glass film when it
    appears dark = 3928.7 angstrom
```

Scilab code Exa 3.d.301 Wavelength of light used in diffraction due to narrow slit

```
1 // Scilab Code Ex3d.1: Page-205 (2008)
2 clc; clear;
3 D = 200;     // Distance between the source and the
    slit, cm
```

```
4 a = 0.02;     // Slit width, cm
5 x = 0.5;     // Position of first minimum, cm
6 n = 1;     // Order of diffraction
7 lambda = a*x/(D*n);     // Wavelength of light used,
    cm
8 printf("\nThe wavelength of light used = %4d
    angstrom", lambda/1e-008);
9
10 // Result
11 // The wavelength of light used = 5000 angstrom
```

Scilab code Exa 3.d.302 Separation between the second minima on either side of the central maximum

```
1 // Scilab Code Ex3d.2: Page-205 (2008)
2 clc; clear;
3 f = 20;     // Focal length of the lens, cm
4 a = 0.06;     // Slit width, cm
5 n = 2;     // Order of diffraction
6 lambda = 6e-005;     // Wavelength of light used, cm
7 x = 2*lambda*f/a;     // Separation between the
    second minima on either side of the central
    maximum, cm
8 printf("\nThe separation between the second minimum
    an central maximum = %4.2f cm", x);
9
10 // Result
11 // The separation between the second minimum an
    central maximum = 0.04 cm
```

Scilab code Exa 3.d.303 Distance of the first dark band from the axis

```
1 // Scilab Code Ex3d.3: Page-206 (2008)
```

```
clc; clear;
n = 1;     // Order of diffraction
f = 40;     // Focal length of the lens, cm
a = 0.03;     // Slit width, cm
lambda = 5890e-008;     // Wavelength of the light
   used, cm
// As a*sind(theta) = n*lambda, solving for theta
theta = asin(n*lambda/a);     // The angle of
   diffraction corresponding to the first minimum,
   radian
x = f*theta;     // The distance of the first dark
   band from the axis, cm
printf("\nThe distance of the first dark band from
   the axis = %6.4f cm", x);

// Result
// The distance of the first dark band from the axis
    = 0.0785 cm
```

Scilab code Exa 3.d.304 Angle of diffraction for the principal maxima

```
// Scilab Code Ex3d.4: Page-206 (2008)
clc; clear;
lambda1 = 5890e-008;     // Wavelength of D1 line of
   sodium lamp, cm
lambda2 = 5896e-008;     // Wavelength of D2 line of
   sodium lamp, cm
d_lambda = lambda2 - lambda1;     // Wavelength
   difference, cm
w = 0.5;     // Width of the grating, cm
N = 2500;     // Total number of grating lines
N_prime = N/w;     // Number of lines per cm, lines/
   cm
a_plus_b = 1/N_prime;     // Grating element, cm
n = 1;     // Order of diffraction
```

```
11 // Case 1
12 theta = asind(n*lambda1/a_plus_b);     // Angle of
      diffraction for D1 line, degree
13 // Case 2
14 theta_prime = asind(n*lambda2/a_plus_b);     // Angle
       of diffraction for D2 line, degree
15 printf("\nThe angle of diffraction for D1 and D2
      lines of sodium are %5.2f dgree and %5.2f degree
      respectively.", theta, theta_prime);
16 // From the condition for just resolution, lambda/
      d_lambda = n*N, solving for N
17 N_min = lambda1/(d_lambda*n);     // Minimum number
      of lines required on the grating
18 if N_min < N then
19     printf("\nThe two lines are well resolved.");
20 else
21     printf("\nThe two lines are not resolved.");
22 end
23
24 // Result
25 // The angle of diffraction for D1 and D2 lines of
      sodium are 17.13 dgree and 17.15 degree
      respectively.
26 // The two lines are well resolved.
```

Scilab code Exa 3.d.305 Wavelength of the spectral line

```
1 // Scilab Code Ex3d.5: Page-207 (2008)
2 clc; clear;
3 N = 4250;     // Number of lines per cm of grating,
      lines/cm
4 a_plus_b = 1/N;     // Grating element, cm
5 n = 2;     // Order of diffraction
6 theta = 30;     // Angle of diffraction, degree
7 lambda = sind(theta)*a_plus_b/n;     // Wavelength of
```

```
    spectral line from diffraction condition, cm
printf("\nThe wavelength of spectral line from
    diffraction condition = %4d angstrom", lambda/1e
    -008);

// Result
// The wavelength of spectral line from diffraction
    condition = 5882 angstrom
```

Scilab code Exa 3.d.306 Number of lines in one centimeter of the grating surface

```
// Scilab Code Ex3d.6: Page-207 (2008)
clc; clear;
n = 2;     // Order of diffraction
lambda = 5e-005;     // Wavelength of light, cm
theta = 30;     // Angle of diffraction, degree
N = sind(theta)/(n*lambda);     // Number of lines
    per cm of grating, lines/cm
printf("\nThe number of lines per cm of grating =
    %4d per cm", ceil(N));

// Result
// The number of lines per cm of grating = 5000 per
    cm
```

Scilab code Exa 3.d.307 Highest order spectrum obtainable with the given diffraction grating

```
// Scilab Code Ex3d.7: Page-208 (2008)
clc; clear;
N = 5000;     // Number of lines per cm ruled on
    grating, lines/cm
```

```
4 lambda = 6e-005;     // Wavelength of light, m
5 a_plus_b = 1/N;      // Grating element, m
6 theta = 90;     // Maximum angle of diffraction,
     degree
7 n = a_plus_b*sind(theta)/lambda;     // Order of
     diffraction
8 printf("\nIn highest order spectrum obtainable with
     the given diffraction grating = %4.2f", n);
9
10 // Result
11 // In highest order spectrum obtainable with the
     given diffraction grating = 3.33
```

Scilab code Exa 3.d.308 Invisible third and higher order principal maxima in a diffraction grating

```
1 // Scilab Code Ex3d.8: Page-208 (2008)
2 clc; clear;
3 lambda = 5.5e-007;     // Wavelength of light, m
4 a_plus_b = 1.5e-006;     // Grating element, m
5 theta = 90;     // Maximum angle of diffraction,
     degree
6 n = a_plus_b*sind(theta)/lambda;     // Order of
     diffraction
7 printf("\nIn this diffraction grating only %dnd
     order will be visible while %drd and higher
     orders are not possible.", n, n+1);
8
9 // Result
10 // In this diffraction grating only 2nd order will
     be visible while 3rd and higher orders are not
     possible.
```

Scilab code Exa 3.d.309 Number of lines per cm on the grating

```
1 // Scilab Code Ex3d.9: Page-208 (2008)
2 clc; clear;
3 theta = 30;     // Maximum angle of diffraction,
     degree
4 lambda1 = 5400e-010;     // Wavelength of light
     giving certain diffraction order, m
5 lambda2 = 4050e-010;     // Wavelength of light
     giving higher diffraction order, m
6 n = poly(0, 'n');
7 n = roots(lambda1*n-(n+1)*lambda2);     // Order of
     diffraction for first wavelength
8 a_plus_b = n*lambda1/sind(theta);     // Grating
     element, m
9 N = 1/a_plus_b;     // Number of lines per cm ruled
     on grating, lines/cm
10 printf("\nThe number of lines per cm on the
     diffraction grating = %d lines per cm", N/100);
11
12 // Result
13 // The number of lines per cm on the diffraction
     grating = 3086 lines per cm
```

Scilab code Exa 3.d.310 Minimum number of lines on the diffraction grating

```
1 // Scilab Code Ex3d.10: Page-209 (2008)
2 clc; clear;
3 lambda = 5890e-008;     // Wavelength of light, cm
4 n = 1;     // Order of diffraction
5 d_lambda = 6e-008;     // Difference in wavelengths
     of D1 and D2 lines, cm
6 N = lambda/(n*d_lambda);     // Number of lines on
     grating
```

```
7 printf("\nThe minimum number of lines on the
    diffraction grating = %d", ceil(N));
8
9 // Result
10 // The minimum number of lines on the diffraction
    grating = 982
```

Scilab code Exa 3.d.311 Design of a plane transmission diffraction grating

```
1 // Scilab Code Ex3d.11: Page-209 (2008)
2 clc; clear;
3 lambda = 6000e-008;     // Wavelength of light, cm
4 n = 2;     // Order of diffraction
5 d_lambda = 6e-008;     // Difference in wavelengths
    of D1 and D2 lines, cm
6 N = lambda/(n*d_lambda);     // Number of lines on
    grating
7 printf("\nThe minimum number of lines in the
    required diffraction grating = %d", N);
8
9 // Result
10 // The minimum number of lines in the required
    diffraction grating = 500
```

Scilab code Exa 3.d.312 Minimum number of lines per cm in grating to just resolve the D1 and D2 lines of sodium

```
1 // Scilab Code Ex3d.12: Page-209 (2008)
2 clc; clear;
3 lambda = 5890e-008;     // Wavelength of light, cm
4 n = 2;     // Order of diffraction
```

```
5 d_lambda = 6e-008;     // Difference in wavelengths
    of D1 and D2 lines, cm
6 w = 2.5;     // Width of the grating, cm
7 N = lambda/(n*d_lambda);     // Number of lines on
    grating
8 printf("\nThe minimum number of lines per cm in the
    diffraction grating = %5.1f", N/w);
9
10 // Result
11 // The minimum number of lines per cm in the
    diffraction grating = 196.3
```

Scilab code Exa 3.d.313 Maximum resolving power of a plane transmission grating

```
1 // Scilab Code Ex3d.13: Page-210 (2008)
2 clc; clear;
3 lambda = 5000e-008;     // Wavelength of light, cm
4 theta = 90;     // Angle of diffraction for the
    maximum resolving power, degree
5 N = 40000;     // Number of lines on grating
6 a_plus_b = 12.5e-005;     // Grating element, cm
7 n = 2;     // Order of diffraction
8 n_max = N*a_plus_b*sind(theta)/lambda;     // Maximum
    resolving power
9 printf("\nThe maximum resolving power = %d", n_max);
10
11 // Result
12 // The maximum resolving power = 100000
```

Scilab code Exa 3.d.314 Maximum number of lines of a grating

```
1 // Scilab Code Ex3d.14: Page-209 (2008)
```

```
clc; clear;
lambda = 5890e-008;     // Wavelength of light , cm
n = 3;     // Order of diffraction
d_lambda = 6e-008;     // Difference in wavelengths
   of D1 and D2 lines , cm
N = lambda/(n*d_lambda);     // Maximum number of
   lines of a grating
printf("\nThe maximum number of lines of the grating
    = %d", N);

// Result
// The maximum number of lines of the grating = 327
```

Chapter 4

Special Theory of Relativity

Scilab code Exa 4.1 Fringe shift in the Michelson Morley experiment

```
1 // Scilab Code Ex4.1: Page-233 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light in vacuum, m/s
4 v = 3e+004;     // Speed of earth, m/s
5 d = 7;     // Effective length of each path, m
6 lambda = 7000e-010;     // Wavelength of light used,
    m
7 n = 2*d*v^2/(lambda*c^2);     // Fringe shift
8 printf("\nThe expected fringe shift = %3.1f", n);
9
10 // Result
11 // The expected fringe shift = 0.2
```

Scilab code Exa 4.2 Apparent length of rod relative to the observer

```
1 // Scilab Code Ex4.2: Page-233 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light in vacuum, m/s
```

```
4 v = 3e+007;     // Speed of metre rod, m/s
5 L0 = 1;     // Actual length of the rod, m
6 L = L0*sqrt(1-v^2/c^2);     // Apparent length of rod
      from Lorentz transformation, m
7 printf("\nThe apparent length of rod realtive to the
      observer = %5.3f m", L);
8
9 // Result
10 // The apparent length of rod realtive to the
     observer = 0.995 m
```

Scilab code Exa 4.3 Apparent length of a meter stick for different speeds

```
1 // Scilab Code Ex4.3: Page-234 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light in vacuum, m/s
4 v = [c c/sqrt(2) sqrt(3)/2*c c/2 0.8*c];     //
     Different speeds of metre rod, m/s
5 L0 = 100;     // Actual length of the rod, cm
6 for i = 1:1:5
7     L = L0*sqrt(1-v(i)^2/c^2);     // Apparent length
          of rod from Lorentz transformation, m
8     printf("\nFor v = %4.2e m/s, L = %4.1f cm", v(i)
          , L);
9 end
10
11 // Result
12 // For v = 3.00e+008 m/s, L =  0.0 cm
13 // For v = 2.12e+008 m/s, L = 70.7 cm
14 // For v = 2.60e+008 m/s, L = 50.0 cm
15 // For v = 1.50e+008 m/s, L = 86.6 cm
16 // For v = 2.40e+008 m/s, L = 60.0 cm
```

Scilab code Exa 4.4 Lorentz transformations applied to a rigid bar

```
1 // Scilab Code Ex4.4: Page-235-236 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light in vacuum, m/s
4 // Part (a)
5 v = 0.98*c ;     // Speed of the rigid bar, m/s
6 L2 = 1.5;     // Length of the rigid bar in S_prime
     frame, m
7 L1 = L2*sqrt(1-v^2/c^2);     // Apparent length of
     rod from Lorentz transformation, m
8 theta2 = 45;     // Angle which the bar makes w.r.t.
     x-aixs in S_prime frame, degree
9 theta1 = atand(tand(theta2)/sqrt(1-v^2/c^2));     //
     Orientation of bar relative to S frame, degree
10 printf("\nThe orientation of the %d m bar relative
     to S frame = %4.1f degree", L2, theta1);
11 // Part(b)
12 v = 0.6*c ;     // Speed of the rigid bar, m/s
13 L2 = 5;     // Length of the rigid bar in S_prime
     frame, m
14 L1 = L2*sqrt(1-v^2/c^2);     // Apparent length of
     rod from Lorentz transformation, m
15 theta2 = 30;     // Angle which the bar makes w.r.t.
     x-aixs in S_prime frame, degree
16 theta1 = atand(tand(theta2)/sqrt(1-v^2/c^2));     //
     Orientation of bar relative to S frame, degree
17 printf("\nThe orientation of the %d m bar relative
     to S frame = %4.1f degree", L2, theta1);
18
19 // Result
20 // The orientation of the 1 m bar relative to S
     frame = 78.7 degree
21 // The orientation of the 5 m bar relative to S
     frame = 35.8 degree
```

Scilab code Exa 4.5 Velocity of pi meson

```
1 // Scilab Code Ex4.5: Page-236 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light in vacuum, m/s
4 t0 = 2.5e-008;     // Proper life time of pi-meson, s
5 t = 2.5e-007;     // MEan life time of pi-meson, s
6 // As t = t0/(sqrt(1-v^2/c^2)), solving for v
7 v = sqrt(1-(t0/t)^2)*c;     // Velocity of pi meson,
    m/s
8 printf("\nThe velocity of pi meson = %5.3f c = %4.2e
    m/s", v/c, v);
9
10 // Result
11 // The velocity of pi meson = 0.995 c = 2.98e+008 m/
    s
```

Scilab code Exa 4.6 Relative speed of the ships as measured by an observer

```
1 // Scilab Code Ex4.6: Page-237 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light in vacuum, m/s
4 v = 0.8*c;     // Speed of the first spaceship, m/s
5 u_prime = 0.9*c;     // Speed of the second spaceship
    , m/s
6 u = (u_prime+v)/(1+u_prime*v/c^2);     // Relative
    speed of the ships as measured by the observer on
    either one from Velocity addition rule, m/s
7 printf("\nThe relative speed of the ships as
    measured by an observer in either one = %5.3f c =
    %4.2e m/s", u/c, u);
```

```
8
9  // Result
10 // The relative speed of the ships as measured by an
       observer in either one = 0.988 c = 2.97e+008 m/s
```

Scilab code Exa 4.7 Velocity of one particle relative to the other

```
1  // Scilab Code Ex4.7: Page-237 (2008)
2  clc; clear;
3  c = 3e+008;     // Speed of light in vacuum, m/s
4  v = 0.9*c;     // Speed of the first particle, m/s
5  u_prime = 0.9*c;     // Speed of the oppositely
       moving second particle, m/s
6  u = (u_prime+v)/(1+u_prime*v/c^2);     // Velocity of
       one particle relative to the other from Velocity
       addition rule, m/s
7  printf("\nThe velocity of one particle relative to
       the other = %5.3f c = %4.2e m/s", u/c, u);
8
9  // Result
10 // The velocity of one particle relative to the
       other = 0.994 c = 2.98e+008 m/s
```

Scilab code Exa 4.8 Velocity of the rocket as observed from the earth

```
1  // Scilab Code Ex4.8: Page-237 (2008)
2  clc; clear;
3  c = 3e+008;     // Speed of light in vacuum, m/s
4  // Case 1: when velocity of firing is away from the
       earth
5  v = 0.5*c;     // Speed of the rocket away from the
       earth, m/s
```

```
u_prime = 0.8*c;     // Speed of the outgoing
    spaceship relative to earth, m/s
u = (u_prime+v)/(1+u_prime*v/c^2);     // Velocity of
    rocket moving away relative to the earth, m/s
printf("\nThe velocity of rocket moving away
    relative to the earth = %4.2f c = %4.2e m/s", u/c
    , u);
// Case 2: when velocity of firing is towards the
    earth
v = 0.5*c;     // Speed of the rocket moving towards
    the earth, m/s
u_prime = -0.8*c;     // Speed of the outgoing
    spaceship relative to earth, m/s
u = (u_prime+v)/(1+u_prime*v/c^2);     // Velocity of
    approaching rocket relative to the earth, m/s
printf("\nThe velocity of approaching rocket
    relative to the earth = %3.1f c = %3.1e m/s", u/c
    , u);

// Result
// The velocity of rocket moving away relative to
    the earth = 0.93 c = 2.79e+008 m/s
// The velocity of approaching rocket relative to
    the earth = -0.5 c = -1.5e+008 m/s
```

Scilab code Exa 4.9 Velocity of the particle when its total energy is thrice its rest energy

```
// Scilab Code Ex4.9: Page-237 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
E0 = 1;     // Assume the rest energy of the particle
    to be unity
E = 3*E0;     // Total energy of the particle
v = sqrt(1-(E0/E)^2)*c;     // Velocity of the
```

```
    particle from relativistic variation of mass with
     speed, m/s
printf("\nThe velocity of the particle when its
    total energy is thrice its rest energy = %5.3e cm
    /s", v);

// Result
// The velocity of the particle when its total
    energy is thrice its rest energy = 2.828e+008 cm/
    s
```

Scilab code Exa 4.10 Relativisti variation of mass of electron with velocity

```
// Scilab Code Ex4.10: Page-238 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
m0 = 9.1e-031;     // Rest mass of the electron, kg
E0 = m0*c^2;     // Rest energy of the electron, J
printf("\nThe rest energy of the electron = %4.2f
    MeV", E0/1.6e-013);
E = 1.25*E0;     // Total energy of the particle
v = sqrt(1-(E0/E)^2)*c;     // Velocity of the
    particle from relativistic variation of mass with
     speed, m/s
printf("\nThe velocity of the electron when its
    total energy is 1.25 times its rest energy = %3.1
    f c = %3.1e cm/s", v/c, v);

// Result
// The rest energy of the electron = 0.51 MeV
// The velocity of the electron when its total
    energy is 1.25 times its rest energy = 0.6 c =
    1.8e+008 cm/s
```

Scilab code Exa 4.11 An electron subjected to relativistic motion

```
1 // Scilab Code Ex4.11: Page-238 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light in vacuum, m/s
4 v = 0.99*c;     // Speed of the electron, m/s
5 m0 = 9.1e-031;     // Rest mass of the electron, kg
6 m = m0/sqrt(1-v^2/c^2);     // Moving mass of the
    electron, kg
7 E = m*c^2;     // Total energy of the electron, J
8 printf("\nThe total energy of the electron = %4.2e J
    ", E);
9 KE_ratio = m0/(2*(m-m0))*(v/c)^2;     // Ratio of
    Newtonian kinetic energy to the relativistic
    kinetic energy
10 printf("\nThe ratio of Newtonian kinetic energy to
    the relativistic kinetic energy = %4.2f",
    KE_ratio);
11
12 // Result
13 // The total energy of the electron = 5.81e-013 J
14 // The ratio of Newtonian kinetic energy to the
    relativistic kinetic energy = 0.08
```

Chapter 5

Quantum Mechanics

Scilab code Exa 5.2 Temperature of the surface of sun

```
1 // Scilab Code Ex5.2: Page-284 (2008)
2 clc; clear;
3 lambda_m = 4753e-010;     // Wavelength from the sun
     at which maximum energy is emitted, m
4 b = 2.88e-003;     // Wein's constant, m-K
5 T = b/lambda_m;     // Temperature of the surface of
     sun
6 printf("\nThe temperature of the surface of sun = %d
     K", ceil(T));
7
8 // Result
9 // The temperature of the surface of sun = 6060 K
```

Scilab code Exa 5.3 Wavelength of maximum intensity of radiation

```
1 // Scilab Code Ex5.3: Page-284 (2008)
2 clc; clear;
3 b = 2.898e-003;     // Wein's constant, m-K
```

```
4 T = 3000 + 273;     // Temperature of the source, K
5 lambda_m = b/T;     // Wavelength of maximum
    intensity of radiation emitted from the source, m
6 printf("\nThe wavelength of maximum intensity of
    radiation emitted from the source = %d angstrom",
     lambda_m/1e-010);
7
8 // Result
9 // The wavelength of maximum intensity of radiation
    emitted from the source = 8854 angstrom
```

Scilab code Exa 5.4 Kinetic energy of the ejected photoelectrons

```
1 // Scilab Code Ex5.4: Page-285 (2008)
2 clc; clear;
3 h = 6.62e-034;     // Planck's constant, Js
4 c = 3e+008;     // Speed of light, m/s
5 lambda = 2300e-010;     // Thereshold wavelength for
    tungsten, m
6 phi = h*c/lambda;     // Work function for tungsten,
    J
7 lambda = 1800e-010;     // Wavelength of incident
    radiation, m
8 E = h*c/lambda;     // Energy of the incidnt
    radiation, J
9 KE = E - phi;     // Kinetic energy of the ejected
    photoelectrons, J
10 printf("\nThe kinetic energy of the ejected
    photoelectrons = %3.1f eV", KE/1.6e-019);
11
12 // Result
13 // The kinetic energy of the ejected photoelectrons
    = 1.5 eV
```

Scilab code Exa 5.5 Possibility of electron emission with the given incident wavelengths

```
1  // Scilab Code Ex5.5: Page-285 (2008)
2  clc; clear;
3  function [] = check_energy(E, L)
4  phi = 4.8;     // Work function for tungsten, eV
5      if E > phi then
6          printf("\nThe wavelength %d angstrom will be
               able to liberate an electron.", ceil(L/1
               e-010));
7      else
8          printf("\nThe wavelength %d angstrom will
               not be able to liberate an electron.",
               ceil(L/1e-010));
9      end
10 endfunction
11 h = 6.62e-034;     // Planck's constant, Js
12 c = 3e+008;     // Speed of light, m/s
13 // Case 1
14 lambda = 2000e-010;     // Wavelength of incident
       radiation, m
15 E = h*c/(lambda*1.6e-019);     // Energy of the
       incidnt radiation, eV
16 check_energy(E, lambda);     // Check for the
       wavelength
17 // Case 2
18 lambda = 5000e-010;     // Wavelength of incident
       radiation, m
19 E = h*c/(lambda*1.6e-019);     // Energy of the
       incidnt radiation, eV
20 check_energy(E, lambda);     // Check for the
       wavelength
21
```

```
22 // Result
23 // The wavelength 2000 angstrom will be able to
      liberate an electron.
24 // The wavelength 5000 angstrom will not be able to
      liberate an electron.
```

Scilab code Exa 5.6 Velocity of emitted photoelectrons

```
1  // Scilab Code Ex5.6: Page-286 (2008)
2  clc; clear;
3  h = 6.62e-034;     // Planck's constant, Js
4  c = 3e+008;     // Speed of light, m/s
5  e = 1.6e-019;     // Energy quivalent of 1 eV, J
6  phi = 2.28*e;     // Work function for material, J
7  m = 9.1e-031;     // Mass of an electron, kg
8  lambda = 3000e-010;     // Wavelength of incident
      radiation, m
9  E = h*c/lambda;     // Energy of the incidnt
      radiation, J
10 KE = E - phi;     // Kinetic energy of the ejected
      photoelectrons, J
11 v = sqrt(2*KE/m);     // Velocity of emitted electron
      , m/s
12 printf("\nThe velocity of the emitted electron = %4
      .2e m/s", v);
13
14 // Result
15 // The velocity of the emitted electron = 8.08e+005
      m/s
```

Scilab code Exa 5.7 A photosensitive material emitting photoelectrons

```
1  // Scilab Code Ex5.7: Page-286 (2008)
```

```
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
e = 1.6e-019;     // Energy quivalent of 1 eV, J
phi = 4.2*e;     // Work function for material, J
lambda = 2000e-010;     // Wavelength of incident
   radiation, m
E = h*c/lambda;     // Energy of the incidnt
   radiation, J
KE_fast = (E - phi)/e;     // Kinetic energy of the
   fastest photoelectron, eV
KE_slow = 0;     // Kinetic energy of the slowest
   photoelectron, eV
printf("\nThe kinetic energy of the fastest
   photoelectron = %d eV", KE_fast);
printf("\nThe kinetic energy of the slowest
   photoelectron = %d eV", KE_slow);
V = (E - phi)/e;     // Stopping potential, V
printf("\nThe stopping potential = %d volt", V);

// Result
// The kinetic energy of the fastest photoelectron =
    2 eV
// The kinetic energy of the slowest photoelectron =
    0 eV
// The stopping potential = 2 volt
```

Scilab code Exa 5.8 Maximum wavelength of radiation which would start the emission of photoelectrons

```
// Scilab Code Ex5.8: Page-287 (2008)
clc; clear;
h = 6.62e-027;     // Planck's constant, erg-s
c = 3e+010;     // Speed of light, cm/s
phi = 3.31e-012;     // Work function for material,
```

```
    erg
6 lambda0 = h*c/phi;     // Wavelength of incident
    radiation , cm
7 printf("\nThe maximum wavelength of radiation which
    would start the emission of photoelectrons = %d
    angstrom", lambda0/1e-008);
8
9 // Result
10 // The maximum wavelength of radiation which would
    start the emission of photoelectrons = 6000
    angstrom
```

Scilab code Exa 5.9 Potassium surface exposed to UV radiation

```
1 // Scilab Code Ex5.9: Page-287 (2008)
2 clc; clear;
3 h = 6.62e-034;     // Planck's constant , Js
4 c = 3e+008;     // Speed of light , m/s
5 e = 1.6e-019;     // Energy quivalent of 1 eV, J
6 phi = 2.1*e;     // Work function for material , J
7 lambda = 3500e-010;     // Wavelength of incident UV
    radiation , m
8 E = 1e-004;     // Energy incident per sec on 1 Sq.
    cm of potassium surface , J
9 eta = 0.5/100;     // Efficiency of potassium surface
10 KE = (h*c/lambda-phi)/e;     // Maximum kinetic
    energy of the ejected photoelectrons , eV
11 N = eta*E/(KE*e);     // Number of photoelectrons
    emitted per second per Sq. cm of potassium
    surface
12 printf("\nThe maximum kinetic energy of the incidnt
    radiation = %4.2f eV", KE);
13 printf("\nThe number of photoelectrons emitted per
    second per Sq. cm of potassium surface = %4.2e",
    N);
```

```
14
15 // Result
16 // The maximum kinetic energy of the incidnt
      radiation = 1.45 eV
17 // The number of photoelectrons emitted per second
      per Sq. cm of potassium surface = 2.16e+012
```

Scilab code Exa 5.10 Planck constant and threshold wavelength of metal

```
1 // Scilab Code Ex5.10: Page-288 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light, m/s
4 KE1 = 3.62e-019;     // Maximum kinetic energy of
      photoelectrons with first wavelength, eV
5 lambda1 = 3000;     // First wavelength of incident
      radiation, angstrom
6 KE2 = 0.972e-019;     // Maximum kinetic energy of
      photoelectrons with second wavelength, eV
7 lambda2 = 5000;     // Second wavelength of incident
      radiation, angstrom
8 A = [c/lambda1, -1; c/lambda2, -1];     // Declare a
      square matrix as per Einstein's Photoelectric
      relation, KE = h*c/lambda - phi
9 B = [KE1; KE2];     // Put KEs in a column matrix
10 X = inv(A)*B;     // Apply inverse multiplication of
      a matrix to fing h and phi
11 lambda0 = X(1)*1e-010*c/X(2);     // Threshold
      wavelength of metal, m
12 printf("\nh = %4.2e Js\nphi = %1.0e J", X(1)*1e-010,
     X(2));
13 printf("\nThe threshold wavelength of metal = %d
      angstrom", ceil(lambda0/1e-010));
14
15 // Result
16 // h = 6.62e-034 Js
```

```
17 // phi = 3e-019 J
18 // The threshold wavelength of metal = 6620 angstrom
```

Scilab code Exa 5.11 Energy and wavelength of incident photon

```
1  // Scilab Code Ex5.11: Page-288 (2008)
2  clc; clear;
3  c = 3e+008;     // Speed of light, m/s
4  e = 1.6e-019;     // Energy equivalent of 1 eV, J
5  h = 6.62e-034;     // Planck's constant, Js
6  m0 = 9.1e-031;     // Rest mass of an electron, kg
7  alpha = 90;     // Scattering angle for X-ray photon,
       degree
8  d_lambda = h/(m0*c)*(1-cosd(alpha));     //
     Wavelength shift after collision, m
9  lambda = d_lambda;     // Wavelength of the incident
     photon according to the condition, m
10 E = h*c/(lambda*e*1e+006);     // Energy of the
     incident photon, MeV
11 printf("\nThe wavelength of the incident photon = %6
     .4e m", lambda);
12 printf("\nThe energy of the incident photon = %4.2f
     MeV", E);
13
14 // Result
15 // The wavelength of the incident photon = 2.4249e
     -012 m
16 // The energy of the incident photon = 0.51 MeV
```

Scilab code Exa 5.12 Energy lost by an X ray photon in collsion with an electron

```
1  // Scilab Code Ex5.12: Page-289 (2008)
```

```
clc; clear;
c = 3e+008;     // Speed of light, m/s
e = 1.602e-019;     // Energy equivalent of 1 eV, J
h = 6.6e-034;     // Planck's constant, Js
lambda = 0.1;     // Wavelength of X ray photon,
   angstrom
m0 = 9.1e-031;     // Rest mass of an electron, kg
alpha = 90;     // Scattering angle for X-ray photon,
    degree
d_lambda = h/(m0*c*1e-010)*(1-cosd(alpha));     //
   Wavelength shift after collision, angstrom
lambda_prime = lambda + d_lambda;     // Wavelength
   of the scattered photon, angstrom
dE = h*c*1e+010/e*(1/lambda - 1/lambda_prime);     //
    Energy lost by the X ray photon by collision, eV
printf("\nThe energy lost by the X ray photon by
   collision = %4.1f KeV", dE/1e+003);

// Result
// The energy lost by the X ray photon by collision
   = 24.1 KeV
```

Scilab code Exa 5.13 The Compton effect stidied at different scattering angles

```
// Scilab Code Ex5.13: Page-289 (2008)
clc; clear;
c = 3e+008;     // Speed of light, m/s
e = 1.602e-019;     // Energy equivalent of 1 eV, J
h = 6.6e-034;     // Planck's constant, Js
m0 = 9.1e-031;     // Rest mass of an electron, kg
alpha = [90 60 45 180];     // Different scattering
   angle for X-ray photon, degrees
d_lambda = zeros(4);
for i = 1:1:4
```

```
10      d_lambda(i) = h/(m0*c*1e-010)*(1-cosd(alpha(i)))
            ;     // Wavelength shift after collision,
            angstrom
11      printf("\nFor alpha = %d degree, d_lambda = %6.4
            f angstrom", alpha(i), d_lambda(i));
12  end
13  lambda = 0.2;     // Given wavelength of incident X-
        ray photon, angstrom
14  lambda_prime = lambda + d_lambda(3);     //
        Wavelength of the scattered photon at 45 degree,
        angstrom
15  printf("\nThe wavelength of the photon scattered at
        45 degree = %5.3f angstrom", lambda_prime);
16  lambda_prime = lambda + d_lambda(4);     // Maximum
        wavelength of the photon scattered at 180 degree,
         angstrom
17  KE_max = h*c*1e+010*(1/lambda - 1/lambda_prime);
        // Maximum kinetic energy of the recoil electron,
         J
18  printf("\nThe maximum kinetic energy of the recoil
        electron = %4.2e J", KE_max);
19
20  // Result
21  // For alpha = 90 degree, d_lambda = 0.0242 angstrom
22  // For alpha = 60 degree, d_lambda = 0.0121 angstrom
23  // For alpha = 45 degree, d_lambda = 0.0071 angstrom
24  // For alpha = 180 degree, d_lambda = 0.0484
        angstrom
25  // The wavelength of the photon scattered at 45
        degree = 0.207 angstrom
26  // The maximum kinetic energy of the recoil electron
         = 1.93e-015 J
```

Scilab code Exa 5.15 de Broglie wavelength associated with moving masses

```
// Scilab Code Ex5.15: Page-292 (2008)
clc; clear;
h = 6.6e-034;     // Planck's constant, Js
// For golf ball
m = 0.046;     // Mass of the golf ball, kg
v = 36;     // Velocity of the golf ball, m/s
lambda = h/(m*v);     // de-Broglie wavelength
   associated with the moving golf ball, m
printf("\nThe de-Broglie wavelength associated with
   the moving golf ball = %1.0e m", lambda);
if lambda/1e-010 > 0.1 then
    printf("\nThe moving golf ball may exhibit wave
       character.");
end
// For an electron
m = 9.11e-031;     // Mass of the electron, kg
v = 1e+007;     // Velocity of the electron, m/s
lambda = h/(m*v);     // de-Broglie wavelength
   associated with the moving electron, m
printf("\nThe de-Broglie wavelength associated with
   the moving electron = %3.1e m", lambda);
if lambda/1e-010 > 0.1 then
    printf("\nThe moving electron may exhibit wave
       character.");
end

// Result
// The de-Broglie wavelength associated with the
   moving golf ball = 4e-034 m
// The de-Broglie wavelength associated with the
   moving electron = 7.2e-011 m
// The moving electron may exhibit wave character.
```

Scilab code Exa 5.16 Voltage applied to the electron microscope to produce the required wavelength

```
// Scilab Code Ex5.16: Page-292 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
e = 1.602e-019;     // Energy equivalent of 1 eV, J
lambda = 0.40e-010; // de-Broglie wavelength
   associated with the moving electron, m
m = 9.11e-031;     // Rest mass of an electron, kg
V = (h/lambda)^2/(2*m*e);     // Voltage applied to
   the electron microscope to produce the required
   wavelength, volt
printf("\nThe voltage applied to the electron
   microscope to produce the required de-Broglie
   wavelength = %5.1f volt", V);

// Result
// The voltage applied to the electron microscope to
    produce the required de-Broglie wavelength =
   938.4 volt
```

Scilab code Exa 5.18 de Broglie wavelength of a neutron of given energy

```
// Scilab Code Ex5.18: Page-293 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
e = 1.602e-019;     // Energy equivalent of 1 eV, J
E_k = 12.8e+006;     // Energy of the moving neutron,
    eV
m0 = 1.675e-027;     // Rest mass of a neutron, kg
lambda = h/sqrt(2*m0*E_k*e) // de-Broglie wavelength
    associated with the moving neutron, m
printf("\nThe de-Broglie wavelength of the moving
   neutron = %3.1e angstrom", lambda/1e-010);

// Result
// The de-Broglie wavelength of the moving neutron =
```

```
8.0e-005 angstrom
```

Scilab code Exa 5.19 Minimum uncertainty in momentum and kinetic energy of a proton confined within nucleus

```
1 // Scilab Code Ex5.19: Page-294 (2008)
2 clc; clear;
3 h = 6.62e-034;     // Planck's constant, Js
4 e = 1.602e-019;     // Energy equivalent of 1 eV, J
5 m = 1.67e-027;     // Rest mass of a proton, kg
6 r = 5e-015;     // Radius of the nucleus, m
7 delta_x = 2*r;     // Minimum uncertainty in position
      of the proton, m
8 delta_p = h/(2*%pi*delta_x);     // Minimum
    uncertainty in proton's momentum, kg-m/s
9 KE = delta_p^2/(2*m);     // Minimum kinetic emergy
    of the proton, J
10 printf("\nThe minimum uncertainty in momentum of the
      proton = %4.2e kg-m/s", delta_p);
11 printf("\nThe minimum kinetic emergy of the proton =
      %5.3f MeV", KE/(e*1e+006));
12
13 // Result
14 // The minimum uncertainty in momentum of the proton
      = 1.05e-020 kg-m/s
15 // The minimum kinetic emergy of the proton = 0.207
    MeV
```

Scilab code Exa 5.20 Minimum uncertainty in the measurement of velocity of the electron

```
1 // Scilab Code Ex5.20: Page-294 (2008)
2 clc; clear;
```

```
3 h = 6.62e-034;     // Planck's constant, Js
4 m = 9.11e-031;     // Rest mass of a electron, kg
5 delta_x = 1e-009;     // Minimum uncertainty in
     position of the electron, m
6 delta_p_min = h/delta_x;     // Minimum uncertainty
     in electron's momentum, kg-m/s
7 delta_v = delta_p_min/m;     // Minimum uncertainty
     in the measurement of velocity of the electron, m
     /s
8 printf("\nThe minimum uncertainty in the measurement
     of velocity of the electron = %4.2e m/s",
     delta_v);
9
10 // Result
11 // The minimum uncertainty in the measurement of
     velocity of the electron = 7.27e+005 m/s
```

Scilab code Exa 5.22 Minimum uncertainty in the position of the particle

```
1 // Scilab Code Ex5.22: Page-295 (2008)
2 clc; clear;
3 h = 6.62e-034;     // Planck's constant, Js
4 m = 1e-009;     // Mass of the particle, kg
5 v = 1;     // Velocity of the particle, m/s
6 delta_v = v*0.01/100;     // Minimum uncertainty in
     the velocity of the particle, m/s
7 delta_x = h/(m*delta_v);     // Minimum uncertainty
     in the position of the particle, m
8 printf("\nThe minimum uncertainty in the position of
     the particle = %4.2e m", delta_x);
9
10 // Result
11 // The minimum uncertainty in the position of the
     particle = 6.62e-021 m
```

Scilab code Exa 5.23 Uncertainty with which position of the electron can be located

```
1 // Scilab Code Ex5.23: Page-295 (2008)
2 clc; clear;
3 h = 6.62e-034;     // Planck's constant, Js
4 m = 9.1e-031;     // Mass of the electron, kg
5 v = 1e+003;     // Velocity of the electron, m/s
6 delta_v = v*0.05/100;     // Minimum uncertainty in
     the velocity of the electron, m/s
7 delta_x = h/(m*delta_v);     // Minimum uncertainty
     in the position of the electron, m
8 printf("\nThe minimum uncertainty in the position of
     the electron = %4.2e m", delta_x);
9
10 // Result
11 // The minimum uncertainty in the position of the
     electron = 1.45e-003 m
```

Scilab code Exa 5.24 Minimum uncertainty in energy of the excited state of an atom

```
1 // Scilab Code Ex5.24: Page-295 (2008)
2 clc; clear;
3 h = 6.62e-034;     // Planck's constant, Js
4 e = 1.602e-019;     // Energy equivalent of 1 eV, J
5 delta_t = 1e-008;     // Life time of excited state
     of an atom, s
6 delta_E = h/(2*%pi*delta_t);     // Minimum
     uncertainty in the energy of the excited state of
     the atom, J
```

```
7 printf("\nThe minimum uncertainty in the energy of
    the excited state of the atom = %3.1e eV",
    delta_E/e);
8
9 // Result
10 // The minimum uncertainty in the energy of the
    excited state of the atom = 6.6e-008 eV
```

Scilab code Exa 5.25 Probable uncertainty in energy and frequency of gamma ray photon

```
1 // Scilab Code Ex5.25: Page-296 (2008)
2 clc; clear;
3 h = 6.62e-034;     // Planck's constant, Js
4 delta_t = 1e-012;     // Life time of a nucleus in
    the excited state, s
5 delta_E = h/(2*%pi*delta_t);     // Minimum
    uncertainty in the energy of the excited state of
     the nucleus, J
6 // As E = h*nu, solving for delta_nu
7 delta_nu = delta_E/h;     // Minimum uncertainty in
    the frequency of the excited state of the nucleus
    , Hz
8 printf("\nThe minimum uncertainty in the energy of
    the excited state of the nucleus = %5.3e J",
    delta_E);
9 printf("\nThe minimum uncertainty in the frequency
    of the excited state of the nucleus = %4.2e MHz",
     delta_nu/1e+006);
10
11 // Result
12 // The minimum uncertainty in the energy of the
    excited state of the nucleus = 1.054e-022 J
13 // The minimum uncertainty in the frequency of the
    excited state of the nucleus = 1.59e+005 MHz
```

Scilab code Exa 5.29 Lowest energy of an electron in one dimensional force free region

```
1 // Scilab Code Ex5.29: Page-300 (2008)
2 clc; clear;
3 h = 6.62e-034;     // Planck's constant, Js
4 e = 1.602e-019;     // Energy equivalent of 1 eV, J
5 m = 9.11e-031;     // Rest mass of the electron, kg
6 l = 4e-010;     // Length of the force free region, m
7 n = 1;     // Principal quantum number for lowest
     energy state
8 E1 = n^2*h^2/(8*m*l^2);     // Lowest energy of an
     electron in one dimensional force free region, J
9 printf("\nThe lowest energy of an electron in one
     dimensional force free region = %4.2f eV", E1/e);
10
11 // Result
12 // The lowest energy of an electron in one
     dimensional force free region = 2.35 eV
```

Scilab code Exa 5.30 The excited state energies of the particle entrapped in a one dimensional box

```
1 // Scilab Code Ex5.30: Page-300 (2008)
2 clc; clear;
3 e = 1.602e-019;     // Energy equivalent of 1 eV, J
4 E1 = 3.2e-018/e;     // Minimum energy possible for a
     particle entrapped in a one dimensional box, eV
5 n = [1 2 3 4];     // Principal quantum number for K,
     L, M and N states
6 printf("\nThe next three energies which the particle
     can have are:");
```

```
for i = 2:1:4
    printf("\nE%d = %d eV", i, ceil(i^2*E1));
end

// Result
// The next three energies which the particle can
   have are:
// E2 = 80 eV
// E3 = 180 eV
// E4 = 320 eV
```

Scilab code Exa 5.31 Probability of finding the particle within a given interval

```
// Scilab Code Ex5.31: Page-301 (2008)
clc; clear;
delta_x = 4;    // Interval at the centre of the box
    at which the probability is to be found out,
   angstrom
l = 10;    // Width of one dimensional infinite
   height box, angstrom
P = 2*delta_x/l;    // Probability of finding the
   particle within 4 angstrom interval
printf("\nThe probability of finding the particle
   within the %d angstrom interval at the centre of
   the box = %3.1f", delta_x, P);

// Result
// The probability of finding the particle within
   the 4 angstrom interval at the centre of the box
   = 0.8
```

Scilab code Exa 5.32 Probability of finding a particle within given range of 1D box for different energy states

```
1 // Scilab Code Ex5.32: Page-301 (2008)
2 clc; clear;
3 L = 1;    // Assume the length of the box to be
     unity, m
4 L1 = 0.4*L;    // Lower limit, m
5 L2 = 0.6*L;    // Upper limit, m
6 x = (L1+L2)/2;    // Mean position of particle, m
7 delta_x = L2 - L1;    // Uncertainty in position of
     the particle, m
8 for n = 1:1:3
9     P = 2/L*sin(n*%pi*x/L)^2;    // Probability
        density, per m
10    printf("\nFor n = %d, the probability, P = %3.1f
        ", n, P*delta_x);
11 end
12
13 // Result
14 // For n = 1, the probability, P = 0.4
15 // For n = 2, the probability, P = 0.0
16 // For n = 3, the probability, P = 0.4
```

Chapter 6

Classical Statistics and Quantum Statistics

Scilab code Exa 6.1 Probability of distribution of distinguishable particles

```
1 // Scilab Code Ex6.1: Page-345 (2008)
2 clc; clear;
3 n = 14;     // Total number of particles
4 C = 2;     // Total number of compartments
5 N_micro = C^n;     // Total number of microstates
6 n1 = [10 7 14];     // Set of number of particles in
     first compartment
7 n2 = [4 7 0];     // Set of number of particles in
     second compartment
8 for i = 1:1:3
9     W = factorial(n1(i) + n2(i))/(factorial(n1(i))*
        factorial(n2(i)));
10    P = W/N_micro;
11    printf("\nThe probability of microstate (%d, %d)
         = %8.6f", n1(i), n2(i), P);
12 end
13
14 // Result
```

```
15 // The probability of microstate (10, 4) = 0.061096
16 // The probability of microstate (7, 7) = 0.209473
17 // The probability of microstate (14, 0) = 0.000061
```

Scilab code Exa 6.6 Most probable distribution for total energy

```
1  // Scilab Code Ex6.6: Page-348 (2008)
2  clc; clear;
3  MAX = 10;
4  // Look for all the possible set of values for n1,
      n2 and n3
5  printf("\nThe most probable distribution is for ");
6  for i = 0:1:5
7      for j = 0:1:5
8          for k = 0:1:5
9        // Check for the condition and avoid
             repetition of set of values
10               if ((i + j + k) == 5) & ((j+2*k) == 3)
                    then
11                   W = factorial(i + j + k)/(factorial(
                        i)*factorial(j)*factorial(k));
12                   if W > MAX then
13                       printf("\nn1 = %d, n2 = %d and n3
                             = %d", i, j, k);
14                   end
15               end
16           end
17       end
18 end
19
20 // Result
21 // The most probable distribution is for
22 // n1 = 3, n2 = 1 and n3 = 1
```

Scilab code Exa 6.8 Probability for a Maxwell Boltzmann system to be in given states

```
1 // Scilab Code Ex6.8: Page-349 (2008)
2 clc; clear;
3 k = 1.38e-016;     // Boltzmann constant, erg/K
4 T = 100;     // Given temperature, K
5 E1 = 0;     // Energy of the first state, erg
6 E2 = 1.38e-014;     // Energy of the second state,
    erg
7 E3 = 2.76e-014;     // Energy of the third state, erg
8 g1 = 2, g2 = 5, g3 = 4;     // Different ways of
    occuring for E1, E2 and E3 states
9 P1 = g1*exp(-E1/(k*T));     // Probability of
    occurence of state E1
10 P2 = g2*exp(-E2/(k*T));     // Probability of
    occurence of state E2
11 P3 = g3*exp(-E3/(k*T));     // Probability of
    occurence of state E3
12 PE_3 = P3/(P1+P2+P3);     // Probability for the
    system to be in any one microstates of E3
13 P0 = P1/(P1+P2+P3);     // Probability for the system
     to be in ground state
14 printf("\nThe probability for the system to be in
    any one microstates of E3 = %6.4f", PE_3);
15 printf("\nThe probability for the system to be in
    ground state = %5.3f", P0);
16
17 // Result
18 // The probability for the system to be in any one
    microstates of E3 = 0.1236
19 // The probability for the system to be in ground
    state = 0.457
```

Scilab code Exa 6.9 Number of microstates in the given macostate of a Fermi Dirac system

```
1 // Scilab Code Ex6.9: Page-350 (2008)
2 clc; clear;
3 g1 = 6, g2 = 8;     // Total number of cells in the
    first and the second compartments respectively
4 n1 = 2, n2 = 3;     // Given number of cells in the
    first and the second compartments respectively
    for given macrostate
5 W_23 = factorial(g1)/(factorial(n1)*factorial(g1 -
    n1))*factorial(g2)/(factorial(n2)*factorial(g2 -
    n2));     // Total number of microstates in the
    macrostate (2, 3)
6 printf("\nThe total number of microstates in the
    macrostate (%d, %d) = %d", n1, n2, W_23);
7
8 // Result
9 // The total number of microstates in the macrostate
     (2, 3) = 840
```

Scilab code Exa 6.10 Number of microstates formed by particles obeying Fermi Dirac statistics

```
1 // Scilab Code Ex6.10: Page-350 (2008)
2 clc; clear;
3 g1 = 8, g2 = 10;     // Total number of cells in the
    first and the second compartments respectively
4 n1 = 3, n2 = 4;     // Given number of cells in the
    first and the second compartments respectively
    for given macrostate
```

```
5 W_34 = factorial(g1)/(factorial(n1)*factorial(g1 -
    n1))*factorial(g2)/(factorial(n2)*factorial(g2 -
    n2));    // Total number of microstates in the
    macrostate (3, 4)
6 printf("\nThe total number of microstates in the
    macrostate (%d, %d) = %d", n1, n2, W_34);
7
8 // Result
9 // The total number of microstates in the macrostate
     (3, 4) = 11760
```

Scilab code Exa 6.11 Fermi energy and internal energy for metallic silver at 0 K

```
1 // Scilab Code Ex6.11: Page-351 (2008)
2 clc; clear;
3 h = 6.6e-034;    // Planck's constant, Js
4 m = 9.1e-031;    // Mass of an electron, kg
5 e = 1.6e-019;    // Energy equivalent of 1 eV, J
6 rho = 10.5;    // Density of silver, g/cc
7 A = 108;    // Atomic weight of Ag, g/mole
8 N_A = 6.023e+023;    // Avogadro's number
9 E_F0 = h^2/(8*m)*(3*N_A*rho*1e+006/(%pi*A))^(2/3);
       // Fermi energy of silver at 0 K, J
10 U = 3/5*(N_A*rho*1e+006/A)*E_F0;    // Internal
    energy of the electron gas per unit volume at 0 K
    , J/metre-cube
11 printf("\nThe Fermi energy of silver at 0 K = %3.1f
    eV", E_F0/e);
12 printf("\nThe internal energy of the electron gas
    per unit volume at 0 K = %4.2e J/cubic-metre", U)
    ;
13
14 // Result
15 // The Fermi energy of silver at 0 K = 5.5 eV
```

```
16 // The internal energy of the electron gas per unit
   volume at 0 K = 3.07e+010 J/cubic-metre
```

Scilab code Exa 6.12 Number of conduction electrons per cc in silver at 0 K

```
1 // Scilab Code Ex6.12: Page-351 (2008)
2 clc; clear;
3 h = 6.6e-034;     // Planck's constant, Js
4 m = 9.1e-031;     // Mass of an electron, kg
5 e = 1.6e-019;     // Energy equivalent of 1 eV, J
6 E_F0 = 5.48;      // Fermi energy of silver at 0 K,
   eV
7 N_bar = (8*m/h^2)^(3/2)*%pi/3*(E_F0*e)^(3/2);    //
   Number density of conduction electrons in silver
   at 0 K, per cc
8 printf("\nThe number density of conduction electrons
    in silver at 0 K = %3.1e per cc", N_bar*1e-006);
9
10 // Result
11 // The number density of conduction electrons in
   silver at 0 K = 5.9e+022 per cc
```

Scilab code Exa 6.13 Fermi energy of conduction electrons in cesium

```
1 // Scilab Code Ex6.13: Page-351 (2008)
2 clc; clear;
3 h = 6.6e-034;     // Planck's constant, Js
4 m = 9.1e-031;     // Mass of an electron, kg
5 e = 1.6e-019;     // Energy equivalent of 1 eV, J
6 E_F0_Be = 14.44      // Fermi energy of Be at 0 K,
   eV
```

```
N_bar_Be = 24.2e+022;    // Number density of
   conduction electrons in Be at 0 K, per cc
N_bar_Cs = 0.91e+022;    // Number density of
   conduction electrons in Cs at 0 K, per cc
E_F0_Cs = E_F0_Be*(N_bar_Cs/N_bar_Be)^(2/3);    //
   Fermi energy of conduction electrons in cesium,
   eV
printf("\nThe Fermi energy of conduction electrons
   in cesium = %5.3f eV", E_F0_Cs);

// Result
// The Fermi energy of conduction electrons in
   cesium = 1.621 eV
// The answer is given wrongly in the textbook
```

Chapter 7

Classical Statistics and Quantum Statistics

Scilab code Exa 7.1 Atomic packing fractions of SC FCC and BCC unit cells

```
1 // Scilab Code Ex7.1: Page-376 (2008)
2 clc; clear;
3 a = poly(0, 'a');     // Lattice parameter for a
     cubic unit cell, m
4 // For simple cubic cell
5 n = 1;     // Number of atoms per simple cubic unit
     cell
6 r = a/2;     // Atomic radius for a simple cubic cell
     , m
7 f = pol2str(int(numer(n*4/3*%pi*r^3/a^3)*100));
     // Atomic packing fraction for a simple cubic
     cell
8 printf("\nFor simple cubic cell, f = %s percent", f)
     ;
9 // For face centered cubic cell
10 n = 2;     // Number of atoms per face centered cubic
     unit cell
11 r = sqrt(3)/4*a;     // Atomic radius for a face
```

```
    centered cubic cell, m
12 f = pol2str(int(numer(n*4/3*%pi*r^3/a^3)*100));
    // Atomic packing fraction for a face centered
    cubic cell
13 printf("\nFor face centered cubic cell, f = %s
    percent", f);
14 // For body centered cubic cell
15 n = 4;    // Number of atoms per body centered cubic
     unit cell
16 r = a/(2*sqrt(2));    // Atomic radius for a body
    centered cubic cell, m
17 f = pol2str(int(numer(n*4/3*%pi*r^3/a^3)*100));
    // Atomic packing fraction for a body centered
    cubic cell
18 printf("\nFor body centered cubic cell, f = %s
    percent", f);
19
20 // Result
21 // For simple cubic cell, f =  52 percent
22 // For face centered cubic cell, f =  68 percent
23 // For body centered cubic cell, f =  74 percent
```

Scilab code Exa 7.3 Distance between two adjacent atoms in the NaCl

```
1 // Scilab Code Ex7.3: Page-377 (2008)
2 clc; clear;
3 M = 58.46;    // Gram atomic mass of NaCl, g/mole
4 N = 6.023e+023;    // Avogadro's number
5 rho = 2.17;    // Density of NaCl, g/cc
6 m = M/N;    // Mass of each NaCl molecule, g
7 n = rho/m;    // Number of NaCl molecules per unit
    volume, molecules/cc
8 N = 2*n;    // Number of atoms per unit volume,
    atoms/cc
9 a = (1/N)^(1/3);    // Distance between two adjacent
```

```
    atoms in the NaCl, cm
printf("\nThe distance between two adjacent atoms in
    the NaCl = %4.2f angstrom", a/1e-008);

// Result
// The distance between two adjacent atoms in the
   NaCl = 2.82 angstrom
```

Scilab code Exa 7.4 Type of unit cell of Cs

```
// Scilab Code Ex7.4: Page-377 (2008)
clc; clear;
function p = find_cell_type(x)
    if x == 1 then
        p = 'simple cubic';
    end
    if x == 2 then
        p = 'body centered';
    end
    if x == 4 then
        p = 'face centered';
    end
endfunction
M = 130;    // Gram atomic weight of Cs, g/mole
N = 6.023e+023;    // Avogadro's number
rho = 2;    // Density of Cs, g/cc
a = 6e-008;    // Distance between two adjacent
   atoms in the Cs, cm
m = M/N;    // Mass of each Cs atom, g
x = rho*a^3*N/M;    // Number of Cs atoms in cubic
   unit cell
c_type = find_cell_type(int(x));    // Call function
    to determine the type of cell
printf("\nThe cubic unit cell of Cs is %s.", c_type)
   ;
```

```
22
23 // Result
24 // The cubic unit cell of Cs is body centered.
```

Scilab code Exa 7.5 Miller indices of given planes

```
1 // Scilab Code Ex7.5: Page-378 (2008)
2 clc; clear;
3 m = 2; n = 3; p = 6; // Coefficients of intercepts
     along three axes
4 m_inv = 1/m;          // Reciprocate the first
     coefficient
5 n_inv = 1/n;          // Reciprocate the second
     coefficient
6 p_inv = 1/p;          // Reciprocate the third
     coefficient
7 mul_fact = double(lcm(int32([m,n,p]))); // Find l.c.
    m. of m,n and p
8 m1 = m_inv*mul_fact;     // Clear the first fraction
9 m2 = n_inv*mul_fact;     // Clear the second fraction
10 m3 = p_inv*mul_fact;    // Clear the third fraction
11 printf("\nThe required miller indices are : (%d %d
    %d) ", m1,m2,m3);
12
13 // Result
14 // The required miller indices are : (3 2 1)
```

Scilab code Exa 7.6 Meaning of hkl notation of planes

```
1 // Scilab Code Ex7.6: Page-378 (2008)
2 clc; clear;
3 // For first set (3, 2, 2)
```

```
m = 3; n = 2; p = 2; // Coefficients of intercepts along three axes
m_inv = 1/m;         // Reciprocate the first coefficient
n_inv = 1/n;         // Reciprocate the second coefficient
p_inv = 1/p;         // Reciprocate the third coefficient
mul_fact = double(lcm(int32([m,n,p]))); // Find l.c.m. of m,n and p
m1 = m_inv*mul_fact;     // Clear the first fraction
m2 = n_inv*mul_fact;     // Clear the second fraction
m3 = p_inv*mul_fact;     // Clear the third fraction
printf("\nThe plane (%d %d %d) has intercepts %da, %db and %dc on the three axes.", m, n, p, m1, m2, m3);
// For second set (1 1 1)
m = 1; n = 1; p = 1; // Coefficients of intercepts along three axes
m_inv = 1/m;         // Reciprocate the first coefficient
n_inv = 1/n;         // Reciprocate the second coefficient
p_inv = 1/p;         // Reciprocate the third coefficient
mul_fact = double(lcm(int32([m,n,p]))); // Find l.c.m. of m,n and p
m1 = m_inv*mul_fact;     // Clear the first fraction
m2 = n_inv*mul_fact;     // Clear the second fraction
m3 = p_inv*mul_fact;     // Clear the third fraction
printf("\nThe plane (%d %d %d) has intercepts a, b and c on the three axes.", m, n, p);

// Result
// The plane (3 2 2) has intercepts 2a, 3b and 3c on the three axes.
// The plane (1 1 1) has intercepts a, b and c on the three axes.
```

Scilab code Exa 7.9 Lengths of intercepts along y and z axis

```
1 // Scilab Code Ex7.9: Page-379 (2008)
2 clc; clear;
3 h = 2; k = 3; l = 1; // Miller indices of the set of
      planes
4 p = 1/h;          // Reciprocate h
5 q = 1/k;          // Reciprocate k
6 r = 1/l;          // Reciprocate l
7 lx = 1.2;     // Intercept cut by plane along x-axis,
      angstrom
8 a = 1.2, b = 1.8, c = 2;     // Primitives of the
    crystal, angstrom
9 mul_fact = double(lcm(int32([h, k, l]))); // Find l.
    c.m. of h, k and l
10 pa = mul_fact*p*a;
11 qb = mul_fact*q*b;
12 rc = mul_fact*r*c;
13 ly = lx*qb/pa;     // Length of intercept along y-
    axis
14 lz = lx*rc/pa;     // Length of intercept along z-
    axis
15 printf("\nThe length of intercept along y-axis = %3
    .1f angstrom", ly);
16 printf("\nThe length of intercept along z-axis = %3
    .1f angstrom", lz);
17
18 // Result
19 // The length of intercept along y-axis = 1.2
    angstrom
20 // The length of intercept along z-axis = 4.0
    angstrom
```

Scilab code Exa 7.10 Interplanar spacing for a set of planes in a cubic lattice

```
1 // Scilab Code Ex7.10: Page-380 (2008)
2 clc; clear;
3 h = 3; k = 2; l = 1; // Miller Indices for planes in
      a cubic crystal
4 a = 4.21D-10;     // Interatomic spacing, m
5 d = a/(h^2+k^2+l^2)^(1/2);  // The interplanar
     spacing for cubic crystals, m
6 printf("\nThe interplanar spacing between
     consecutive (321) planes = %3.1e m", d);
7
8 // Result
9 // The interplanar spacing between consecutive (321)
      planes = 1.1e-010 m
```

Scilab code Exa 7.11 Determining Planck constant from given set of X ray data

```
1 // Scilab Code Ex7.11: Page-380 (2008)
2 clc; clear;
3 e = 1.6e-019;     // The energy equivalent of 1 eV, J
4 c = 3e+008;     // Speed of light in vacuum, m/s
5 V = [30 44 50 200];     // Operating voltages of X
     ray, kV
6 lambda_min = [0.414 0.284 0.248 0.062];     //
     Minimum wavelengths of emitted continuous X rays,
      angstrom
7 for i = 1:1:4
8     h = e*V(i)*1e+003*lambda_min(i)*1e-010/c;     //
        Planck's constant, Js
```

```
9       printf("\nFor V = %d kV and lambda_min = %5.3f
            angstrom, h = %4.2e Js", V(i), lambda_min(i),
            h);
10  end
11
12  // Result
13  // For V = 30 kV and lambda_min = 0.414 angstrom, h
        = 6.62e-034 Js
14  // For V = 44 kV and lambda_min = 0.284 angstrom, h
        = 6.66e-034 Js
15  // For V = 50 kV and lambda_min = 0.248 angstrom, h
        = 6.61e-034 Js
16  // For V = 200 kV and lambda_min = 0.062 angstrom, h
        = 6.61e-034 Js
```

Scilab code Exa 7.12 Maximum speed of striking electron and the shortest wavelength of X ray produced

```
1   // Scilab Code Ex7.12: Page-381 (2008)
2   clc; clear;
3   e = 1.6e-019;      // The energy equivalent of 1 eV, J
4   m = 9.11e-031;      // Rest mass of an electron, kg
5   h = 6.62e-034;      // Planck's constant, Js
6   c = 3e+008;      // Speed of light in vacuum, m/s
7   V = [20 100];      // Operating voltages of X ray, kV
8   for i = 1:1:2
9       v = sqrt(2*e*V(i)*1e+003/m);      // Maximum
            striking speed of the electron, m/s
10      lambda_min = c*h/(e*V(i)*1e+003*1e-010);      //
            Minimum wavelength of emitted continuous X
            rays, angstrom
11      printf("\nFor V = %d kV:", V(i));
12      printf("\nThe maximum striking speed of the
            electron = %5.2e m/s", v);
13      printf("\nThe minimum wavelength of emitted
```

```
        continuous X rays = %5.3f angstrom\n",
        lambda_min);
14 end
15
16 // Result
17 // For V = 20 kV:
18 // The maximum striking speed of the electron = 8.38
     e+007 m/s
19 // The minimum wavelength of emitted continuous X
     rays = 0.621 angstrom
20 //
21 // For V = 100 kV:
22 // The maximum striking speed of the electron = 1.87
     e+008 m/s
23 // The minimum wavelength of emitted continuous X
     rays = 0.124 angstrom
24 // There are small variation in the answers as
     approximations are used in the text
```

Scilab code Exa 7.13 Interatomic spacing using Bragg relation

```
1 // Scilab Code Ex7.13: Page-381 (2008)
2 clc; clear;
3 n = 1;     // Order of diffraction
4 lambda = 1.75e-010;     // Wavelength of X rays, m
5 h = 1, k = 1, l = 1;     // Miller indices for the
     set of planes
6 theta = 30;     // Bragg's angle, degree
7 // As from Bragg's law, 2*d*sind(theta) = n*lambda
     and d = a/sqrt(h^2+k^2+l^2). solving for a we
     have
8 a = sqrt(h^2+k^2+l^2)*n*lambda/(2*sind(theta)*1e
     -010);     // Interatomic spacing of the crystal,
     angstrom
9 printf("\nThe interatomic spacing of the crystal =
```

```
    %5.3f angstrom", a);
10
11 // Result
12 // The interatomic spacing of the crystal = 3.031
    angstrom
```

Scilab code Exa 7.14 Value of Planck constant from Bragg relation

```
1  // Scilab Code Ex7.14: Page-382 (2008)
2  clc; clear;
3  e = 1.6e-019;     // The energy equivalent of 1 eV, J
4  c = 3e+008;     // Speed of light in vacuum, m/s
5  n = 1;     // Order of diffraction
6  d = 2.82e-010;     // Interplanar spacing, m
7  V = 9.1e+003;     // Operating voltage of X rays
8  theta = 14;     // Bragg's angle, degree
9  lambda = 2*d*sind(theta)/n;     // Wavelength of X
    rays, m
10 nu = c/lambda;     // Frequency of X rays, Hz
11 h = e*V/nu;     // Planck's constant, Js
12 printf("\nThe value of Planck constant, h = %4.2e Js
    ", h);
13
14 // Result
15 // The value of Planck constant, h = 6.62e-034 Js
```

Scilab code Exa 7.15 Diffraction of X rays from a crystal

```
1  // Scilab Code Ex7.15: Page-382 (2008)
2  clc; clear;
3  e = 1.6e-019;     // The energy equivalent of 1 eV, J
4  c = 3e+008;     // Speed of light in vacuum, m/s
5  lambda = 0.5e-010;     // Wavelength of X rays, m
```

```
theta = 5;     // Bragg's angle, degree
n = 1;     // Order of diffraction
d = n*lambda/(2*sind(theta)*1e-010);     //
   Interplanar spacing, angstrom
n = 2;     // Ordr of diffraction
theta1 = asind(n*lambda/(2*d*1e-010));     // Angle
   at which the second maximum occur, degree
printf("\nThe spacing between adjacent planes of the
    crystal = %4.2f angstrom", d);
printf("\nThe angle at which the second maximum
   occur = %5.2f degree", theta1);

// Result
// The spacing between adjacent planes of the
   crystal = 2.87 angstrom
// The angle at which the second maximum occur =
   10.04 degree
```

Scilab code Exa 7.16 Wavelength of X rays from grating space of the rock salt

```
// Scilab Code Ex7.16: Page-383 (2008)
clc; clear;
M = 58.5     // Gram atomic mass of NaCl, kg/mole
N = 6.023e+026;     // Avogadro's number per kmol
rho = 2.17e+003;     // Density of NaCl, kg/metre-
   cube
m = M/N;     // Mass of each NaCl molecule, g
V = m/rho;     // Volume of each NaCl molecule, metre
   -cube
d = (V/2)^(1/3)/1e-010;     // Atomic apacing in the
   NaCl crystal, angstrom
theta = 26;     // Bragg's angle, degree
n = 2;     // Order of diffraction
lambda = 2*d*sind(theta)/n;     // Wavelength of X
```

```
    rays, m
12 printf("\nThe grating spacing of rock salt = %4.2f
    angstrom", d);
13 printf("\nThe wavelength of X rays = %4.2f angstrom"
    , lambda);
14
15 // Result
16 // The grating spacing of rock salt = 2.82 angstrom
17 // The wavelength of X rays = 1.24 angstrom
```

Scilab code Exa 7.17 Diffraction of X rays by the calcite crystal

```
1  // Scilab Code Ex7.17: Page-383 (2008)
2  clc; clear;
3  d = 3.02945e-010;     // Atomic apacing in the
    calcite crystal, m
4  lambda_alpha = 0.563e-010;     // Wavelength of the K
    -alpha line of Ag, m
5  n = 1;     // Order of diffraction
6  theta = asind(n*lambda_alpha/(2*d));     // Angle of
    reflection for the first order, degree
7  theta_max = 90;     // Angle of reflection for the
    highest order, degree
8  n = 2*d*sind(theta_max)/lambda_alpha;     // The
    highest order for which the line may be observed
9  printf("\nThe angle of reflection for the first
    order = %4.2f degree", theta);
10 printf("\nThe highest order for which the line may
    be observed = %d", n);
11
12 // Result
13 // The angle of reflection for the first order =
    5.33 degree
14 // The highest order for which the line may be
    observed = 10
```

Scilab code Exa 7.18 Interatomic spacing for given crystal planes

```
1 // Scilab Code Ex7.18: Page-384 (2008)
2 clc; clear;
3 lambda = 1.8e-010;     // Wavelength of the X rays, m
4 n = 1;     // Order of diffraction
5 theta = 60;     // Angle of diffraction for the first
      order, degree
6 d = n*lambda/(2*sind(theta));     // Interplanar
     spacing, m
7 // Since for a simple cubic lattice, d_111 = d = a/
     sqrt(3), solving for a
8 a = sqrt(3)*d;     // The interatomic spacing for the
      given crystal planes, m
9 printf("\nThe interatomic spacing for the given
     crystal planes, a = %3.1f angstrom", a/1e-010);
10
11 // Result
12 // The interatomic spacing for the given crystal
     planes, a = 1.8 angstrom
```

Scilab code Exa 7.19 Smallest angle between the crystal plane and the X ray beam

```
1 // Scilab Code Ex7.19: Page-384 (2008)
2 clc; clear;
3 function [d, m] = deg2degmin(theta)
4     d = int(theta);
5     m = (theta-d)*60;
6 endfunction
7 h = 6.626e-034;     // Planck's constant, Js
```

```
e = 1.6e-019;     // The energy equivalent of 1 eV, J
c = 3e+008;     // Speed of light in vacuum, m/s
V = 50e+003;     // Operating voltage of X ray, V
lambda_min = h*c/(e*V);     // Minimum wavelength of
   emitted continuous X rays, angstrom
n = 1;     // Order of diffraction
d = 3.02945e-010;     // Interplanar spacing, m
theta = asind(n*lambda_min/(2*d));     // The
   smallest angle between the crystal plane and the
   X ray beam, degree
[deg , m] = deg2degmin(theta);
printf("\nThe smallest angle between the crystal
   plane and the X ray beam = %d degree %d min", deg
   , m);

// Result
// The smallest angle between the crystal plane and
   the X ray beam = 2 degree 21 min
```

Chapter 8

Laser and Fibre Optics

Scilab code Exa 8.1 Image produced by laser beam

```
1 // Scilab Code Ex8.1: Page-397 (2008)
2 clc; clear;
3 lambda = 6000e-008;      // Wavelength of the lase
     beam, cm
4 P = 10e-003;     // Power of the laser beam, W
5 theta = 1.5e-004;    // Angular spread of laser beam,
      rad
6 f = 10;     // Focal length of the lens, cm
7 r = f*theta;     // Radius of the image, cm
8 rho = P/(%pi*r^2*1e+003);  // Power density of the
     image, kW/Sq.cm
9 L_w = lambda/(theta/10);      // Coherence width, mm
10 printf("\nThe radius of the image = %3.1e cm", r);
11 printf("\nThe power density of the image = %3.1f kW/
     Sq.cm", rho);
12 printf("\nThe coherence width = %d mm", L_w);
13
14 // Result
15 // The radius of the image = 1.5e-03 cm
16 // The power density of the image = 1.4 kW/Sq.cm
17 // The coherence width = 4 mm
```

Scilab code Exa 8.2 Pumping energy required for He Ne laser transition

```
1 // Scilab Code Ex8.2: Page-398 (2008)
2 clc; clear;
3 lambda = 632.8e-009;      // Wavelength of the lase
    beam, cm
4 E_2P = 15.2e-019;     // Energy of 2P level, J
5 h = 6.626e-034;      // Planck's constant, Js
6 c = 3e+008;      // Speed of light, m/s
7 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
8 E_Pump = E_2P + h*c/lambda; // The required pumping
    energy, J
9 printf("\nThe pumping energy required for He Ne
    laser transition = %5.2f eV", E_Pump/e);
10
11 // Result
12 // The pumping energy required for He Ne laser
    transition = 11.46 eV
```

Scilab code Exa 8.3 Wavelength of radiation emitted at room temperature

```
1 // Scilab Code Ex8.3: Page-398 (2008)
2 clc; clear;
3 h = 6.626e-034;      // Planck's constant, Js
4 c = 3e+008;      // Speed of light, m/s
5 T = 27+273;      // Room temperature, K
6 k = 1.38e-023;   // Boltzmann constant, J/mol/K
7 lambda = h*c/(k*T); // Wavelength of radiation
    mitted at room temperature, m
8 printf("\nThe wavelength of radiation mitted at room
     temperature = %3.1e m", lambda);
```

```
9
10 // Result
11 // The wavelength of radiation mitted at room
   temperature = 4.8e-05 m
```

Scilab code Exa 8.4 Refractive index of the cladding in an optical fibre

```
1 // Scilab Code Ex8.4: Page-398 (2008)
2 clc; clear;
3 NA = 0.5;   // Numerical aperture of the optical
   fibre
4 n1 = 1.54;  // Refractive index of the core material
5 n2 = sqrt(n1^2-NA^2);   // Refractive index of the
   cladding in an optical fibre
6 printf("\nThe refractive index of the cladding in
   the optical fibre = %4.2f", n2);
7
8 // Result
9 // The refractive index of the cladding in the
   optical fibre = 1.46
```

Scilab code Exa 8.5 Numerical aperture and acceptance angle of the optical fibre

```
1 // Scilab Code Ex8.5: Page-398 (2008)
2 clc; clear;
3 n1 = 1.51;  // Refractive index of the core material
4 n2 = 1.47;   // Refractive index of the cladding
5 NA = sqrt(n1^2-n2^2);   // Numerical aperture of the
    optical fibre
6 n0 = 1;     // Refractive index of air
7 theta_a = asin(NA/n0);  // Acceptance angle of the
   optical fibre, rad
```

```
8  printf("\nThe numerical aperture of the optical
       fibre = %6.4f", NA);
9  printf("\nThe acceptance angle of the optical fibre
       = %4.2f degrees", theta_a*180/3.14);
10
11 // Result
12 // The numerical aperture of the optical fibre =
       0.3453
13 // The acceptance angle of the optical fibre = 20.21
        degrees
```

Chapter 9

Nuclear Physics

Scilab code Exa 9.1.1 Binding energy per nucleon for Ni

```
1 // Scilab Code Ex9.1.1:Page-411 (2008)
2 clc; clear;
3 u = 931.508;     // Energy equivalent of 1 amu, MeV
4 Z = 28;     // Atomic number of ni-64
5 A = 64;     // Mass number of Ni-64
6 m_p = 1.007825;      // Mass of a proton, u
7 m_n = 1.008665;     // Mass of a neutron, u
8 M_Ni = 63.9280;     // Atomic mass of Ni-64 nucleus,
    u
9 delta_m = Z*m_p + (A-Z)*m_n - M_Ni;     // Mass
    difference, u
10 BE = delta_m*u;     // Binding energy of Ni-64
    nucleus, MeV
11 BE_bar = BE/A;     // Binding energy per nucleon of
    Ni-64 nucleus, MeV
12 printf("\nThe binding energy per nucleon for Ni-64
    nucleus = %4.2f MeV/nucleon", BE_bar);
13
14 // Result
15 // The binding energy per nucleon for Ni-64 nucleus
    = 8.78 MeV/nucleon
```

Scilab code Exa 9.1.2 Binding energy per nucleon for deutron

```
1 // Scilab Code Ex9.1.2:Page-411 (2008)
2 clc; clear;
3 e = 1.6e-013;     // Energy equivalent of 1 MeV, J
4 m_p = 1.672e-027;      // Mass of a proton, kg
5 m_n = 1.675e-027;     // Mass of a neutron, kg
6 M_D = 3.343e-027;     // Mass of a deutron, kg
7 c = 3.00e+008;     // Speed of light in vacuum, m/s
8 delta_m = m_p + m_n - M_D;     // Mass defect, kg
9 E_B = delta_m*c^2/e;     // Binding energy for the
    deutron, MeV
10 BE_bar = E_B/2;     // Binding energy per nucleon for
     the deutron, MeV
11 printf("\nThe binding energy per nucleon for the
    deutron = %5.3f MeV/nucleon", BE_bar);
12
13 // Result
14 // The binding energy per nucleon for the deutron =
    1.125 MeV/nucleon
```

Scilab code Exa 9.1.3 Packing fraction and binding energy per nucleon for oxygen

```
1 // Scilab Code Ex9.1.3:Page-411 (2008)
2 clc; clear;
3 u = 931.508;     // Energy equivalent of 1 amu, MeV
4 Z = 8;     // Atomic number of O-16
5 A = 16;     // Mass number of O-16
6 m_p = 1.008142;      // Mass of a proton, u
7 m_n = 1.008982;     // Mass of a neutron, u
```

```
8  M_O = 15.994915;     // Atomic mass of O-16 nucleus,
     u
9  delta_m = Z*m_p + (A-Z)*m_n - M_O;     // Mass
     difference, u
10 BE = delta_m*u;     // Binding energy of O-16 nucleus
     , MeV
11 BE_bar = BE/A;     // Binding energy per nucleon of O
     -16 nucleus, MeV
12 delta_m = abs(M_O - A);     // Mass difference, u
13 PF = delta_m/A;     // Packing fraction for O-16
     nucleus, u
14 printf("\nThe binding energy per nucleon for O-16
     nucleus = %4.2f MeV/nucleon", BE_bar);
15 printf("\nThe packing fraction for O-16 nucleus = %5
     .3e u", PF);
16
17 // Result
18 // The binding energy per nucleon for O-16 nucleus =
      8.27 MeV/nucleon
19 // The packing fraction for O-16 nucleus = 3.178e
     -004 u
```

Scilab code Exa 9.1.4 Atomic mass of neon

```
1  // Scilab Code Ex9.1.4: Page-411 (2008)
2  clc; clear;
3  u = 931.508;     // Energy equivalent of 1 amu, MeV
4  Z = 10;     // Atomic number of Ne-20
5  A = 20;     // Mass number of Ne-0
6  m_p = 1.007825;     // Mass of a proton, u
7  m_n = 1.008665;     // Mass of a neutron, u
8  BE = 160.64;     // Binding energy of Ne-20 nucleus,
     MeV
9  M = Z*m_p + (A-Z)*m_n + Z*0.51/u - BE/u;     //
     Atomic mass of Ne-20 nucleus, u
```

```
10 printf("\nThe atomic mass of Ne = %7.4f a.m.u", M);
11
12 // Result
13 // The atomic mass of Ne = 19.9979 a.m.u
```

Scilab code Exa 9.2.1 Average number of photons pe cubic metre in a monochromatic beam

```
1 // Scilab Code Ex9.2.1: Page-414 (2008)
2 clc; clear;
3 h = 6.63e-034;     // Planck's constant, Js
4 c = 3.00e+008;     // Speed of light in vacuum, m/s
5 I = 1e+004;     // Intensity of monochromatic beam, W
    /Sq.m
6 nu = 1e+004;     // Frequency of monochromatic beam,
    Hz
7 n = I/(h*nu*c);     // Average number of photons per
    cubic metre, photons/metre-cube
8 printf("\nThe average number of photons in the
    monochromatic beam of radiation = %4.2e photons/
    metre-cube", n);
9
10 // Result
11 // The average number of photons in the
    monochromatic beam of radiation = 5.03e+024
    photons/metre-cube
```

Scilab code Exa 9.2.2 Average number of photons pe cubic metre in a monochromatic beam

```
1 // Scilab Code Ex9.2.2: : Page-414 (2008)
2 clc; clear;
3 h = 6.63e-034;     // Planck's constant, Js
```

```
4 c = 3.00e+008;     // Speed of light in vacuum, m/s
5 I = 1e+004;     // Intensity of monochromatic beam, W
    /Sq.m
6 nu = 1e+004;     // Frequency of monochromatic beam,
    Hz
7 n = I/(h*nu*c);     // Average number of photons per
    cubic metre, photons/metre-cube
8 printf("\nThe average number of photons in the
    monochromatic beam of radiation = %4.2e photons/
    metre-cube", n);
9
10 // Result
11 // The average number of photons in the
    monochromatic beam of radiation = 5.03e+024
    photons/metre-cube
```

Scilab code Exa 9.2.3 Photoelectric effect with silver

```
1 // Scilab Code Ex9.2.3: Page-414 (2008)
2 clc; clear;
3 h = 6.63e-034;     // Planck's constant, Js
4 c = 3.00e+008;     // Speed of light in vacuum, m/s
5 e = 1.6e-019;     // Energy equivalent of 1 eV, J
6 m_e = 9.1e-031;     // Rest mass of an electron, kg
7 lambda0 = 2762e-010;     // Thereshold wavelength of
    silver, m
8 lambda = 2000e-010;     // Wavelength of ultraviolet
    rays, m
9 E_max = h*c*(1/lambda - 1/lambda0);     // Maximum
    kinetic energy of the ejected electrons from
    Einstein's photoelectric equation, J
10 // As E_max = 1/2*m_e*v^2, solving for v
11 v_max = sqrt(2*E_max/m_e);     // Maximum velocity of
     the photoelectrons, m/s
12 V0 = E_max/e;     // Stopping potential for the
```

```
    electrons , V
13 printf("\nThe maximum kinetic energy of the ejected
    electrons = %5.3e J", E_max);
14 printf("\nThe maximum velocity of the photoelectrons
    = %4.2e m/s", v_max);
15 printf("\nThe stopping potential for the electrons =
    %5.3f V", V0);
16
17 // Result
18 // The maximum kinetic energy of the ejected
    electrons = 2.744e-019 J
19 // The maximum velocity of the photoelectrons = 7.77
    e+005 m/s
20 // The stopping potential for the electrons = 1.715
    V
```

Scilab code Exa 9.2.4 Work function of the metallic surface

```
1 // Scilab Code Ex9.2.4: Page-415 (2008)
2 clc; clear;
3 lambda1 = 3333e-010;     // First wavelength of the
    incident light , m
4 lambda2 = 2400e-010;     // Second wavelength of the
    incident light , m
5 c = 3e+008;     // Speed of light in free space , m/s
6 e = 1.6e-019;     // Energy equivalent of 1 eV, J
7 E1 = 0.6;     // Kinetic energy of the emitted
    photoelectrons for the first wavelength , eV
8 E2 = 2.04;     // Kinetic energy of the emitted
    photoelectrons for the second wavelength , eV
9 h = (E2 - E1)*lambda1*lambda2*e/(c*(lambda1 -
    lambda2));     // Planck's constant , Js
10 W0 = (E2*lambda2 - E1*lambda1)/(lambda1 - lambda2);
      // Work function of the metal , eV
11 printf("\nThe value of Planck constant = %3.1e Js",
```

```
    h);
12 printf("\nThe work function of the metal = %3.1f eV"
    , W0);
13
14 // Result
15 // The value of Planck constant = 6.6e-034 Js
16 // The work function of the metal = 3.1 eV
```

Scilab code Exa 9.2.5 Wavelength of the scattered photon

```
1 // Scilab Code Ex9.2.5: Page-415 (2008)
2 clc; clear;
3 c = 3e+008;     // Speed of light in free space, m/s
4 h = 6.63e-034;     // Planck's constant, Js
5 m_e = 9.11e-031;     // Rest mass of an electron, kg
6 lambda = 0.3;     // Wavelength of incident X-ray
    photon, angstrom
7 phi = 45;     // The angle of scattering, degrees
8 lambda_prime = lambda + h/(m_e*c*1e-010)*(1-cosd(phi
    ));     // The wavelength of the scattered photon,
     angstrom
9 printf("\nThe wavelength of the scattered photon =
    %6.4f angstrom", lambda_prime);
10
11 // Result
12 // The wavelength of the scattered photon = 0.3071
    angstrom
```

Scilab code Exa 9.2.6 de Broglie wavelength of the valence electron in metallic sodium

```
1 // Scilab Code Ex9.2.6: Page-416 (2008)
2 clc; clear;
```

```
3 h = 6.63e-034;     // Planck's constant, Js
4 m_e = 9.11e-031;     // Rest mass of an electron, kg
5 e = 1.6e-019;     // Energy equivalent of 1 eV, J
6 K = 3*e;     // Kinetic energy of the electron in
    metllic sodium, J
7 lambda = h/sqrt(2*m_e*K)/1e-010;     // de Broglie
    wavelength of the valence electron, angstrom
8 printf("\nThe de-Broglie wavelength of the valence
    electron = %3.1f angstrom", lambda);
9
10 // Result
11 // The de-Broglie wavelength of the valence electron
    = 7.1 angstrom
```

Scilab code Exa 9.2.7 de Broglie wavelength of a moving electron

```
1 // Scilab Code Ex9.2.7: Page-416 (2008)
2 clc; clear;
3 h = 6.63e-034;     // Planck's constant, Js
4 m = 9.11e-031;     // Rest mass of an electron, kg
5 c = 3e+008;     // Speed of light in vacuum, m/s
6 bita = 3/5;     // Boost parameter
7 v = 3/5*c;     // Spped of the electron, m/s
8 lambda = h/(m*v)*sqrt(1-bita^2);     // de Broglie
    wavelength of the electron, m
9 printf("\nThe de-Broglie wavelength of the moving
    electron = %6.4f angstrom", lambda/1e-010);
10
11 // Result
12 // The de-Broglie wavelength of the moving electron
   = 0.0323 angstrom
```

Scilab code Exa 9.2.8 Uncertainty in energy and frequency of emitted light

```
1 // Scilab Code Ex9.2.8: Page-416 (2008)
2 clc; clear;
3 h = 6.63e-034;     // Planck's constant, Js
4 h_bar = h/(2*%pi);     // Reduced Planck's constant,
    Js
5 delta_t = 1e-008;     // Time during which the
    radiation is emitted, s
6 delta_E = h_bar/delta_t;     // Minimum uncertainty
    in energy of emitted light, J
7 // As delta_E = h*delta_nu from Planck's quantum
    theory, solving for delta_nu
8 delta_nu = delta_E/h;     // Minimum uncertainty in
    frequency of emitted light, Hz
9 printf("\nThe minimum uncertainty in energy of
    emitted light = %5.3e J", delta_E);
10 printf("\nThe minimum uncertainty in frequency of
    emitted light = %4.2e Hz", delta_nu);
11
12 // Result
13 // The minimum uncertainty in energy of emitted ligh
    = 1.055e-026 J
14 // The minimum uncertainty in frequency of emitted
    ligh = 1.59e+007 Hz
```

Scilab code Exa 9.2.9 Shortest wavelength present in the radiation from an X ray machine

```
1 // Scilab Code Ex9.2.9: Page-417 (2008)
2 clc; clear;
3 h = 6.63e-034;     // Planck's constant, Js
4 c = 3e+008;     // Speed of light in free space, m/s
5 e = 1.6e-019;     // Energy equivalent of 1 eV, J
```

```
6 V = 50000;     // Accelerating potential, V
7 lambda_min = h*c/(e*V);     // The shortest
    wavelength present in the radiation from an X-ray
     machine, m
8 printf("\nThe shortest wavelength present in the
    radiation from an X-ray machine = %6.4f nm",
    lambda_min/1e-009);
9
10 // Result
11 // The shortest wavelength present in the radiation
    from an X-ray machine = 0.0249 nm
```

Scilab code Exa 9.2.11 Q value of nuclear reaction

```
1 // Scilab Code Ex9.2.11: Page-418(2008)
2 clc; clear;
3 u = 931.5;     // Energy equivalent of 1 amu, MeV
4 m_x = 4.002603;     // Mass of projectile (alpha-
    particle), u
5 m_y = 1.007825;     // Mass of emitted particle (
    proton), u
6 M_X = 14.0031;      // Mass of target nucleus (N-14),
     u
7 M_Y = 16.9994;     // Mass of daughter nucleus (O-16)
    , u
8 Q = ((m_x + M_X) - (m_y + M_Y))*u;     // Q-value of
    the reaction, MeV
9 printf("\nThe Q-value of the nuclear reaction = %5.3
    f MeV", Q);
10
11 // Result
12 // The Q-value of the nuclear reaction = -1.418 MeV
```

Scilab code Exa 9.2.12 Threshold energy for the reactions

```
// Scilab Code Ex9.2.12: Page-418(2008)
clc; clear;
u = 931.5;    // Energy equivalent of 1 amu, MeV
// First reaction
m_x = 1.007825;    // Mass of projectile (proton), u
m_y = 2.014102;    // Mass of emitted particle (deutron), u
M_X = 208.980394;    // Mass of target nucleus (Bi-209), u
M_Y = 207.979731;    // Mass of daughter nucleus (Bi-208), u
Q = ((m_x + M_X) - (m_y + M_Y))*u;    // Q-value of the reaction, MeV
Ex_threshold = -Q*(m_x + M_X)/M_X;    // The smallest value of the projectile energy, MeV
printf("\nThe threshhold energy of the reaction Bi(209,83) + p --> Bi(208,83) + d = %4.2f MeV", Ex_threshold);
// Second reaction
m_x = 4.002603;    // Mass of projectile (alpha-particle), u
m_y = 1.007825;    // Mass of emitted particle (proton), u
M_X = 27.98210;    // Mass of target nucleus (Al-27), u
M_Y = 30.973765;    // Mass of daughter nucleus (P-31), u
Q = ((m_x + M_X) - (m_y + M_Y))*u;    // Q-value of the reaction, MeV
Ex_threshold = -Q*(m_x + M_X)/M_X;    // The smallest value of the projectile energy, MeV
printf("\nThe threshhold energy of the reaction Al(27,13) + He --> P(31,15) + p = %4.2f MeV", Ex_threshold);

// Result
```

```
22 // The threshhold energy of the reaction Bi(209,83)
     + p --> Bi(208,83) + d = 5.25 MeV
23 // The threshhold energy of the reaction Al(27,13) +
      He --> P(31,15) + p = -3.31 MeV
```

Scilab code Exa 9.2.13 Finding unknown particles in the nuclear reactions

```
1  // Scilab Code Ex9.2.13: Page-418(2008)
2  clc; clear;
3  function p = Find(Z, A)
4      if Z == 2 & A == 4 then
5          p = 'alpha';
6      end
7      if Z == -1 & A == 0 then
8          p = 'beta-';
9      end
10     if  Z == 1 & A == 0 then
11         p = 'beta+';
12     end
13 endfunction
14 R1 = cell(4,3);
15 R2 = cell(4,3);
16 // Enter data for first cell (Reaction)
17 R1(1,1).entries = 'Li'; // Element
18 R1(1,2).entries = 3;    // Atomic number
19 R1(1,3).entries = 6;     // Mass number
20 R1(2,1).entries = 'd';
21 R1(2,2).entries = 1;
22 R1(2,3).entries = 2;
23 R1(3,1).entries = 'X';
24 R1(3,2).entries = 0;
25 R1(3,3).entries = 0;
26 R1(4,1).entries = 'He';
27 R1(4,2).entries = 2;
```

```
28 R1(4,3).entries = 4;
29 // Enter data for second cell (Reaction)
30 R2(1,1).entries = "Te";
31 R2(1,2).entries = 52;
32 R2(1,3).entries = 122;
33 R2(2,1).entries = 'X';
34 R2(2,2).entries = 0;
35 R2(2,3).entries = 0;
36 R2(3,1).entries = 'I';
37 R2(3,2).entries = 53;
38 R2(3,3).entries = 124;
39 R2(4,1).entries = 'd';
40 R2(4,2).entries = 1;
41 R2(4,3).entries = 2;
42 R1(3,2).entries = R1(1,2).entries+R1(2,2).entries-R1
      (4,2).entries
43 R1(3,3).entries = R1(1,3).entries+R1(2,3).entries-R1
      (4,3).entries
44 particle = Find(R1(3,2).entries, R1(3,3).entries);
         // Find the unknown particle
45 printf("\nFor the reaction\n")
46              printf("\t%s(%d) + %s(%d) --> %s + %s(%d
                   )\n X must be an %s particle", R1
                   (1,1).entries, R1(1,3).entries, R1
                   (2,1).entries, R1(2,3).entries, R1
                   (3,1).entries, R1(4,1).entries, R1
                   (4,3).entries, particle);
47 R2(2,2).entries = R2(3,2).entries+R2(4,2).entries-R2
      (1,2).entries
48 R2(2,3).entries = R2(3,3).entries+R2(4,3).entries-R2
      (1,3).entries
49 particle = Find(R2(2,2).entries, R2(2,3).entries);
         // Find the unknown particle
50 printf("\n\nFor the reaction\n")
51              printf("\t%s(%d) + %s --> %s(%d)+%s(%d)\
                   n X must be an %s particle", R2(1,1).
                   entries, R2(1,3).entries, R2(2,1).
                   entries, R2(3,1).entries, R2(3,3).
```

```
                entries, R2(4,1).entries, R2(4,3).
                entries, particle);
52
53 // Result
54 // For the reaction
55 //  Li(6) + d(2) --> X + He(4)
56 //  X must be an alpha particle
57
58 // For the reaction
59 //  Te(122) + X --> I(124)+d(2)
60 //  X must be an alpha particle
```

Scilab code Exa 9.2.14 Comptom scattering

```
1 // Scilab Code Ex9.2.14: Page-419(2008)
2 clc; clear;
3 h = 6.63e-034;     // Planck's constant, Js
4 c = 3e+008;     // Speed of light, m/s
5 lambda = 10e-012;     // Wavelength of incident X-
    rays, m
6 lambda_c = 2.426e-012;     // Compton wavelength for
    the electron, m
7 phi = 45;     // Angle of scattering of X-rays,
    degree
8 lambda_prime = lambda + lambda_c*(1 - cosd(phi));
      // Wavelength of scattered X-rays, m
9 // For maximum wavelength
10 phi = 180;     // Angle for maximum scattering,
    degree
11 lambda_prime_max = lambda + lambda_c*(1 - cosd(phi))
     ;     // Maximum wavelength present in the
    scattered X-rays, m
12 KE_max = h*c*(1/lambda-1/lambda_prime_max);     //
    Maximum kinetic energy of the recoil electrons, J
13 printf("\nThe wavelength of scattered X-rays = %5.2e
```

```
    m", lambda_prime);
14 printf("\nThe maximum wavelength present in the
    scattered X-rays = %6.3f pm", lambda_prime_max/1e
    -012);
15 printf("\nThe maximum kinetic energy of the recoil
    electrons = %5.3e J", KE_max);
16
17 // Result
18 // The wavelength of scattered X-rays = 1.07e-011 m
19 // The maximum wavelength present in the scattered X
    -rays = 14.852 pm
20 // The maximum kinetic energy of the recoil
    electrons = 6.498e-015 J
```

Scilab code Exa 9.2.16 Miller indices for the lattice planes

```
1  // Scilab Code Ex9.2.16: Page-420(2008)
2  clc; clear;
3  m = 3; n = 3; p = 2; // Coefficients of intercepts
    along three axes
4  m_inv = 1/m;         // Reciprocate the first
    coefficient
5  n_inv = 1/n;         // Reciprocate the second
    coefficient
6  p_inv = 1/p;         // Reciprocate the third
    coefficient
7  mul_fact = double(lcm(int32([m,n,p]))); // Find l.c.
    m. of m,n and p
8  m1 = m_inv*mul_fact;     // Clear the first fraction
9  m2 = n_inv*mul_fact;     // Clear the second fraction
10 m3 = p_inv*mul_fact;     // Clear the third fraction
11 printf("\nThe miller indices for planes with set of
    intercepts (%da, %db, %dc) are (%d %d %d) ", m, n
    , p, m1, m2, m3);
12 m = 1; n = 2; p = %inf; // Coefficients of
```

```
    intercepts along three axes
13 m_inv = 1/m;          // Reciprocate the first
    coefficient
14 n_inv = 1/n;          // Reciprocate the second
    coefficient
15 p_inv = 1/p;          // Reciprocate the third
    coefficient
16 mul_fact = double(lcm(int32([m,n]))); // Find l.c.m.
     of m,n and p
17 m1 = m_inv*mul_fact;     // Clear the first fraction
18 m2 = n_inv*mul_fact;     // Clear the second fraction
19 m3 = p_inv*mul_fact;     // Clear the third fraction
20 printf("\nThe miller indices for planes with set of
    intercepts (%da, %db, %dc) are (%d %d %d) ", m, n
    , p, m1, m2, m3);
21
22 // Result
23 // The miller indices for planes with set of
    intercepts (3a, 3b, 2c) are (2 2 3)
24 // The miller indices for planes with set of
    intercepts (1a, 2b, Infc) are (2 1 0)
```

Scilab code Exa 9.2.19 Glancing angles for the second and third order reflections

```
1 // Scilab Code Ex9.2.19: Page-421(2008)
2 clc; clear;
3 d = 1;     // For simplicity assume interplanar
    spacing to be unity, m
4 theta = 15;     // Glancing angle for first order,
    degree
5 n = 1;     // Order of reflection
6 // From Bragg's law, 2*d*sind(theta) = n*lambda,
    solving for lambda
7 lambda = 2*d*sind(theta)/n;     // Wavelength of
```

```
    incident X-ray, angstrom
// For second order reflection
n = 2
theta = asind(n*lambda/(2*d));     // Glancing angle
    for second order reflection, degree
printf("\nThe glancing angle for the second order
    reflection = %4.1f degree", theta);
// For third order reflection
n = 3;
theta = asind(n*lambda/(2*d));     // Glancing angle
    for third order reflection, degree
printf("\nThe glancing angle for the third order
    reflection = %4.1f degree", theta);

// Result
// The glancing angle for the second order
    reflection = 31.2 degree
// The glancing angle for the third order reflection
    = 50.9 degree
```