Scilab Textbook Companion for Callister's Materials Science and Engineering (adopted By R. Balasubramaniam) by W. D. Callister[1]

Created by
Ankit Rai
B.Tech
Mechanical Engineering
Madan Mohan Malaviya University of Technology
College Teacher
None
Cross-Checked by
Avik Kumar Das

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: Callister's Materials Science and Engineering (adopted By R. Balasubramaniam)

Author: W. D. Callister

Publisher: Wiley-Eastern

Edition: 2

Year: 2008

ISBN: 9788126521432

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

List of Figures

Chapter 3

Fundamentals of structure of crystalline solids

Scilab code Exa 3.1 Location of point having specified coordinates

```
1  clc
2  // Given
3  a = 0.48// let side of a unit cell
4  b = 0.46 //  let side of a unit cell
5  c = 0.40 // let side of a unit cell
6  p_x = 1/4
7  p_y = 1
8  p_z = 1/2
9  printf("Example 3.1")
10 qa = a*p_x // distance of point along x axis from
       origin in nm
11 rb = b*p_y // distance of point along y axis from
       origin in nm
12 sc = c*p_z // distance of point along z axis from
       origin
13 printf("\n Point P (%.2f,%.2f,%.2f) corresponds to",
       p_x,p_y,p_z)
14 printf(" %.2fnm,%.2fnm,%.2fnm ",qa,rb,sc)
```

Scilab code Exa 3.2 Specification of point coordinates

```
1
2  clc
3  //given that
4  // unit cell is BCC
5  printf("Example 3.2")
6  printf("\n\nPoint \t Fractional Length\t Point \n")
7  printf("Number \t x axix\t y axix\ z axix \t
       Coordinate\n")
8  n = 1 // serial number
9
10 for x = 0:1
11     for y = 0:1
12         for z = 0:1
13
14             printf("%d\t    %d        %d       %d \t %d
                   %d %d \n",n,x,y,z,x,y,z)
15            n = n+1
16         end
17     end
18 end
19 x = 0.5// x coordinate for center point
20 y = 0.5// y coordinate for center point
21 z = 0.5// y coordinate for center point
22 printf("%d\t  %.1f      %0.1f     %.1f    %.1f %.1f %.1
       f \n",n,x,y,z,x,y,z)
23 // sequence of point number is changed to make
       problem programable
```

Scilab code Exa 3.3 Specification of support Post Diameter

```
1
2 clc
3 // Given
4 a = 1 // let side of a unit cell
5 b = 1 // let side of a unit cell
6 c = 1 // let side of a unit cell
7 x_proj = 0.5*a // x coordinate of point
8 y_proj = 1*b // y coordinate of point
9 z_proj = 0*c // z coordinate of point
10
11 printf(" Example 3.3\n")
12 k = 1/x_proj
13 x_reduction = x_proj*k // reduction in x direction
14 y_reduction = y_proj*k // reduction in x direction
15 z_reduction = z_proj*k // reduction in x direction
16
17 printf("\n Indices of direction are as [%d %d %d]\n"
   ,x_reduction,y_reduction,z_reduction)
```

Scilab code Exa 3.4 Construction of specified crystallographic direction

```
1 clc
2 printf(" Example 3.4\n")
3 printf("\n It is a diagram based theoretically
   solved example\n")
```

Scilab code Exa 3.5 Determination of directional indices for a hexagonal unit cell

```
1
2
3 clc
4 // Given
5 u1 = 1 // Unit cell parameter
6 v1 = 1 // Unit cell parameter
7 w1 = 1 // Unit cell parameter
8
9 printf(" Example 3.5\n")
10 u_1 = 1/3*(2*u1-v1)
11 v_1 = 1/3*(2*v1-u1)
12 t_1 = -(u_1+v_1)
13 w_1 = w1
14 k = 1/u_1
15 u = u_1*k
16 v = v_1*k
17 w = w_1*k
18 t = t_1*k
19 printf("\n Indices of direction are as [%d %d %d %d
   ]\n",u,v,t,w)
20 // while in notation negative value is indicated by
   a bar over it
```

Scilab code Exa 3.6 Specification of support Post Diameter

```
1
2
3 clc
4 // Given
5 a = 1// let side of a unit cell
6 b = 1 //  let side of a unit cell
7 c = 1 // let side of a unit cell
8
9 printf(" Example 3.6\n")
10 x_proj = 0 // x coordinate of point
11 y_proj = -1*b // y coordinate of point
12 z_proj = 0.5*c // z coordinate of point
13 x_reduction = x_proj*a // reduction in x direction
14 y_reduction = 1/y_proj// reduction in x direction
15 z_reduction = 1/z_proj // reduction in x direction
16
17 printf("\n Indices of direction are as [%d %d %d]\n"
   ,x_reduction,y_reduction,z_reduction)
18 // while in notation negative value is indicated by
   a bar over it
```

Scilab code Exa 3.7 Construction of specified crystallographic plane

```
1
2 clc
3 printf(" Example 3.7 \n")
4 printf("\n It is a diagram based theoretically
   solved example\n")
```

Scilab code Exa 3.8 Determination of directional indices for a hexagonal unit cell

```
1
2
3 clc
4 // Given
5 u1 = 1 // Unit cell parameter
6 v1 = -1 // Unit cell parameter
7 w1 = 1 // Unit cell parameter
8
9 printf(" Example 3.8\n")
10 u_1 = 1/3*(2*u1-v1)
11 v_1 = 1/3*(2*v1-u1)
12 t_1 = -(u_1+v_1)
13 w_1 = w1
14 k = 1/u_1
15 u = u_1*k
16 v = v_1*k
17 w = w_1*k
18 t = t_1*k
19 printf("\n Indices of direction are as [%d %d %d %d
   ]\n",u,v,t,w)
20 // while in notation negative value is indicated by
   a bar over it
```

Chapter 4

Structure of crystalline solids

Scilab code Exa 4.1 Determination of FCC Unit Cell Volume

```
1
2
3 clc
4 //given that
5 R = 1 // let radius of an atom is unity
6 printf("Example 4.1\n")
7 //For FCC a=2*R*sqrt(2)
8
9 a=2*R*sqrt(2) //Edge Length
10 V=a^3 //Volume determination
11 printf("\n Volume of an FCC unit cell is %fR^3  \n\n
   ",V)
```

Scilab code Exa 4.2 Computation of the Atomic Packing Factor for FCC

```
1
2
```

```
clc
//given that
R = 1 // let radius of an atom to be unity
n=4 // no. of atoms for FCC are 4
printf("Example 4.2\n");
a=2*R*sqrt(2) // edge length for FCC
Vc=a^3 //Volume of cube
Vs=n*4*%pi*R^3/3 //Volume of sphere
APF=Vs/Vc //Atomic packing Fraction

printf("\n Atomic packing factor is %.2f",APF)
```

Scilab code Exa 4.3 Theoretical Density Computation for Copper

```


clc
// given that
R=1.28D-08//Atomic radius in cm
A_Cu=63.5 //Atomic wt of copper
n=4   //For FCC
Na=6.023D23 //Avogadro no.
printf("Example 4.3\n")

a=2*R*sqrt(2)
Vc=a^3
rho=n*A_Cu/(Vc*Na)
printf("\n Density of copper is %.2f g/cm^3.\n",rho)
```

Scilab code Exa 4.4 Computation of minimum cation to anion radius ratio for a coordination number of 3

```
clc
//given that
theta = 30 // angle between a line joining centres of anion with their median in degree
printf("Example 4.4\n")
r_ratio = (1- cos(theta*%pi/180))/ cos(theta*%pi/180)
printf("\n Minimum cation to anion radius ratio is %.3f",r_ratio)
```

Scilab code Exa 4.5 Ceramic crystal structure prediction

```
clc
// given that
r_fe2 = 0.077 // radius of Iron ion in nm
r_o2 = 0.14 // radius of oxygen ion in nm
r_ratio = r_fe2 /r_o2 // cation - anion radius ratio for FeO
printf("Example 4.5\n")
if r_ratio > 0.414 & r_ratio < 0.732 then
    printf("\n  As ratio lies between 0.414 and 0.732 and coordination number 6 \n so it is of AX crystal structure.")
end

// coordination number 6 is taken from table
```

Scilab code Exa 4.6 Theoretical density calculation for Sodium chloride

```
1
2
3 clc
4 //given that
5 n = 4 // number of crystals per unit cell
6 A_c = 22.99 // molar mass of cation i.e. sodium
7 A_a = 35.45 // molar mass of cation i.e. chlorine
8 r_c = 1.02e-8 // radius of sodium atom in cm
9 r_a = 1.81e-8 // radius of sodium atom in cm
10 N_a = 6.023e23 // Avogadro constant
11
12 printf("Example 4.6\n")
13 a = 2*(r_c+r_a)// edge length of Nacl molecule
14 V_c = a^3 // Volume of unit cell
15 rho = (n*(A_c+A_a))/(V_c*N_a)
16 printf("\n Theoretical density of crystal is %0.2f g
   /cm^3 .",rho)
17 printf("\n\n This result compares very favourable
   with \n experimental value of 2.16 g/cm^3 .")
```

Scilab code Exa 4.7 Computation of density and percentage crystallinity of polyethylene

```
1
2
3 clc
```

```
// given that
n = 2 // number of repeated units within unit cell
A_c = 12.01  // molar mass of carbon
A_h = 1.008 // molar mass of hydrogen
a = 0.741 // edge length in x axis in nm
b = 0.494 // edge length in y axis in nm
c = 0.255 // edge length in z axis in nm
N_a = 6.023e23 // Avogadro constant
rho_s = 0.925 // density of branched polyethylene in
    g/cm^3
rho_a = 0.870 // density of totally amorphous
   polyethylene in g/cm^3
printf("Example 4.7\n")
printf(" \n Part A:")
A = 2*A_c+4*A_h // Molar mass of polyethylene
V_c = a*b*c*(1e-7)^3 // Volume of unit cell
rho_c = (n*A)/(V_c*N_a)

printf("\n Density of totally crystalline
   polyethylene is %0.3f g/cm^3 .",rho_c)
printf(" \n\n Part B:")
per_cry = (rho_c*(rho_s-rho_a))*100/(rho_s*(rho_c-
   rho_a))
printf("\n Percentage crystallinity is %0.1f%%",
   per_cry)
```

Scilab code Exa 4.8 Inter planer angle and Diffraction Angle Computations

```
1
2
3 clc
4 //given that
```

```
5  a = 0.2866 // lattice parameter in nm
6  h = 2 // component of set of plane
7  k = 2// component of set of plane
8  l = 0// component of set of plane
9  n = 1 // order of detraction
10 lambda = 0.1790 // wavelength of light in nm
11
12 printf("Example 4.8\n")
13 d_hkl=a/(sqrt(h^2+k^2+l^2))
14 theta=asind(n*lambda/(2*d_hkl))
15 printf("\n Answer A:")
16 printf("\n Interplanar spacing is %.4f nm.",d_hkl)
17 printf("\n\n Answer B:")
18 printf("\n Diffraction angle is %0.2f degree.\n",2*
     theta);
19 // Answer in book is 124.26 degree. It is so because
     of consideration of different number of
     significant figures in calculation.
```

Chapter 5

Imperfection in solids

Scilab code Exa 5.1 Number of Vacancies Computation at a Specified temperature

```
1
2
3 clc
4 // given that
5 Na=6.023*10^23 //Avogadro No.
6 rho=8.4e6 //Density of Copper in g/m^3
7 A=63.5 //Atomic weight of Copper
8 Qv=0.9 //Activation energy in eV
9 k=8.62*10^-5 //Boltzmann Constant in eV/K
10 T=1000+273//Temperature in K
11 printf("Example 5.1\n");
12 N=Na*rho/A //No. of atomic site per cubic meter
13 Nv=N*exp(-Qv/(k*T))
14 printf("\n Equilibrium number of vacancies/m^3 is %
   .1e for 1273K",Nv)
```

Scilab code Exa 5.2 Derivation of composition conversion equation

```
1
2 clc
3 printf(" Example 5.2 \n")
4 printf("\n It is a derivational problem \n")
```

Scilab code Exa 5.3 Composition Conversion From weight percent to Atom percent

```
1
2
3 clc
4 // given that
5 C_Al=97 //Aluminium wt%
6 C_Cu=3 //Copper wt%
7 A_Al=26.98 //Atomic wt of Aluminium
8 A_Cu=63.55 //Atomic wt of Copper
9
10 printf(" Example 5.3\n")
11     CAl=C_Al*A_Cu*100/((C_Al*A_Cu)+(C_Cu*A_Al))
12     CCu=C_Cu*A_Al*100/((C_Cu*A_Al)+(C_Al*A_Cu))
13 printf("\n Atomic %% of Al is %.1f%%",CAl);
14 printf("\n Atomic %% of Cu is %.1f%%\n",CCu)
```

Scilab code Exa 5.4 Computation of Number of Scotty Defect in KCl

```
1
2
```

```
3
4 clc
5 // given that
6 Na=6.023*10^23 //Avogadro No.
7 rho=1.955 //Density of KCl in g/cm^3
8 A_k= 39.10 //Atomic weight of potassium in g/mol
9 A_cl= 35.45 //Atomic weight of Chlorine in g/mol
10 Qs=2.6 //Activation energy in eV
11 k=8.62*10^-5 //Boltzmann Constant in eV/K
12 T=500+273 //Temperature in K
13
14 printf("Example 5.4\n")
15 A = A_k+A_cl // Molar mass of KCl in gram
16 N=Na*rho*1e6/A //No. of atomic site per cubic meter
17 Ns=N*exp(-Qs/(2*k*T))
18 printf("\n Number of Schottky defects are %.2e
   defects/m^3.",Ns)
```

Scilab code Exa 5.5 Determination of possible point defect types in Nacl due to presence of Ca ions

```
1
2 clc
3 printf(" Example 5.5\n")
4 printf("\n It is a theoretically solved example\n")
```

Scilab code Exa 5.6 Computations of ASTM Grain Size Number of Grains per Unit Area

```
1
2
3  clc
4  // given that
5  N=45 //Number of grains per square inch
6  M=85 // magnification
7
8  printf("Example 5.6\n");
9  printf("\n Part A");
10 n=(log(N)/log(2))+1 //calculation for grain size no.
       N=2^(n-1)
11 printf("\n Grain size number is %.1f\n",n)
12 printf("\n Part B")
13 Nm=(100/M)^2*2^(n-1)
14 printf("\n At magnification of 85x\n")
15 printf(" Number of grains per inch square are %.2f\n
     ",Nm)
16 // answer in book is 62.6. It is because of rounding
       off at intermediate stages
```

Chapter 6

Diffusion

Scilab code Exa 6.1 Diffusion Flux Computation

```
1
2
3 clc
4 Ca=1.2 //Concentration at A in  kg/m^3
5 Cb=0.8 //Concentration at B in  kg/m^3
6 xa=5*10^-3//Position 1 in m
7 xb=10*10^-3//Position 2 in m
8 D=3*10^-11//Diffusion coefficient in m^2/s
9 printf("\Example 6.1\n")
10 J=-D*(Ca-Cb)/(xa-xb)
11 printf("\n Diffusion flux is %.1e kg/m^2-s",J)
```

Scilab code Exa 6.2 Nonsteady State Diffusion Time Computation I

```
1
2
3 clc
```

```
// given that
x=5*10^-4      //Position in m
D=1.6*10^-11   //Diffusion coefficient  in m^2/s
Co=0.25//Initial Concentration in wt%
Cs=1.2 //Surface concentration in wt%
Cx=0.8 //Concentration at any x  in wt%
z1 = 0.35 // tabular data
z2 = 0.4 // tabular data
erf_z1 = 0.3794 // tabular data
erf_z2 = 0.4284 // tabular data
printf("Example 6.2\n")
C=1-((Cx-Co)/(Cs-Co))
z = (C-erf_z1)/(erf_z2-erf_z1) * (z2-z1) + z1 //
   Calculation by interpolation
t= ((x/(2*sqrt(D)))/z)^2 // calculation of time
printf("\n Time required is %d s or %.2f h\n",t,t
   /3600);
// Answer in book is 25400 sec or 7.1 h. It is
   because of considering different number of
   significant figure
```

Scilab code Exa 6.3 Nonsteady State Diffusion Time Computation II

```


clc
// Given that
D_500=4.8*10^-14 //Diffusion coefficient at 500 C
D_600=5.3*10^-13 //Diffusion coefficient at 600 C
t_600=10 //Time in hours to diffuse
printf("Example 6.3\n")
t_500=D_600*t_600/D_500
```

```
11 printf("\n Time to diffuse at 500 degree Celsius is
     %.1f h\n",t_500)
```

Scilab code Exa 6.4 Diffusion Coefficient Determination

```
1
2
3 clc
4 //given that
5 T=550+273 // temperature of aluminium in K
6 D_0=1.2*10^-4 //Temperature independent
     preexponential in m^2/s
7 Q_d=131000 //Activation energy in J/mol-K
8 R=8.31  //Universal Gas constant
9 printf("Example 6.4\n");
10 D=D_0*exp(-Q_d/(R*T));
11
12 printf("\n Diffusion coefficient is %.1e m^2/s\n",D)
     ;
```

Scilab code Exa 6.5 Diffusion Coefficient Activation Energy and Preexponential Calculations

```
1
2
3 clc
4 //From graph log D ad 1/T are deducted
5 inv_T1=0.8*10^-3//Reciprocal of temp.  in K^-1
6 inv_T2=1.1*10^-3 //Reciprocal of temp.  in K^-1
```

```
7 logD1=-12.4
8 logD2=-15.45
9
10 R=8.31 //Gas law Constant in J/mol-K
11
12 printf("Example 6.5\n")
13 Q_d=-2.3*R*(logD1-logD2)/(inv_T1-inv_T2)
14 printf("\n Activation energy is %d kJ/mol",Q_d/1000)
15 D_0=10^(logD2+(Q_d*inv_T2/(2.3*R)))//For calculating
     Preexponential factor
16 printf("\n Preexponential factor D_0 is %.1e m^2/s\n
   ",D_0)
17 // Answer in book is 5.2e-5 m^2/s. It is because of
   consideration of different number of significant
   figures
```

Scilab code Exa 6.6 Computation of Diffusion Flux of carbon dioxide through a plastic beverage shelf life

```
1
2
3 clc
4 //given that
5 P_m = 2.3e-14 // permissibility coefficient of CO2
   through PET
6 P_1 = 400// Pressure inside bottle in KPa
7 P_2 = 0.4 // Pressure outside bottle in KPa
8 A = 500 // Surface area of bottle in cm^2
9 x = 0.05 // wall thickness of bottle in cm
10 V = 750 // volume in cm^3
11 printf("\Example 6.6\n")
12 J = -P_m*(P_2-P_1)*1e3/x // calculation of diffusion
     flux
```

```
13 printf("\n Part A:")
14 printf("\n Diffusion flux is %0.1e cm^3 STP/(cm^2-s)
     ",J)
15 printf("\n\n Part B:")
16 V_co2 = J*A
17 t = V/V_co2 // calculation of self life
18 printf("\n Self life for bottle of pop is %d days (
     or about %d months).", t/(60*60*24),t
     /(60*60*24*30))
19 // Answer in book is 97 days. It is because of
     considering different number of significant
     figure
```

Chapter 7

Phase diagram

Scilab code Exa 7.1 Derive Lever rule

```
1 clc
2 printf(" Example 7.1 \n")
3 printf("\n It is a derivational problem \n")
```

Scilab code Exa 7.2 Relative phase amount determination mass and volume fraction

```
1
2 clc
3 printf(" Example 7.2\n")
4 printf("\n It is a diagram based theoretically
    solved example\n")
```

Scilab code Exa 7.3 Determination of phase present and computation of phase composition

```
1
2 clc
3 // given that
4 C1=40  // Overall alloy composition
5 Cb=98
6 Ca=10
7 rho_Sn=7.24 //  density in g/cm^3  density of tin
8 rho_Pb=11.23  // density in g/cm^3  density of lead
9 Ca_Sn=10
10 Ca_Pb=90
11 Cb_Sn=98
12 Cb_Pb=2
13 printf("Example 7.3\n")
14
15 printf("\n Part A:")
16
17 Wa=(Cb-C1)/(Cb-Ca)
18 Wb=(C1-Ca)/(Cb-Ca)
19
20 printf("\n Mass fraction for alpha and beta phases
      is %.2f and %.2f respectively\n",Wa,Wb);
21 printf("\n Part B:");
22
23 rho_a=100/((Ca_Sn/rho_Sn)+(Ca_Pb/rho_Pb));
24 rho_b=100/((Cb_Sn/rho_Sn)+(Cb_Pb/rho_Pb));
25
26 printf("\n Density of alpha phase is : %.2f g/cm^3",
      rho_a);
27 printf("\n Density of beta phase is : %.2f g/cm^3",
      rho_b);
28
29 Va=Wa/(rho_a*((Wa/rho_a)+(Wb/rho_b)));
30 Vb=Wb/(rho_b*((Wa/rho_a)+(Wb/rho_b)));
31 printf("\n Volume fraction of alpha phase : %.2f",Va
      );
```

```
32 printf("\n Volume fraction of beta phase : %.2f",Vb)
   ;
```

Scilab code Exa 7.4 Determining ferrite and cementite phase

```
1
2
3 clc
4 // given that
5 C0=0.35 // given composition
6 C_a=0.022 // given composition
7 C_Fe3C=6.7 // given composition
8 C_p=0.76
9 printf("Example 7.4\n")
10
11 printf("\n Part A: ")
12 W_a=(C_Fe3C-C0)/(C_Fe3C-C_a)
13 W_Fe3C=(C0-C_a)/(C_Fe3C-C_a)
14
15 printf("\n Mass fraction of total ferrite phase is
   %.2f",W_a)
16 printf("\n Mass fraction of cementide phase is %.2f\
   n",W_Fe3C)
17
18 printf("\n Part B:")
19 Wp=(C0-C_a)/(C_p-C_a)
20 W_a1=(C_p-C0)/(C_p-C_a)
21
22 printf("\n Mass fraction of Pearlite is %.2f",Wp)
23 printf("\n Mass fraction of proeutectoid ferrite is
   %.2f\n",W_a1)
24
25 printf("\n Part C:")
```

```
26
27 Wae=W_a-W_a1
28 printf("\n Mass fraction of eutectoid ferrite : %.3f
      \n",Wae)
29 // Answer in book is 0.39. It is due to considering
      different number of significant figure at
      intermediate steps
```

Chapter 8

Phase transformation

Scilab code Exa 8.1 Computation of Critical Nuclear Radius and Activation energy

```
1
2
3 clc
4 // Given that
5 Hf=-1.16*10^9  //  Latent heat of fusion in J/m^3
6 Y=0.132  //   Surface energy in J/m^2
7 Tm=1064+273  // Melting point of gold in K
8 T=1064+273-230  //  230 is supercoiling value in K
9 a=0.413*10^-9  //  Unit Cell edge length in m
10 n = 4 // Number of atoms in a FCC unit cell
11 printf("Example 8.1\n");
12
13 printf("\n Part A");
14
15 r=-2*Y*Tm/(Hf*(Tm-T));
16
17 printf("\n Critical Radius is : %.2f nm\n",r/10^-9);
18
19 G=16*%pi*Y^3*Tm^2/(3*Hf^2*(Tm-T)^2);
20
```

```
21 printf(" Activation free energy is : %.2e J\n",G) //
    Answer in book is 9.64e-19 J. It is due to
   approximation at intermediate stage
22
23 printf("\n Part B")
24 u_c=4*%pi*r^3/(3*a^3)
25
26 printf("\n Unit cells per paricle are : %d",u_c);
27 tot_uc = n*int(u_c) // Total no. of atoms per
   critical nucleus
28 printf("\n Total no. of atoms per critical nucleus
   are : %d\n",tot_uc);
```

Scilab code Exa 8.2 Microstructure determination for three isothermal heat transfer

```
1 clc
2 printf(" Example 8.2 \n")
3 printf("\n It is a diagram based theoretically
   solved example\n")
```

Chapter 9

Mechanical properties of metals

Scilab code Exa 9.1 Elongation or Elastic Computation

```
1
2
3 clc
4 // given that
5 E=110*10^3  //Young's modulus of Copper in MPa
6 sigma=276   //Applied stress in MPa
7 l_o=305     //Original length in mm
8 printf("Example 9.1\n")
9
10 del_l=sigma*l_o/E //Deformation
11
12 printf("\n Elongation obtained is %.2f mm \n",del_l)
```

Scilab code Exa 9.2 Computation of Load to Produce Specified Diameter Change

```
1
2 clc
3 // given that
4 del_d=-2.5*10^-3 //Deformation in diameter in mm
5 d_0=10   //Initial diameter in mm
6 v=0.34 //Poisson ratio for brass
7 E=97*10^3 //Modulus of elasticity in MPa
8 printf("Example 9.2\n")
9
10 e_x=del_d/d_0 // Strain in x-direction
11 printf("\n Strain in x-direction is %.1e",e_x)
12
13 e_z=-e_x/v
14 printf("\n Strain in z-direction is %.2e",e_z)
15
16 sigma=e_z*E      // Stress produced
17 F=sigma*%pi*(d_0^2)/4
18
19 printf("\n Applied force is %d N",F)
20 // Answer in book is 5600N. It is due to rounding
   off at intermediate steps
```

Scilab code Exa 9.3 Mechanical property determination from stress stain curve

```
1
2 clc
3 // given that
4
5 sigma_2=150 // Stress at a point in MPa
6 sigma_1=0 // Stress at a point in MPa
7 epsilon_2=0.0016 // Strain at a point in MPa
8 epsilon_1=0 // Strain at a point in MPa
```

```
d_0=12.8*10^-3 //Initial Diameter in m
sigma=450*10^6      //tensile strength in MPa
l0=250;      //Initial length in mm
e=0.06;      //strain
printf("Example 9.3\n")

printf("\n Part A")
E=(sigma_2-sigma_1)/(epsilon_2-epsilon_1) //Young's
   Modulus = stress/strain
printf("\n Modulus of elasticity is %.1f GPa \n
   which is very close to its true value 97 GPa",E
   /10^3)

printf("\n\n Part C")
A_0=%pi*d_0^2/4

F=sigma*A_0
printf("\n Maximum load sustained is %d N\n",floor(F
   /10)*10)
printf("\n Part D")
dl=e*l0
printf("\n Change in length is %d mm",dl);
```

Scilab code Exa 9.4 Ductile and true stress at fracture composition

```
clc
// given that

di=12.8 //Initial diameter in mm
df=10.7 //Final diameter  in mm
sigma=460*10^6      //Tensile strength
printf("Example 9.4\n")

```

```
printf("\n Part A")

RA = ((di^2-df^2)/di^2)*100 //Ductility in terms of Reduction Area
printf("\n Percent reduction in area is %d%%\n",RA)

printf("\n Part B")
A_o=%pi*di^2*10^-6/4

F=sigma*A_o
Af=%pi*df^2/4
sigma_t=F/Af
printf("\n True stress at failure is %d MPa",sigma_t)
// Answer in book is 660. It is due to founding off at intermediate stage
```

Scilab code Exa 9.5 Calculation of Strain Hardening Exponent

```


clc
//Given that

sigma_t=415 //True stress in MPa
et=0.1 //True strain
K=1035 // Bulk modulus in MPa

printf(" Example 9.5\n")

n=log(sigma_t/K)/log(et)

printf("\n Strain hardening coefficient is %.2f",n);
```

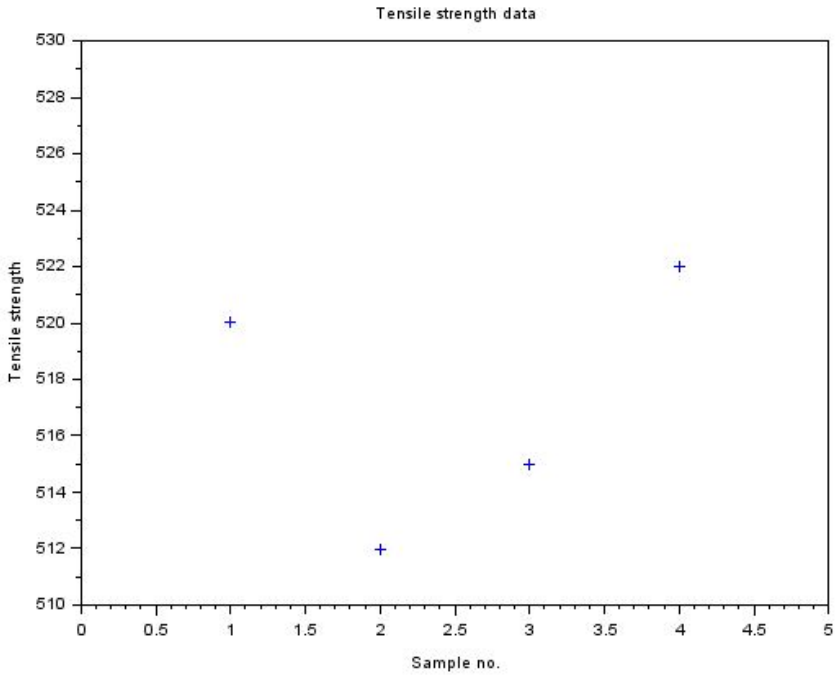

Figure 9.1: Average Computations

Scilab code Exa 9.6 Average Computations

```
1
2
3 clc
4 // given that
5 n=[0 1 2 3 4 5]
6 T_S=[510 520 512 515 522 530]
```

```
7
8  printf(" Example 9.6 \n")
9
10 //First and Last points are arbitrary to plot the
       required points
11
12 printf("\n Part A:")
13 plot(n,T_S,'+')
14 xtitle('Tensile strength data','Sample no.','Tensile
       strength');
15
16 //Mean Tensile strength
17 i=2;
18 TS_mean=0;
19
20 for i=2:5
21     TS_mean=TS_mean+(T_S(i)/4);
22 end
23 printf("\n Average tensile strength is %d MPa.\n",
       TS_mean);
24 //Standard Deviation
25 printf("\n\n Part B:")
26 j=0
27 std=0
28
29 for i=2:5
30     std=std+((T_S(i)-TS_mean)^2/(4-1))
31 end
32
33 printf("\n Standard deviation is %.1f MPa\n",sqrt(
       std))
```

Scilab code Exa 9.7 Specification of support Post Diameter

```
1
2
3 // Page no 334
4 clc
5 // Given
6 f = 220000 // Maximum load in N
7 sigma_y = 310 // Minimum field strength in MPa
8 sigma_t = 565 // Tensile strength strength in MPa
9 N= 5 // Factor of safety
10 printf("Design Example 9.1\n")
11 sigma_w = sigma_y*1e6/N
12 d = 2*sqrt (f/(2*%pi*sigma_w))// Calculation of
   diameter
13 printf("\n Diameter of each of two rods should be %
   .1 f mm\n",d*1e3)
```

Chapter 10

Dislocation and strengthening mechanism

Scilab code Exa 10.1 Resolved shear stress and stress to initiate yielding computations

```
1
2
3
4 clc
5 u1 = 0 // Bravais index
6 v1 = 1// Bravais index
7 w1 = 0// Bravais index
8 u2 = 1// Bravais index
9 v2 = 1// Bravais index
10 w2 = 0// Bravais index
11 u3 = -1// Bravais index
12 v3 = 1// Bravais index
13 w3 = 1// Bravais index
14 tau_r1 = 30 // Critical resolved shear stress
15 sigma = 52// Tensile strength in MPa
16 printf("Example 10.1\n")
17 printf("\n Part A:")
18 phi = acos((u1*u2+v1*v2+w1*w2)/sqrt((u1^2+v1^2+w1^2)
```

```
    *(u2^2+v2^2+w2^2)))
19 lambda = acos((u3*u1+v3*v1+w3*w1)/sqrt((u1^2+v1^2+w1
    ^2)*(u3^2+v3^2+w3^2)))
20 tau_r = sigma*cos(phi)*cos(lambda)
21 printf("\n Resolved shear stress is %.1f MPa",tau_r)
22 // Answer in book is 21.3 MPa which is due to
    approximation
23 printf("\n\n Part B:")
24 sigma1 = tau_r1/(cos(phi)*cos(lambda))
25 printf("\n Applied tensile force to initiate
    yielding is %.1f MPa",sigma1)
26 // Answer in book is 73.4 MPa which is due to
    approximation
```

Scilab code Exa 10.2 Tensile strength and ductility Determination for cold worked copper

```
1
2
3 clc
4 // Given that
5 d1 = 15.2 // Initial diameter in mm
6 d2 = 12.2 // Final diameter in mm
7 printf("Example 10.2\n")
8 per_CW = (d1^2 - d2^2)*100/d1^2
9 printf("\n The tensile strength is read directly
    from the curve for copper(figure 10.9b) \n as 340
     MPa From figure 10.19c, the Ductility at %0.1f
    CW is about 7%% EL.",per_CW)
10
11 // Some values are deduced from figures
```

Scilab code Exa 10.3 Tensile strength and ductility Determination for cold worked copper

```
1
2
3 //page no 371
4 clc
5 // Given that
6 d1 = 6.4 // Initial diameter in mm for first drawing
7 sigma = 345 // tensile strength in MPa
8 el = 20 // ductility in percent
9 d2 = 5.1 // Final diameter in mm
10 per_cw = 21.5 // deformation
11 printf("Design Example 10.1\n")
12 per_CW = (d1^2 - d2^2)*100/d1^2
13 d0 = sqrt((d2^2*100)/(100-el))
14 printf("\n Theoretical %% Cold Work is %.1f",per_CW)
15 printf("\n Original Diameter for second drawing is %
   .1f mm",d0)
16 // Answer in book is 5.8 mm which is due to
   approximation at intermediate steps
```

Chapter 11

Failure of metals

Scilab code Exa 11.1 Maximum Flaw Length Computation

```
1
2
3
4 clc
5 // given that
6
7 sigma=40*10^6 // in Pa  Tensile stress
8 E=69*10^9  //Modulus of elasticity  in pa
9 Ys=0.3  //Specific surface energy  in N/m^2
10 printf(" Example 11.1\n");
11
12 a=2*E*Ys/(%pi*sigma^2) //Maximum length of surface
   flaw
13
14 printf("\n Maximum length of surface flaw without
   fracture is %.1f micro meter \n",a*1e6);
```

Scilab code Exa 11.2 Materials specification for a pressurized steel tank

```
clc
//page no 399
printf(" Design Example 11.1\n")
printf("\n It is a diagram and table based
   theoretically solved example\n")
```

Scilab code Exa 11.3 Rupture Lifetime Prediction

```

//page no 421
clc
// Given that
T=800+273;  // Ambient temperature  in K
//stress is 140 MPa
L_M=24*10^3 // Larson - miller parameter
//From Graph of Fig. 8.32 Larson-Miller Parameter is
    deduced
printf("\n Design Example 11.2\n")
t=10^((L_M/T)-20)
printf("\n Time to rupture for a component is %d
   hours(%.1f days)\n",t,t/(24))
// Answer in book is 233 hours. it is because of
   approximation at intermediate stage
```

Chapter 13

Structure of polymers

Scilab code Exa 13.1 Computation of Average molecular weights and degree of polymerization

```
1
2
3 clc
4 // Given that
5 m1 = 7500 // Mean molecular weight in g/mol
6 m2 = 12500 // Mean molecular weight in g/mol
7 m3 = 17500 // Mean molecular weight in g/mol
8 m4 = 22500 // Mean molecular weight in g/mol
9 m5 = 27500 // Mean molecular weight in g/mol
10 m6 = 32500 // Mean molecular weight in g/mol
11 m7 = 37500 // Mean molecular weight in g/mol
12 x1 = 0.05 // Mole fraction
13 x2 = 0.16 // Mole fraction
14 x3 = 0.22 // Mole fraction
15 x4 = 0.27 // Mole fraction
16 x5 = 0.20 // Mole fraction
17 x6 = 0.08 // Mole fraction
18 x7 = 0.02 // Mole fraction
19 w1 = 0.02 // weight fraction
20 w2 = 0.10// weight fraction
```

```
21 w3 = 0.18// weight fraction
22 w4 = 0.29// weight fraction
23 w5 = 0.26// weight fraction
24 w6 = 0.13// weight fraction
25 w7 = 0.02// weight fraction
26 m_c = 12.01 // molar mass of carbon in gram/mole
27 m_h = 1.01 // molar mass of hydrogen in gram/mole
28 m_cl = 35.45 // molar mass of chlorine in gram/mole
29
30 printf("\n Problem 13.1")
31 printf("\n Part A:")
32 M_n = m1*x1+m2*x2+m3*x3+m4*x4+m5*x5+m6*x6+m7*x7
33 printf("\n The number average molecular weight is %d
       gram/mol",M_n)
34 printf("\n\n Part B:")
35 m = 2*m_c+3*m_h+1*m_cl // Mass of repeating unit
36 DP = M_n/m
37 printf("\n The degree of polimarization is %d ",DP)
38 printf("\n\n Part C:")
39 M_w = m1*w1+m2*w2+m3*w3+m4*w4+m5*w5+m6*w6+m7*w7
40 printf("\n The weight average molecular weight is %d
       gram/mol",M_w)
```

Chapter 15

Composite materials

Scilab code Exa 15.1 Property Determination for a glass fiber reinforced composite longitudinal direction

```
1
2
3 clc
4 // given that
5 v_f = 40 // volume percent of fibre in composite
6 E_f= 69 // Modulus of elasticity of fibre in GPa
7 v_m = 60  // volume percent of matrix in composite
8 E_m = 3.4// Modulus of elasticity of matrix in GPa
9 a = 250 // cross sectional area in mm^2
10 sigma = 50 // Tensile stress in MPa
11 Fm = 1 // let
12 Ff = 13.5*Fm
13 printf("\n Example 15.1")
14 printf("\n Part A:")
15 E_cl = (v_f*E_f+v_m*E_m)/100
16 printf("\n Modulus of elasticity of composite is %0
     .0f GPa",E_cl)
17 printf("\n\n Part B:")
18 Fc = a*sigma
19 Fm = Fc/(Fm+Ff)
```

```
20 Ff = Fc - Fm
21 printf("\n Force supported by m is %d N \n Force
      supported by fibre is %d N",Fm,Ff)
22 // Answer in book is as Fm = 860 N and Ff = 11640.
      It is due to approximation.
23 printf("\n\n Part C")
24 a_m = v_m*a/100
25 a_f = v_f*a/100
26 sigma_m = Fm/a_m
27 sigma_f = Ff/a_f
28 epsilon_m = sigma_m/(E_m*1000)
29 epsilon_f = sigma_f/(E_f*1000)
30 printf("\n Strain for matrix phase is %0.2e\n",
      epsilon_m)
31 printf(" Strain for fibre phase is %.2e.\n Both are
      identical",epsilon_f)
```

Scilab code Exa 15.2 Elastic Modulus Determination for a Glass Fiber Reinforced Composite Transverse Direction

```
1
2
3 clc
4
5 printf(" Example 15.2\n")
6 E_gf=69  // Elasticity of glass fibre in GPa
7 mf_gf=0.4  //Volume percentage of glass fibre
8 E_pr=3.4  //  Elasticity of polyester resin in GPa
9 mf_pr=0.6  //Vol percentage of polyester resin
10
11 E_ct=E_pr*E_gf/((E_pr*mf_gf)+(E_gf*mf_pr))  //
      Calculation of modulus of elasticity in GPa
12
```

```
13 printf("\n In transverse direction, modulus of
     elasticity is %.1f GPa.\n",ceil(E_ct*10)/10)
```

Scilab code Exa 15.3 Design a tubular composite shaft

```
1  //page no 563
2  clc
3  // Given
4  F = 1000 // Force in N
5  L = 1 // length in m
6  del_y = 0.35 // extension in mm
7  d_o = 70 // Outer diameter in mm
8  d_i = 50 // Innrer diameter in mm
9  V_f_max = 0.6 // Maximum allowable fiber Volume in
     cm fraction
10 Vf_glass = 0.945  // V_f for glass
11 Vf_C_standard = 0.293// V_f for carbon standard
     modulus
12 Vf_c_intermediate = 0.237// V_f for carbon
     intermediate modulus
13 Vf_c_high = 0.168 //  V_f for carbon high modulus
14 d_epoxy = 1.14 // density of epoxy resin in g/cm^3
15 d_C_sm = 1.8 // density of carbon fiber (Standard
     modulus) in g/cm^3
16 d_C_im = 1.8 // density of carbon fiber (
     intermediate modulus) in g/cm^3
17 d_C_hm = 1.8 // density of carbon fiber (high
     modulus) in g/cm^3
18 C_im_cost = 70.00 // cost of carbon fiber (
     intermediate modulus) in USD/kg
19 C_sm_cost = 35.00 // cost of carbon fiber (Standard
     modulus) in USD/kg
20 C_hm_cost = 175.00 // cost of carbon fiber (high
```

```
    modulus) in USD/kg
21 d_epoxy = 1.14 // density of epoxy resin in g/cm^3
22 epoxy_cost = 9.00  // cost of epoxy resin in USD/kg
23
24 printf("Design Example 15.1 \n")
25 I = %pi/64* (1e-12*(d_o*1e-3)^4-(d_i*1e-3)^4)
26 E = 4*F*L^3/(3*%pi*del_y*1e-3*((d_o*1e-3)^4-(d_i*1e
    -3)^4)) // Required modulus of elasticity
27 printf("\n Part A:\n")
28 if Vf_glass < V_f_max then
29   printf("\n Glass, when embedded in epoxy matrix,
       meet the stipulated criteria. \n")
30 end
31 if Vf_C_standard < V_f_max then
32   printf("\n Carbon (standard modulus), when
       embedded in epoxy matrix, meet the stipulated
       criteria. \n")
33 end
34 if Vf_c_intermediate < V_f_max then
35   printf("\n Carbon (intermediate modulus), when
       embedded in epoxy matrix, meet the stipulated
       criteria.\n ")
36 end
37 if Vf_c_high < V_f_max then
38   printf("\n Carbon (high modulus), when embedded in
        epoxy matrix, meet the stipulated criteria.\n"
       )
39 end
40 printf("\n Part B:\n")
41 Vc = %pi*L*1e-6*(d_o^2 - d_i^2)/4
42 F_v_C_sm = Vc*Vf_C_standard*1e6  // Fiber Volume in
    cm^3 for carbon (Standard modulus)
43
44 F_m_C_sm = F_v_C_sm * d_C_sm/1000 // Fiber mass for
    carbon (Standard modulus) in kg
45
46 F_c_C_sm = F_m_C_sm * C_sm_cost // Fiber cost for
    carbon (Standard modulus) in USD
```

```
47
48 m_v_C_sm = Vc*(1-Vf_C_standard)*1e6 // Matrix Volume
       in cm^3 for carbon (Standard modulus)
49
50 m_m_C_sm = m_v_C_sm * d_epoxy/1000 // Matrix mass
       for carbon (Standard modulus) in kg
51
52 m_c_C_sm = m_m_C_sm * epoxy_cost // Matrix cost for
       carbon (Standard modulus) in USD
53
54 Total_c_C_sm = m_c_C_sm + F_c_C_sm // Total cost for
       carbon (Standard modulus) in USD
55
56 F_v_C_im = Vc*Vf_c_intermediate*1e6 // Fiber Volume
       in cm^3 for carbon (intermediate modulus)
57
58 F_m_C_im = F_v_C_im * d_C_im/1000 // Fiber mass for
       carbon (intermediate modulus) in kg
59
60 F_c_C_im = F_m_C_im * C_im_cost // Fiber cost for
       carbon (intermediate modulus) in USD
61
62 m_v_C_im = Vc*(1-Vf_c_intermediate)*1e6 // Matrix
       Volume in cm^3 for carbon (intermediate modulus)
63
64 m_m_C_im = m_v_C_im * d_epoxy/1000 // Matrix mass
       for carbon (intermediate modulus) in kg
65
66 m_c_C_im = m_m_C_im * epoxy_cost // Matrix cost for
       carbon (intermediate modulus) in USD
67
68 Total_c_C_im = m_c_C_im + F_c_C_im // Total cost for
       carbon (intermediate modulus) in USD
69
70 F_v_C_hm = Vc*Vf_c_high*1e6  // Fiber Volume in cm^3
       for carbon (high modulus)
71
72 F_m_C_hm = F_v_C_hm * d_C_hm/1000 // Fiber mass for
```

```
    carbon (high modulus) in kg
73
74 F_c_C_hm = F_m_C_hm * C_hm_cost // Fiber cost for
    carbon (high modulus) in USD
75
76 m_v_C_hm = Vc*(1-Vf_c_high)*1e6 // Matrix Volume in
    cm^3 for carbon (high modulus)
77
78 m_m_C_hm = m_v_C_hm * d_epoxy/1000 // Matrix mass
    for carbon (high modulus) in kg
79
80 m_c_C_hm = m_m_C_hm * epoxy_cost // Matrix cost for
    carbon (high modulus) in USD
81 Total_c_C_hm = m_c_C_hm + F_c_C_hm // Total cost for
     carbon (high modulus) in USD
82 printf(" Cost of Carbon (standard modulus) composite
     is:%.2f ",Total_c_C_sm) // whereas Value in
    table is 48.50 USD
83
84 printf("\n Cost of Carbon (intermediate modulus)
    composite is:%.2f ",Total_c_C_im)// whereas Value
     in table is 71.10 USD
85 printf(" \n Cost of Carbon (high modulus) composite
    is:%.2f ",Total_c_C_hm) // whereas Value in table
     is 115.00 USD
86
87 printf("\n\n The material of choice (i.e. least
    expensive) \n is standard modulus carbon fiber
    composite; the relatively low cost per unit mass
    of this fiber offsets its relatively low modulus
    of elasticity and \n required high Volume
    fraction.")
```

Chapter 16

Corrosion and degradation

Scilab code Exa 16.1 Determination of Electrochemical Cell Characteristics

```
1
2
3 clc
4 // Given that
5
6 V_Cd=-0.403  //Half Cell Potential of Cd++|Cd in
     volt
7 V_Ni=-0.250  //Half Cell Potential of Ni++|Ni volt
8 C_Ni=10^-3
9 C_Cd=0.5
10 n=2  //Net electron exchange in Redox reaction
11 printf(" Example 16.1\n")
12 printf("\n Part A:")
13 dV=V_Ni-V_Cd // Potential difference in volts
14 printf("\n Standard Cell potential is : %.3f V\n",dV
     )
15 printf("\n Part B:")
16
17 V=-dV-(0.0592*log10(C_Ni/C_Cd)/n)
18 printf("\n Net EMF is : %.3f V\n",V)
```

```
19 printf("\n That is,")
20
21 if V<0 then
22     printf("\t Cd is oxidised & Ni is reduced \n")
23 else
24     printf("\t Cd is reduced & Ni is oxidised\n")
25 end
```

Scilab code Exa 16.2 Rate of Oxidation Computation

```
1
2 clc
3 // given that
4 VH_H2=0 // half cell voltage
5 VZn_Zn2=-0.763 // half cell voltage
6 iZn=10^-7 // current density in A/cm^2
7 iH2=10^-10 // current density in A/cm^2
8 beta_Zn=0.09
9 beta_H2=-0.08
10 n=2 //Exchange of 2 electrons
11 F=96500//Faradays constant
12
13 printf(" Example 16.2\n")
14 printf("\n Part A")
15 i_c=10^[(VH_H2-VZn_Zn2-(beta_H2*log10(iH2))+(beta_Zn
   *log10(iZn)))/(beta_Zn-beta_H2)]
16 printf("\n  Rate of oxidation with zinc is %.2e A/cm
   ^2",i_c)
17 r=i_c/(n*F)
18 printf("\n  Rate of oxidation is %.2e mol/cm^2-s\n",
   floor(r*1e12)/1e12)
19 printf("\n Part B")
20 Vc=VH_H2+(beta_H2*log10(i_c/iH2))
```

```
21 printf("\n Corrosion potential is %.3f V\n",Vc)
```

Chapter 17

Electrical properties of materials

Scilab code Exa 17.1 Computation of the Room Temperature Intrinsic Carrier Concentration for Gallium Arsenide

```
1
2 clc
3 // Given that
4 sigma=10^-6  // ( Electrical Conductivity in Ohm-m)
    ^-1
5 e=1.6*10^-19  // Charge on electron in Coulomb
6 m_e=0.85 //Mobility of electron in m^2/V-s
7 m_h=0.04 // Mobility of holes in m^2/V-s
8
9 printf(" Example 17.1\n")
10 n_i=sigma/(e*(m_e+m_h))//n_i is Intrinsic carrier
    concentration
11
12 printf("\n Intrinsic Carrier Concentration is %.1e m
    ^-3\n",n_i);
```

Scilab code Exa 17.2 Electrical Conductivity Determination for Intrinsic Silicon at 150 degree C

```
1
2 clc
3 // Given that
4 e=1.6*10^-19  //Charge on electron in Coulomb
5 ni=4*10^19 //  number of electron per unit volume
    for Si at 423 K   (m^-3)
6 //Values of m_e and m_h are deduced from graphs at
    page No.689
7 m_e=0.06  //Mobility of electron in m^2/V-s
8 m_h=0.022  //  Mobility of holes in m^2/V-s
9 printf(" Example 17.2\n")
10 sigma=ni*e*(m_e+m_h) // electrical conductivity
11
12 printf("\n Electrical Conductivity is %.2f (Ohm-m)
    ^-1\n",sigma)
```

Scilab code Exa 17.3 Room Temperature for Extrinsic Silicon

```
1
2 clc
3 // given that
4 n=10^23 //  Carrier Concentration in m^-3
5 e=1.6*10^-19  //Charge on electron in Coulomb
6 //From graph 18.18 m_e is calculated corresponding
    to n=10^23
```

```
m_e=0.07  // Mobility of electron in m^2/V-s
m_e2=0.04  //  Mobility of electron m^2/V-s
printf(" Example 17.3\n")
printf("\n Part A:\n ")
printf("\n Material is n-type \n ")
printf("\n Part B\n ")
sigma=n*e*m_e // electrical conductivity calculation for extrinsic n-type
printf("\n Conductivity is just %d (Ohm-m)^-1\n", sigma)
printf("\n Part C\n");
//From graph 18.19a m_e2 is calculated corresponding to 373 K
sigma2=n*e*m_e2
printf("\n Conductivity at T=373 K becomes %d (Ohm-m)^-1\n",sigma2);
```

Scilab code Exa 17.4 Hall Voltage Computation

```

clc
// given that
sigma=3.8*10^7  // Electrical Conductivity in (Ohm-m)^-1
m_e=0.0012  // Mobility of electron in m^2/V-s
I_x=25  // Current in Ampere(A)
d=15*10^-3 //Thickness in m
B_z=0.6  // Magnetic field in Tesla
printf("Example 17.4\n")
Rh=-m_e/sigma  //Hall coefficient
printf("\n Hall coefficient is %.2e V-m/A-Tesla\n", Rh)
Vh=Rh*I_x*B_z/d
```

```
13 printf("\n Hall Voltage is %.2e V\n",Vh)
```

Scilab code Exa 17.5 Computation of Capacitor properties

```
1
2 clc
3 // given that
4 A = 6.45e-4 // Area of plat in m^2
5 l = 2e-3 // separation between plates in m
6 epsilon_r = 6 //dielectric constant of material
7 epsilon_0 = 8.85e-12 // universal constant
8 V = 10 // Applied voltage in Volt
9 printf("\n Example 17.5")
10 printf("\n Part A:")
11 C = epsilon_0*epsilon_r*A/l // Capacitance of a
     parallel plat capacitor
12 printf("\n Capacitance of capacitor is %.2e F",C)
13 printf("\n\n Part B:")
14 Q = C*V // Stored charge calculation
15 printf("\n Stored charge in capacitor is %.2e C",Q)
16 printf("\n\n Part C:")
17 D = epsilon_0*epsilon_r*V/l // Dielectric
     displacement
18 printf("\n Dielectric displacement in capacitor is %
     .2e C/m^2",ceil(D*1e9)/1e9)
19 printf("\n\n Part D:")
20 P = D - epsilon_0*V/l // Polarisation
21 printf("\n Polarization is %.2e C/m^2", ceil(P*1e9)
     /1e9)
```

Scilab code Exa 17.6 Acceptor impurity doping in Silicon

```
1
2
3
4 //page no 649
5 clc
6 // given that
7 p1 = 1e22 // Number of electrons per unit volume
8 e = 1.6e-19 // Charge on electron in coulomb
9 mu_h1 = 0.04 // concentration of holes mobility in m
    ^/Vs
10 sigma_d = 50 // Desired conductivity in (ohm-m)^-1
11 p2 = 1e21// Number of electrons per unit volume
12 mu_h2 = 0.045// concentration of holes mobility in m
    ^/Vs
13 p3 = 8e21// Number of electrons per unit volume
14 mu_h3 = mu_h1// concentration of holes mobility in m
    ^/Vs
15 N_a = 6.023e23 // Avogadro s constant
16 rho_si = 2.33e6 // density of silicon in g/m^3
17 A_si = 28.09 // molecular weight in g/mol
18 printf("\n Design Example 17.1\n")
19 sigma1 = p1*e*mu_h1
20 sigma2 = p2*e*mu_h2
21 sigma3 = p3*e*mu_h3
22 N_Si = N_a*rho_si/A_si
23 Ca  = p3*100/(p3+N_Si)
24 printf("\n Silicon material of p-type of
    conductivity %d (ohm-m)^-1 \n must have %.2e%%
    doping material",sigma_d,Ca)
```

Chapter 18

Magnetic Properties

Scilab code Exa 18.1 Calculation of saturation magnetization and flux density for Nickel

```
1
2 clc
3 // Given that
4 b_m=9.27*10^-24 // Bohr Magneton in ampere*m^2
5 Na=6.023*10^23  //Avogadro's Number
6 d=8.9*10^6  // density in g/m^3
7 uo=4*%pi*10^-7  //Permittivity of free space
8 A=58.71  // Atomic weight of Nickel g/mol
9 printf("Example 18.1\n")
10 N=d*Na/A  //No. of atoms per cubic meter
11 M=0.6*b_m*N  //0.6= Bohr Magneton/atom
12 printf("\n Part A:")
13 printf("\n Saturation Magnetization is %.1e Ampere/m
   ",M)
14 M=0.6*b_m*N  //0.6= Bohr Magneton/atom
15 printf("\n\n Part B:")
16 B=uo*M
17 printf("\n Saturation Flux Density is %.2f Tesla\n",
   B);
```

Scilab code Exa 18.2 Calculation of saturation magnetization of Fe3O4

```
1
2
3 clc
4 // Given that
5 a=0.839*10^-9 //a is edge length in m
6 b_m=9.27*10^-24 //  Bohr Magneton in ampere*m^2
7 n_b=8*4     //8 is no. of Fe++ ions per unit cell and
      4 is Bohr magnetons per Fe++ ion
8 printf("Example 18.2\n")
9 M=n_b*b_m/a^3  //M is Saturation magnetisation
10 printf("\n Saturation Magnetization is %.1e Ampere/m
     \n",M)
```

Scilab code Exa 18.3 Designing a cubic mixed ferrite magnetic material

```
1
2 //page no 692
3 clc
4 // Given that
5 Ms_Fe=5.25*10^5;  //Required saturation
     Magnetisation in A/m
6 b_m=9.27*10^-24; //Bohr Magneton in ampere*m^2
7 a=0.839*10^-9; //a is edge length in m
8 M=5*10^5;  //From previous question result
9 printf(" Design Example 18.1\n");
10 y=poly([0],'y')  // Defining X
```

```
11 nb=Ms_Fe*a^3/b_m;
12 // 'x' represent fraction of Mn++ that have
      substituted Fe++
13 n=roots(8*[5*y+4*(1-y)]-nb);  //5 is Bohr magnetons
      per Fe++ ion
14                                //4 is Bohr magnetons
                                    per Mn++ ion
15 printf("\n Replacing %.1f%% of Fe++ with Mn++ would
      produce \n the required saturation magnetisation
      of %.2e A/m\n",n*100,Ms_Fe);
```