Scilab Textbook Companion for Thermodynamics, Statistical Thermodynamics and Kinetics by T. Engel and P. Reid[1]

Created by
Ebby George
M.tech
Electrical Engineering
NITK
College Teacher
None
Cross-Checked by
None

July 31, 2019

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

Book Description

Title: Thermodynamics, Statistical Thermodynamics and Kinetics

Author: T. Engel and P. Reid

Publisher: Prentice Hall, New York

Edition: 1

Year: 2007

ISBN: 13978032161503

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

Fundamental Concepts of Thermodynamics

Scilab code Exa 1.1 Final Tyre pressure

```
1 ////Variable Declaration
2 Pi = 3.21e5                    //Recommended tyre pressure,
      Pa
3 Ti = -5.00                     //Initial Tyre temperature,
      C
4 Tf = 28.00                     //Final Tyre temperature,
      C
5
6 //Calculations
7 Ti = 273.16 + Ti
8 Tf = 273.16 + Tf
9 pf = Pi*Tf/Ti                  //Final tyre pressure, Pa
10
11 //Results
12 printf("\n Final Tyre pressure is %6.2e Pa",pf)
```

Scilab code Exa 1.2 Partial pressure of pHe

```
1 ////Variable Declaration
2 phe = 1.5           //Pressure in Helium chamber, bar
3 vhe = 2.0           //Volume of Helium chamber, L
4 pne = 2.5           //Pressure in Neon chamber, bar
5 vne = 3.0           //Volume of Neon chamber, L
6 pxe = 1.0           //Pressure in Xenon chamber, bar
7 vxe = 1.0           //Volume of Xenon chamber, L
8 R = 8.314e-2        //Ideal Gas Constant, L.bar/(mol.
    K)
9 T = 298             //Temperature of Gas, K
10 //Calculations
11
12 nhe = phe*vhe/(R*T)              //Number of moles of
    Helium, mol
13 nne = pne*vne/(R*T)              //Number of moles of
    Neon, mol
14 nxe = pxe*vxe/(R*T)              //Number of moles of
    Xenon, mol
15 n = nhe + nne + nxe              //Total number of
    moles, mol
16 V = vhe + vne + vxe              //Total volume of
    system, L
17 xhe = nhe/n
18 xne = nne/n
19 xxe = nxe/n
20 P = n*R*T/(V)
21 phe = P*xhe                 //Partial pressure of
    Helium, bar
22 pne = P*xne                 //Partial pressure of Neon,
     bar
23 pxe = P*xxe                 //Partial pressure of Xenon
    , bar
24
25 //Results
26 printf("\n Moles of He=%4.3f, Ne=%4.3f and, Xe=%4.3f
    in mol",nhe,nne,nxe )
```

```
27
28 printf("\n Mole fraction of xHe=%4.3f, xNe=%4.3f and
       , xXe=%4.3f",xhe,xne,xxe)
29
30 printf("\n Final pressure is %4.3f bar",P)
31
32 printf("\n Partial pressure of pHe=%4.3f, pNe=%4.3f
       and, pXe=%4.3f in bar",phe,pne,pxe)
```

Scilab code Exa 1.4 Pressure from ideal gas law

```
1 ////Variable Declaration
2 T = 300.0                    //Nitrogen temperature, K
3 v1 = 250.00                  //Molar volume, L
4 v2 = 0.1                     //Molar volume, L
5 a = 1.37                     //Van der Waals parameter a,
      bar.dm6/mol2
6 b = 0.0387                   //Van der Waals parameter b,
      dm3/mol
7 R = 8.314e-2                 //Ideal Gas Constant, L.bar
     /(mol.K)
8 n = 1.
9 //Calculations
10
11 p1 = n*R*T/v1
12 p2 = n*R*T/v2
13 pv1 = n*R*T/(v1-n*b)- n**2*a/v1**2
14 pv2 = n*R*T/(v2-n*b)- n**2*a/v2**2
15
16 //Results
17 printf("\n Pressure from ideal gas law = %4.2e bar
       nad from Van der Waals equation = %4.2e bar ",p1,
      pv1)
18
19 printf("\n Pressure from ideal gas law = %4.1f bar
```

```
nad from Van der Waals equation = %4.1f bar ",p2,
 pv2)
```

Chapter 2

Heat Work Internal Energy Enthalpy and The First Law of Thermodynamics

Scilab code Exa 2.1 Part d Work done stretching th fiber

```
1 ////
2 //Variable Declaration  Part a
3 vi = 20.0                 //Initial volume of ideal gas,
      L
4 vf = 85.0                 //final volume of ideal gas, L
5 Pext = 2.5                //External Pressure against
     which work is done, bar
6
7 //Calculations
8 w = -Pext*1e5*(vf-vi)*1e-3
9
10 //Results
11 printf("\n Part a: Work done in expansion is %6.1f
     kJ",w/1000)
12
13
14 //Variable Declaration  Part b
```

```
15 ri = 1.00                //Initial diameter of bubble,
      cm
16 rf = 3.25                //final diameter of bubble, cm
17 sigm = 71.99             //Surface tension, N/m
18
19 //Calculations
20 w = -2*sigm*4*%pi*(rf**2-ri**2)*1e-4
21
22 //Results
23 printf("\n Part b: Work done in expansion of bubble
      is %4.2f J",w)
24
25
26 //Variable Declaration  Part c
27 i = 3.20                //Current through heating coil,
       A
28 v = 14.5                //fVoltage applied across coil,
       volts
29 t = 30.0                //time for which current is
      applied,s
30
31 ////Calculations
32 w = v*i*t
33
34 //Results
35 printf("\n Part c: Work done in paasing the cuurent
      through coil is %4.2f kJ",w/1000)
36
37
38 //Variable Declaration  Part d
39 k = 100.0                //Constant in F = -kx, N/cm
40 dl = -0.15               //stretch , cm
41
42 ////Calculations
43 w = -k*(dl**2-0)/2
44
45 //Results
46 printf("\n Part d: Work done stretching th fiber is
```

```
%4.2f J",w)
```

Scilab code Exa 2.2 Work done in vaporizing liquid

```
////Variable Declaration
m = 100.0                //Mass of water, g
T = 100.0                //Temperature of water, C
Pext = 1.0               //External Pressure on assembly
    , bar
x = 10.0                 //percent of water vaporised at
     1 bar,-
i = 2.00                 //current through heating coil,
     A
v = 12.0                 //Voltage applied, v
t = 1.0e3                //time for which current
    applied, s
rhol = 997               //Density of liquid, kg/m3
rhog = 0.59              //Density of vapor, kg/m3

//Calculations
q = i*v*t
vi = m/(rhol*100)*1e-3
vf = m*(100-x)*1e-3/(rhol*100) + m*x*1e-3/(rhog*100)
w = -Pext*(vf-vi)*1e5
//Results
printf("\n Heat added to the water %4.2f kJ",q/1000)

printf("\n Work done in vaporizing liquid is %4.2f J
    ",w)
```

Scilab code Exa 2.3 Heat removed by water at constant pressure

```
////Variable Declaration  Part d
```

```
2 m = 1.5
3 dT = 14.2              //Change in temperature of
     water, C or K
4 cp = 4.18              //Specific heat of water at
     constant pressure, J/(g.K)
5
6 //Calculations
7 qp = m*cp*dT
8
9 //Results
10 printf("\n Heat removed by water at constant
     pressure %4.2f kJ",qp)
```

Scilab code Exa 2.4 Work done

```
1 ////
2 //Variable declaration
3 n = 2.0              //moles of ideal gas
4 R = 8.314            //Ideal gas constant, bar.L/(mol.K)
5 //For reverssible Isothermal expansion
6 Pi1 = 25.0           //Initial Pressure of ideal gas,
     bar
7 Vi1 = 4.50           //Initial volume of ideal gas, L
8 Pf1 = 4.50           //Fianl Pressure of ideal gas, bar
9 Pext = 4.50          //External pressure, bar
10 Pint = 11.0         //Intermediate pressure, bar
11
12 //Calcualtions reverssible Isothermal expansion
13 T1 = Pi1*Vi1/(n*R)
14 Vf1 = n*R*T1/Pf1
15 w = -n*R*T1*log(Vf1/Vi1)
16
17 //Results
18 printf("\n For reverssible Isothermal expansion')
19 printf("\n Work done = %4.2e J",w)
```

```
20
21
22 //Calcualtions Single step irreverssible expansion
23
24 w = -Pext*1e5*(Vf1-Vi1)*1e-3
25
26 //Results
27 printf("\n For Single step reverssible expansion')
28 printf("\n Work done = %4.2e J",w)
29
30
31 //Calcualtions Two step irreverssible expansion
32 Vint = n*R*T1/(Pint)
33 w = -Pint*1e5*(Vint-Vi1)*1e-3 - Pf1*1e5*(Vf1-Vint)*1
      e-3
34
35 //Results
36 printf("\n For Two step reverssible expansion')
37 printf("\n Work done = %4.2e J",w)
```

Scilab code Exa 2.5 energy values

```
1 ////
2 //Variable declaration
3 n = 2.5              //moles of ideal gas
4 R = 0.08314          //Ideal gas constant, bar.L/(mol.K)
5 cvm = 20.79          //Heat Capacity at constant volume,
      J/(mol.K)
6
7 p1 = 16.6           //Pressure at point 1, bar
8 v1 = 1.00           //Volume at point 1, L
9 p2 = 16.6           //Pressure at point 2, bar
10 v2 = 25.0          //Volume at point 2, L
11 v3 = 25.0          //Volume at point 3, L
12
```

```
13 //Calculations
14 T1 = p1*v1/(n*R)
15 T2 = p2*v2/(n*R)
16 T3 = T1          //from problem statement
17          //for path 1-2
18 DU12 = n*cvm*(T2-T1)
19 w12 = -p1*1e5*(v2-v1)*1e-3
20 q12 = DU12 - w12
21 DH12 = DU12 + n*R*(T2-T1)*1e2
22
23          //for path 2-3
24 w23 = 0.0
25 DU23=n*cvm*(T3-T2)
26 ;q23=n*cvm*(T3-T2)
27 ;
28 DH23 = -DH12
29
30
31          //for path 3-1
32 DU31 = 0.0        //Isothemal process
33 DH31 = 0.0
34 w31 = -n*R*1e2*T1*log(v1/v3)
35 q31 = -w31
36
37 DU = DU12+DU23+DU31
38 w = w12+w23+w31
39 q = q12+q23+q31
40 DH = DH12+DH23+DH31
41
42 //Results
43 printf("\n For Path        q          w          DU
                  DH          ')
44 printf("\n 1-2        %7.2f   %7.2f     %7.2f      %7.2f
    ",q12,w12,DU12,DH12)
45
46 printf("\n 2-3        %7.2f     %7.2f    %7.2f     %7.2f
    ",q23,w23,DU23,DH23)
47
```

```
48 printf("\n 3-1          %7.2f     %7.2f      %7.2f
          %7.2f",q31,w31,DU31,DH31)
49
50 printf("\n Overall     %7.2f   %7.2f      %7.2f
   %7.2f",q,w,DU,DH)
51
52 printf("\n all values are in J')
```

Scilab code Exa 2.6 The work done in expansion of adiabatic procees

```
1 ////Variable Declaration  Part d
2 n = 2.5            //moles of ideal gas
3 R = 8.314          //Ideal gas constant, J/(mol.K)
4 cvm = 12.47        //Heat Capacity at constant volume,
      J/(mol.K)
5
6 pext = 1.00        //External Pressure, bar
7 Ti = 325.          //Initial Temeprature, K
8 pi = 2.50          //Initial Pressure, bar
9 pf = 1.25          //Final pressure, bar
10
11 //Calculations  Adiabatic process q = 0; DU = w
12 q = 0.0
13 Tf = Ti*(cvm + R*pext/pi)/(cvm + R*pext/pf )
14 DU=n*cvm*(Tf-Ti)
15 ;w=n*cvm*(Tf-Ti)
16 ;
17 DH = DU + n*R*(Tf-Ti)
18
19 //Results
20 printf("\n The final temperature at end of adiabatic
      procees is %4.1f K",Tf)
21
22 printf("\n The enthalpy change of adiabatic procees
   is %4.1f J",DH)
```

```
23
24 printf("\n The Internal energy change of  adiabatic
     procees is %4.1f J",DU)
25
26 printf("\n The work done in expansion of adiabatic
     procees is %4.1f J",w)
```

Scilab code Exa 2.7 Final temperature of cloud

```
1 ////
2 //Variable Declaration  Part d
3 h1 = 1000.0           //initial Altitude of cloud, m
4 hf = 3500.0           //Final Altitude of cloud, m
5 p1 = 0.802            //Pressure at h1, atm
6 pf = 0.602            //Pressure at hf, atm
7 T1 = 288.0            //Initial temperature of cloud,
     K
8 cp = 28.86            //Specific heat of air, J/mol.K
9 R = 8.314             //Gas constant, J/mol.K
10
11 //Calculations
12 Tf = exp(-(cp/(cp-R)-1)/(cp/(cp-R))*log(p1/pf))*T1
13 //Results
14 printf("\n Final temperature of cloud %4.1f K",Tf)
```

Chapter 3

Importance of State Functions Internal Energy and Enthalpy

Scilab code Exa 3.2 Pressure increase in capillary

```
1 //////Variable Declaration
2 betaOH = 11.2e-4              //Thermal exapnasion
    coefficient of ethanol, C
3 betagl = 2.00e-5              //Thermal exapnasion
    coefficient of glass, C
4 kOH = 11.0e-5                 //Isothermal
    compressibility of ethanol, /bar
5 dT = 10.0                     //Increase in Temperature
    , C
6
7 //Calcualtions
8 vfbyvi = (1+ betagl*dT)
9 dP = betaOH*dT/kOH-(1./kOH)*log(vfbyvi)
10
11 //Results
12 printf("\n Pressure increase in capillary %4.1f bar"
    ,dP)
```

Scilab code Exa 3.4 Minimum detectable temperature change of gas

```
1 ////Variable Declaration
2 cpsubysy = 1000            //Specific heat ration of
     surrounding and system
3 Tpreci = 0.006              //Precision in
     Temperature measurement,  C
4
5 //Calcualtions
6 dtgas = -cpsubysy*(-Tpreci)
7
8 //Results
9 printf("\n Minimum detectable temperature change of
     gas +-%4.1f  C ",dtgas)
```

Scilab code Exa 3.9 Enthalpy change for change in state of methanol

```
1 ////Variable Declaration
2 m = 124.0          //Mass of liquid methanol, g
3 Pi = 1.0           //Initial Pressure, bar
4 Ti = 298.0         //Intial Temperature, K
5 Pf = 2.5           //Final Pressure, bar
6 Tf = 425.0         //Intial Temperature, K
7 rho = 0.791        //Density, g/cc
8 Cpm = 81.1         //Specifi heat, J/(K.mol)
9 M = 32.04
10
11 //Calculations
12 n = m/M
13 DH = n*Cpm*(Tf-Ti)+ m*(Pf-Pi)*1e-6/rho
14
15 //Results
```

```
16 printf("\n Enthalpy change for change in state of
      methanol is %4.1f kJ",DH/1000)
```

Chapter 4

Thermochemistry

Scilab code Exa 4.1 Average bond energy required for breaking both OH bonds

```
1 ////Varialble Declaration
2 DH0_H2O = 241.8           //Std Enthalpy of reaxtion
     of Water Fomation backward rxn, kJ/mol
3 DH0_2H = 2*218.0          //Std Enthalpy of formation
     of Hydrogen atom, kJ/mol
4 DH0_O = 249.2             //Std Enthalpy of formation
     of Oxygen atom, kJ/mol
5 R = 8.314                 //Ideal gas constant, J/(
     mol.K)
6 Dn = 2.0
7 T = 298.15                //Std. Temperature, K
8 //Calculation
9 DH0_2HO = DH0_H2O + DH0_2H + DH0_O
10 DU0 = (DH0_2HO - Dn*R*T*1e-3)/2
11
12 //Results
13 printf("\n Avergae Enthalpy change required for
     breaking both OH bonds %4.1f kJ/mol",DH0_2HO)
14
15 printf("\n Average bond energy required for breaking
     both OH bonds %4.1f kJ/mol",DU0)
```

Scilab code Exa 4.3 Enthalpy of rection for benzene

```
1 ////Varialble Declaration
2 ms1 = 0.972           //Mass of cyclohexane, g
3 DT1 = 2.98            //Change in temperature for
     bath, C
4 DUR1 = -3913e3        //Std Internal energy change, J
     /mol
5 mw = 1.812e3          //Mass of water, g
6 ms2 = 0.857           //Mass of benzene, g
7 Ms1 = 84.16
8 Ms2 = 78.12
9 DT2 = 2.36            //Change in temperature for
     bath, C
10 Mw = 18.02
11 Cpw = 75.3
12
13 //Calculation
14
15 Ccal = ((-ms1/Ms1)*DUR1-(mw/Mw)*Cpw*DT1)/DT1
16 DUR2 = (-Ms2/ms2)*((mw/Mw)*Cpw*DT2+Ccal*DT2)
17
18 //Results
19 printf("\n Calorimeter constant %4.2e J/ C ",Ccal)
20
21 printf("\n Enthalpy of rection for benzene %4.2e J/
     mol",DUR2)
```

Scilab code Exa 4.4 Enthalpy of solution for Na2SO4 from Data

```
1 ////Varialble Declaration
```

```
ms = 1.423              //Mass of Na2SO4, g
mw = 100.34             //Mass of Na2SO4, g
DT = 0.037              //Change in temperature for
    solution, K
Mw = 18.02              //Molecular wt of Water
Ms = 142.04             //Molecular wt of ms Na2SO4
Ccal = 342.5            //Calorimeter constant, J/K
Cpw = 75.3
//Data
DHfNa = -240.1
DHfSO4 = -909.3
DHfNa2SO4 = -1387.1

//Calculation
DHs = (-Ms/ms)*((mw/Mw)*Cpw*DT+Ccal*DT)
DHsolD = 2*DHfNa + DHfSO4 - DHfNa2SO4

//Results
printf("\n Enthalpy of solution for Na2SO4 %4.2e J/
    mol",DHs)

printf("\n Enthalpy of solution for Na2SO4 from Data
    %4.2e J/mol",DHsolD)
```

Chapter 5

Enthalpy and the Second and Third Laws of Thermodynamics

Scilab code Exa 5.1 Work done by heat engine

```
1 ////Variable Declaration
2 Th = 500.
3 Tc = 200.      //Temeperatures IN Which reversible
     heat engine works, K
4 q = 1000.               //Heat absorbed by heat
     engine, J
5
6 //Calcualtions
7 eps = 1.-Tc/Th
8 w = eps*q
9
10 //Results
11 printf("\n Efficiency of heat engine is %4.3f",eps)
12
13 printf("\n Work done by heat engine is %4.1f J",w)
```

Scilab code Exa 5.5 Entropy change of process

```
1 //////
2 //Variable Declaration
3 n = 2.5                    //Number of moles of CO2
4 Ti = 450.      //Initial and final state
     Temeperatures of CO2, K
5 Tf = 800.
6 pi = 1.35      //Initial and final state pressure of
     CO2, K
7 pf = 3.45
8 [A,B,C,D] = (18.86,7.937e-2,-6.7834e-5,2.4426e-8)
9                            //Constants in constant
                                pressure Heat capacity
                                equation in J, mol, K
                                units
10 R = 8.314                 //Ideal Gas Constant, J/(mol.
     K)
11 //Calcualtions
12
13 dS1 = n*integrate('(A+B*T+C*T**2+D*T**3)/T','T',Ti,
     Tf)
14 dS2 = n*R*log(pf/pi)
15 dS = dS1 - dS2
16 //Results
17 printf("\n Entropy change of process is %4.2f J/(mol
     .K)",dS)
```

Scilab code Exa 5.6 Ratio of pressure to temperature dependent term nhence effect

```
1 ////
2 //Variable Declaration
3 n = 3.0                    //Number of moles of CO2
4 Ti = 300      //Initial and final state Temeperatures
      of CO2, K
```

```
5 Tf = 600
6 pi = 1.00      //Initial and final state pressure of
     CO2, K
7 pf = 3.00
8 cpm = 27.98              //Specific heat of mercury, J
     /(mol.K)
9 M = 200.59               //Molecualr wt of mercury, g
     /(mol)
10 beta = 1.81e-4          //per K
11 rho = 13.54             //Density of mercury, g/cm3
12 R = 8.314               //Ideal Gas Constant, J/(mol.
     K)
13
14 //Calcualtions
15 dS1 = n*cpm*log(Tf/Ti)
16 dS2 = n*(M/(rho*1e6))*beta*(pf-pi)*1e5
17 dS = dS1 - dS2
18
19 //Results
20 printf("\n Entropy change of process is %4.1f J/(mol
     .K)",dS)
21
22 printf("\n Ratio of pressure to temperature
     dependent term %3.1e\nhence effect of pressure
     dependent term is very less",dS2/dS1)
23
24 printf("\n The above value is different as given in
     the text")
```

Scilab code Exa 5.7 Total Entropy change

```
1 ////
2 //Variable Declaration
3 n = 1.0                  //Number of moles of CO2
4 T = 300.0                //Temeperatures of Water bath
```

```
    , K
vi = 25.0      //Initial and final state Volume of
    Ideal Gas, L
vf = 10.0
R = 8.314              //Ideal Gas Constant, J/(mol.
    K)

//Calcualtions
qrev = n*R*T*log(vf/vi)
w = -qrev
dSsys = qrev/T
dSsur = -dSsys
dS = dSsys + dSsur

//Results
printf("\n Entropy change of surrounding is %4.1f J
    /(mol.K)",dSsur)

printf("\n Entropy change of system is %4.1f J/(mol.
    K)",dSsys)

printf("\n Total Entropy change is %4.1f J/(mol.K)",
    dS)
```

Scilab code Exa 5.8 Total Entropy change

```
////
//Variable Declaration
n = 1.0                //Number of moles of CO2
T = 300.0              //Temeperatures of Water bath
    , K
vi = 25.0      //Initial and final state Volume of
    Ideal Gas, L
vf = 10.0
R = 8.314              //Ideal Gas Constant, J/(mol.
```

```
    K)
8
9  //Calcualtions
10 pext = n*R*T/(vf/1e3)
11 pi = n*R*T/(vi/1e3)
12 q = pext*(vf-vi)/1e3
13 qrev = n*R*T*log(vf/vi)
14 w = -q
15 dSsur = -q/T
16 dSsys = qrev/T
17 dS = dSsys + dSsur
18
19 //Results
20 printf("\n Constant external pressure and initial
      pressure are %4.3e J,and %4.3e J respectively",
      pext,pi)
21
22 printf("\n Heat in reverssible and irreversible
      processes are %4.1f J,and %4.1f J respectively",
      qrev,q)
23
24 printf("\n Entropy change of system is %4.1f J/(mol.
      K)",dSsys)
25
26 printf("\n Entropy change of surrounding is %4.2f J
      /(mol.K)",dSsur)
27
28 printf("\n Total Entropy changeis %4.2f J/(mol.K)",
      dS)
```

Chapter 6

Chemical Equilibrium

Scilab code Exa 6.1 Maximum Available work through combustion of C8H18

```
1 ////Variable Declaration
2 dHcCH4 = -891.0          //Std. heat of combustion for
      CH4, kJ/mol
3 dHcC8H18 = -5471.0       //Std. heat of combustion for
      C8H18, kJ/mol
4
5 T = 298.15
6 [SmCO2,SmCH4,SmH2O,SmO2,SmC8H18] =
     (213.8,186.3,70.0,205.2,316.1)
7 dnCH4 = -2.
8 dnC8H18 = 4.5
9 R = 8.314
10 //Calculations
11 dACH4 = dHcCH4*1e3 - dnCH4*R*T - T*(SmCO2 + 2*SmH2O
     - SmCH4 - 2*SmO2)
12 dAC8H18 = dHcC8H18*1e3 - dnC8H18*R*T - T*(8*SmCO2 +
     9*SmH2O - SmC8H18 - 25.*SmO2/2)
13 //Results
14 printf("\n Maximum Available work through combustion
      of CH4 %4.1f kJ/mol",dACH4/1000)
15
```

```
16 printf("\n Maximum Available work through combustion
       of C8H18 %4.1f kJ/mol",dAC8H18/1000)
```

Scilab code Exa 6.2 Maximum nonexapnasion work through combustion of C8H18

```
1 ////Variable Declaration
2 dHcCH4 = -891.0          //Std. heat of combustion for
      CH4, kJ/mol
3 dHcC8H18 = -5471.0       //Std. heat of combustion for
      C8H18, kJ/mol
4
5 T = 298.15
6 [SmCO2,SmCH4,SmH2O,SmO2,SmC8H18] =
     (213.8,186.3,70.0,205.2,316.1)
7 dnCH4 = -2.
8 dnC8H18 = 4.5
9 R = 8.314
10 //Calculations
11 dGCH4 = dHcCH4*1e3  - T*(SmCO2 + 2*SmH2O - SmCH4 -
     2*SmO2)
12 dGC8H18 = dHcC8H18*1e3 - T*(8*SmCO2 + 9*SmH2O -
     SmC8H18 - 25.*SmO2/2)
13 //Results
14 printf("\n Maximum nonexpansion work through
     combustion of CH4 %4.1f kJ/mol",dGCH4/1000)
15
16 printf("\n Maximum nonexpansion work through
     combustion of C8H18 %4.1f kJ/mol",dGC8H18/1000)
```

Scilab code Exa 6.4 Std free energy of formation for Fe g at 400 K

```
1 ////Variable Declaration
```

```
dGf298 = 370.7      //Std. free energy of formation
   for Fe (g), kJ/mol
dHf298 = 416.3      //Std. Enthalpy of formation for
   Fe (g), kJ/mol
T0 = 298.15         //Temperature in K
T = 400.            //Temperature in K
R = 8.314

//Calculations

dGf = T*(dGf298*1e3/T0 + dHf298*1e3*(1./T - 1./T0))

//Results
printf("\n Std. free energy of formation for Fe(g at
    400 K is %4.1f kJ/mol",dGf/1000)
```

Scilab code Exa 6.5 Std entropy Change on mixing

```
////
//Variable Declaration
nHe = 1.0           //Number of moles of He
nNe = 3.0           //Number of moles of Ne
nAr = 2.0           //Number of moles of Ar
nXe = 2.5           //Number of moles of Xe
T = 298.15          //Temperature in K
P = 1.0             //Pressure, bar
R = 8.314

//Calculations
n = nHe + nNe + nAr + nXe
dGmix = n*R*T*((nHe/n)*log(nHe/n) + (nNe/n)*log(nNe/
   n) +(nAr/n)*log(nAr/n) + (nXe/n)*log(nXe/n))
dSmix = n*R*((nHe/n)*log(nHe/n) + (nNe/n)*log(nNe/n)
    +(nAr/n)*log(nAr/n) + (nXe/n)*log(nXe/n))

```

```
16 //Results
17 printf("\n Std. free energy Change on mixing is %3.1
      e J",dGmix)
18
19 printf("\n Std. entropy Change on mixing is %4.1f J"
      ,dSmix)
```

Scilab code Exa 6.6 Std Gibbs energy change for reaction

```
1 ////Variable Declaration
2 dGfFe  = 0.0        //Std. Gibbs energy of formation
     for Fe (S), kJ/mol
3 dGfH2O = -237.1     //Std. Gibbs energy of formation
     for Water (g), kJ/mol
4 dGfFe2O3 = -1015.4  //Std. Gibbs energy of formation
     for Fe2O3 (s), kJ/mol
5 dGfH2 = 0.0         //Std. Gibbs energy of formation
     for Hydrogen (g), kJ/mol
6 T0 = 298.15         //Temperature in K
7 R = 8.314
8 [nFe,nH2,nFe2O3,nH2O] = (3,-4,-1,4)
9
10 //Calculations
11 dGR = nFe*dGfFe + nH2O*dGfH2O + nFe2O3*dGfFe2O3 +
     nH2*dGfH2
12
13 //Results
14 printf("\n Std. Gibbs energy change for reaction is
     %4.2f kJ/mol",dGR)
```

Scilab code Exa 6.7 Std Gibbs energy change for reactionat

```
1 ////Variable Declaration
```

```
2 dGR  = 67.0          //Std. Gibbs energy of formation
    for reaction, kJ, from previous problem
3 dHfFe  = 0.0         //Enthalpy of formation for Fe (S
    ), kJ/mol
4 dHfH2O = -285.8      //Enthalpy of formation for Water
    (g), kJ/mol
5 dHfFe2O3 = -1118.4 //Enthalpy of formation for Fe2O3
    (s), kJ/mol
6 dHfH2 = 0.0          //Enthalpy of formation for
    Hydrogen (g), kJ/mol
7 T0 = 298.15          //Temperature in K
8 T = 525.             //Temperature in K
9 R = 8.314
10 [nFe,nH2,nFe2O3,nH2O] = (3,-4,-1,4)
11
12 //Calculations
13 dHR = nFe*dHfFe + nH2O*dHfH2O + nFe2O3*dHfFe2O3 +
    nH2*dHfH2
14 dGR2 = T*(dGR*1e3/T0 + dHR*1e3*(1./T - 1./T0))
15
16 //Results
17 printf("\n Std. Enthalpy change for reactionat %4.1f
    is %4.2f kJ/mol",T, dHR)
18
19 printf("\n Std. Gibbs energy change for reactionat
    %4.1f is %4.0f kJ/mol",T, dGR2/1e3)
```

Scilab code Exa 6.8 Std Gibbs energy change for reaction

```
1 ////
2 //Variable Declaration
3 dGfNO2  = 51.3       //Std. Gibbs energy of formation
    for NO2 (g), kJ/mol
4 dGfN2O4 = 99.8       //Std. Gibbs energy of formation
    for N2O4 (g), kJ/mol
```

```
5 T0 = 298.15          //Temperature in K
6 pNO2 = 0.350         //Partial pressure of NO2, bar
7 pN2O4 = 0.650        //Partial pressure of N2O4, bar
8 R = 8.314
9 [nNO2,nN2O4] = (-2,1)
10
11 //Calculations
12 dGR = nN2O4*dGfN2O4*1e3 + nNO2*dGfNO2*1e3 + R*T0*log
      (pN2O4/(pNO2)**2)
13
14 //Results
15 printf("\n Std. Gibbs energy change for reaction is
      %5.3f kJ/mol",dGR/1e3)
```

Scilab code Exa 6.9 Equilibrium constant for reaction

```
1 ////
2 //Variable Declaration
3 dGfCO2  = -394.4    //Std. Gibbs energy of formation
      for CO2 (g), kJ/mol
4 dGfH2 = 0.0         //Std. Gibbs energy of formation
      for H2 (g), kJ/mol
5 dGfCO = 237.1       //Std. Gibbs energy of formation
      for CO (g), kJ/mol
6 dGfH2O = 137.2      //Std. Gibbs energy of formation
      for H24 (l), kJ/mol
7 T0 = 298.15         //Temperature in K
8 R = 8.314
9 [nCO2, nH2, nCO, nH2O] = (1,1,1,1) //Stoichiomentric
      coeff of CO2,H2,CO,H2O respectively in reaction
10
11 //Calculations
12 dGR = nCO2*dGfCO2 + nH2*dGfH2 + nCO*dGfCO + nH2O*
      dGfH2O
13 Kp = exp(-dGR*1e3/(R*T0))
```

```
14
15 //Results
16 printf("\n Std. Gibbs energy change for reaction is
      %5.3f kJ/mol",dGR/1e3)
17
18 printf("\n Equilibrium constant for reaction is %5.3
      f ",Kp)
```

Scilab code Exa 6.12 Equilibrium constants at 1000 1100 and 1200 K

```
1 ////
2 //Variable Declaration
3 dGfCaCO3  = -1128.8      //Std. Gibbs energy of
      formation for CaCO3 (s), kJ/mol
4 dGfCaO = -603.3          //Std. Gibbs energy of
      formation for CaO (s), kJ/mol
5 dGfCO2 = -394.4          //Std. Gibbs energy of
      formation for O2 (g), kJ/mol
6 dHfCaCO3 = -1206.9       //Std. Enthalpy Change of
      formation for CaCO3 (s), kJ/mol
7 dHfCaO = -634.9          //Std. Enthalpy Change of
      formation for CaO (s), kJ/mol
8 dHfCO2 = -393.5          //Std. Enthalpy Change of
      formation for O2 (g), kJ/mol
9 T0 = 298.15              //Temperature in K
10 R = 8.314
11 [nCaCO3,nCaO,nO2] = (-1,1,1)
12
13 //Calculations
14 dGR = nCaO*dGfCaO + nO2*dGfCO2 + nCaCO3*dGfCaCO3
15 dHR = nCaO*dHfCaO + nO2*dHfCO2 + nCaCO3*dHfCaCO3
16
17 deff('[x]=func(T)','x=exp(-dGR*1e3/(R*T0) - dHR*1e3
      *(1/T - 1/T0)/R)')
18
```

```
19 Kp10 = func(1000)
20 Kp11 = func(1100)
21 Kp12 = func(1200)
22
23 //Results
24 printf("\n Std. Gibbs energy change for reaction is
      %4.1f kJ/mol",dGR)
25
26 printf("\n Std. Enthalpy change for reaction is %4.1
      f kJ/mol",dHR)
27
28 printf("\n Equilibrium constants at 1000, 1100, and
      1200 K are %4.4f, %4.3fe, and %4.3f",Kp10,Kp11,
      Kp12)
```

Scilab code Exa 6.13 Pressure at which graphite and dimond will be in equilibrium

```
1 ////
2 //Variable Declaration
3 dGfCG  = 0.0              //Std. Gibbs energy of
      formation for CaCO3 (s), kJ/mol
4 dGfCD = 2.90              //Std. Gibbs energy of
      formation for CaO (s), kJ/mol
5 rhoG = 2.25e3             //Density of Graphite, kg/m3
6 rhoD = 3.52e3             //Density of dimond, kg/m3
7 T0 = 298.15               //Std. Temperature, K
8 R = 8.314                 //Ideal gas constant, J/(mol
      .K)
9 P0 = 1.0                  //Pressure, bar
10 M = 12.01                //Molceular wt of Carbon
11 //Calculations
12 P = P0*1e5 + dGfCD*1e3/((1./rhoG-1./rhoD)*M*1e-3)
13
14 //Results
15 printf("\n Pressure at which graphite and dimond
```

```
will be in equilibrium is %4.2e bar",P/1e5)
```

Scilab code Exa 6.14 dUbydVm

```
////
//Variable Declaration
beta = 2.04e-4              //Thermal exapansion
    coefficient, /K
kapa = 45.9e-6              //Isothermal compressibility
    , /bar
T = 298.15                  //Std. Temperature, K
R = 8.206e-2                //Ideal gas constant, atm.L
    /(mol.K)
T1 = 320.0                  //Temperature, K
Pi = 1.0                    //Initial Pressure, bar
V = 1.00                    //Volume, m3
a = 1.35                    //van der Waals constant a
    for nitrogen, atm.L2/mol2
P0 =1
//Calculations
dUbydV=(beta*T1-kapa*P0)/kapa
;Pf=(beta*T1-kapa*P0)/kapa
;
dVT = V*kapa*(Pf-Pi)
dVbyV = dVT*100/V
Vm = Pi/(R*T1)
dUbydVm = a/(Vm**2)

//Results
printf("\n dUbydV = %4.2e bar",dUbydV)

printf("\n dVbyV = %4.3f percent",dVbyV)

printf("\n dUbydVm = %4.0e atm",dUbydVm)
```

Scilab code Exa 6.15 Enthalpy change

```
1 ////
2 //Variable Declaration
3 m = 1000.0                    //mass of mercury, g
4 Pi  = 1.00      //Intial pressure and temperature,
     bar, K
5 Ti = 300
6 Pf = 300.      //Final pressure and temperature, bar,
     K
7 Tf  = 600.0
8 rho = 13534.                  //Density of mercury, kg/
     m3
9 beta = 18.1e-4                //Thermal exapansion
     coefficient for Hg, /K
10 kapa = 3.91e-6               //Isothermal
     compressibility for Hg, /Pa
11 Cpm = 27.98                  //Molar Specific heat at
     constant pressure, J/(mol.K)
12 M = 200.59                   //Molecular wt of Hg, g/
     mol
13
14 //Calculations
15 Vi = m*1e-3/rho
16 Vf = Vi*exp(-kapa*(Pf-Pi))
17 Ut = m*Cpm*(Tf-Ti)/M
18 Up =  (beta*Ti/kapa-Pi)*1e5*(Vf-Vi) + (Vi-Vf+Vf*log(
     Vf/Vi))*1e5/kapa
19 dU = Ut + Up
20 Ht = m*Cpm*(Tf-Ti)/M
21 Hp = ((1 + beta*(Tf-Ti))*Vi*exp(-kapa*Pi)/kapa)*(exp
     (-kapa*Pi)-exp(-kapa*Pf))
22 dH =  Ht + Hp
23 //Results
```

```
24 printf("\n Internal energy change is %6.2e J/mol in
     which \ncontribution of temperature dependent
     term %6.4f percent",dU,Ut*100/dH)
25
26 printf("\n Enthalpy change is %4.3e J/mol in which \
     ncontribution of temperature dependent term %4.1f
      percent",dH,Ht*100/dH)
```

Scilab code Exa 6.16 Molar Specific heat of Hg at const volume

```
1 ////Variable Declaration
2 T = 300.0                    //Temperature of Hg, K
3 beta = 18.1e-4               //Thermal exapansion
     coefficient for Hg, /K
4 kapa = 3.91e-6               //Isothermal
     compressibility for Hg, /Pa
5 M = 0.20059                  //Molecular wt of Hg, kg
     /mol
6 rho = 13534                  //Density of mercury, kg
     /m3
7 Cpm = 27.98                  //Experimental Molar
     specif heat at const pressure for mercury, J/(mol
     .K)
8
9 //Calculations
10 Vm = M/rho
11 DCpmCv = T*Vm*beta**2/kapa
12 Cvm = Cpm - DCpmCv
13 //Results
14 printf("\n Difference in molar specific heats \nat
     constant volume and constant pressure %4.2e J/(
     mol.K)",DCpmCv)
15
16 printf("\n Molar Specific heat of Hg at const.
     volume is %4.2f J/(mol.K)",Cvm)
```

Scilab code Exa 6.17 Molar Gibbs energy of Water

```
1 ////Variable Declaration
2 T = 298.15                    //Std. Temperature, K
3 P = 1.0                       //Initial Pressure, bar
4 [Hm0,Sm0] = (0.0,154.8)
5 [Sm0H2,Sm0O2] = (130.7,205.2)
6 dGfH2O = -237.1          //Gibbs energy of formation
     for H2O(l), kJ/mol
7 [nH2,nO2] = (1,1/2)
8
9 //Calculations
10 Gm0 = Hm0 - T*Sm0
11 dGmH2O = dGfH2O*1000 - T*(nH2*Sm0H2 + nO2*Sm0O2)
12 //Results
13 printf("\n Molar Gibbs energy of Ar %4.3f kJ/mol",
     Gm0/1e3)
14
15 printf("\n Molar Gibbs energy of Water %4.3f kJ/mol"
     ,dGmH2O/1e3)
```

Chapter 7

Properties of Real Gases

Scilab code Exa 7.3 Percentage error

```
1 ////Variable Declaration
2 m = 1.0              //Mass of Methane, kg
3 T = 230              //Temeprature of Methane, K
4 P = 68.0             //Pressure, bar
5 Tc = 190.56          //Critical Temeprature of Methane
6 Pc = 45.99           //Critical Pressure of Methane
7 R = 0.08314          //Ideal Gas Constant, L.bar/(mol.K
    )
8 M = 16.04            //Molecular wt of Methane
9
10 //Calcualtions
11 Tr = T/Tc
12 Pr = P/Pc
13 z = 0.63            //Methane compressibility factor
14 n = m*1e3/M
15 V = z*n*R*T/P
16 Vig = n*R*T/P
17 DV = (V - Vig)/V
18
19 //Results
20 printf("\n V-Videal %4.2f L",V-Vig)
```

```
21
22 printf("\n Percentage error %5.2f",DV*100)
```

Chapter 8

Phase Diagrams and the Relative Stability of Solids Liquids and Gases

Scilab code Exa 8.2 Triple point pressure

```
1 ////
2 //Varialble Declaration
3 Tn = 353.24          //normal boiling point of Benzene,
      K
4 pi = 1.19e4          //Vapor pressure of benzene at 20
      C , Pa
5 DHf = 9.95           //Latent heat of fusion, kJ/mol
6 pv443 = 137.         //Vapor pressure of benzene at
     -44.3 C , Pa
7 R = 8.314            //Ideal Gas Constant, J/(mol.K)
8 Pf = 101325          //Std. atmospheric pressure, Pa
9 T20 = 293.15         //Temperature in K
10 P0 = 1.
11 Pl = 10000.
12 Ts = -44.3          //Temperature of solid benzene,
      C
13
```

```
14 //Calculations
15 Ts = Ts + 273.15
16 //Part a
17
18 DHv = -(R*log(Pf/pi))/(1./Tn-1./T20)
19 //Part b
20
21 DSv = DHv/Tn
22 DHf = DHf*1e3
23 //Part c
24
25 Ttp = -DHf/(R*(log(Pl/P0)-log(pv443/P0)-(DHv+DHf)/(R
      *Ts)+DHv/(R*T20)))
26 Ptp = exp(-DHv/R*(1./Ttp-1./Tn))*101325
27
28 //Results
29 printf("\n Latent heat of vaporization of benzene at
       20 C  %4.1f kJ/mol",DHv/1000)
30
31 printf("\n Entropy Change of vaporization of benzene
       at 20 C  %3.1f J/mol",DSv)
32
33 printf("\n Triple point temperature = %4.1f K for
      benzene",Ttp)
34
35 printf("\n Triple point pressure = %4.2e Pa for
      benzene",Ptp)
```

Scilab code Exa 8.3 Force exerted by one leg

```
1 ////
2 //Varialble Declaration
3 gama = 71.99e-3    //Surface tension of water, N/m
4 r = 1.2e-4         //Radius of hemisphere, m
5 theta = 0.0        //Contact angle, rad
```

```
6
7 //Calculations
8 DP = 2*gama*cos(theta)/r
9 F = DP*%pi*r**2
10
11 //Results
12 printf("\n Force exerted by one leg %5.3e N",F)
```

Scilab code Exa 8.4 water can not reach top of tree

```
1 ////
2 //Varialble Declaration
3 gama = 71.99e-3    //Surface tension of water, N/m
4 r = 2e-5           //Radius of xylem, m
5 theta = 0.0        //Contact angle, rad
6 rho = 997.0        //Density of water, kg/m3
7 g = 9.81           //gravitational acceleration, m/s2
8 H = 100            //Height at top of redwood tree, m
9
10 //Calculations
11 h = 2*gama/(rho*g*r*cos(theta))
12
13 //Results
14 printf("\n Height to which water can rise by
      capillary action is %3.2f m",h)
15
16 printf("\n This is very less than %4.1f n, hence
      water can not reach top of tree",H)
```

Chapter 9

Ideal and Real Solutions

Scilab code Exa 9.2 Entropy change of mixing

```
1 ////
2 //Variable Declaration
3 nb = 5.00        //Number of moles of Benzene, mol
4 nt = 3.25        //Number of moles of Toluene, mol
5 T = 298.15       //Temperature, K
6 P = 1.0          //Pressure, bar
7 R = 8.314        //Ideal Gas Constant, J/(mol.K)
8
9 //Calculations
10 n = nb + nt
11 xb = nb/n
12 xt = 1. - xb
13 dGmix = n*R*T*(xb*log(xb)+xt*log(xt))
14 dSmix = -n*R*(xb*log(xb)+xt*log(xt))
15
16 //Results
17 printf("\n Gibbs energy change of mixing is %4.3e J"
   ,dGmix)
18
19 printf("\n Gibbs energy change of mixing is < 0,
   hence the mixing is spontaneous')
```

```
20 printf("\n Entropy change of mixing is %4.2f J/K",
     dSmix)
```

Scilab code Exa 9.3 Toulene fraction in vapor

```
1 ////Variable Declaration
2 nb = 5.00        //Number of moles of Benzene, mol
3 nt = 3.25        //Number of moles of Toluene, mol
4 T = 298.15       //Temperature, K
5 R = 8.314        //Ideal Gas Constant, J/(mol.K)
6 P0b = 96.4       //Vapor pressure of Benzene, torr
7 P0t = 28.9       //Vapor pressure of Toluene, torr
8
9 //Calculations
10 n = nb + nt
11 xb = nb/n
12 xt = 1. - xb
13 P = xb*P0b + xt*P0t
14 y = (P0b*P - P0t*P0b)/(P*(P0b-P0t))
15 yt = 1. -y
16
17 //Results
18 printf("\n Total pressure of the vapor is %4.1f torr
     ",P)
19
20 printf("\n Benzene fraction in vapor is %4.3f ",y)
21
22 printf("\n Toulene fraction in vapor is %4.3f ",yt)
```

Scilab code Exa 9.6 Vapor pressure of solvent

```
1 ////Variable Declaration
2 m = 4.50          //Mass of substance dissolved, g
```

```
3 ms = 125.0       //Mass of slovent (CCl4), g
4 TbE = 0.65       //Boiling point elevation, C
5 [Kf, Kb] = (30.0, 4.95)     //Constants for freezing
     point elevation
6                           // and boiling point
                               depression for CCl4, K kg
                               /mol
7 Msolvent = 153.8  //Molecualr wt of solvent, g/mol
8 //Calculations
9 DTf = -Kf*TbE/Kb
10 Msolute = Kb*m/(ms*1e-3*TbE)
11 nsolute = m/Msolute
12 nsolvent = ms/Msolvent
13 x = 1.0 -  nsolute/(nsolute + nsolvent)
14
15 //Results
16 printf("\n Freezing point depression %5.2f K",DTf)
17
18 printf("\n Molecualr wt of solute %4.1f g/mol",
     Msolute)
19
20 printf("\n Vapor pressure of solvent is reduced by a
     factor of %4.3f",x)
```

Scilab code Exa 9.7 Osmotic pressure

```
1 ////Variable Declaration
2 csolute = 0.500 //Concentration of solute, g/L
3 R = 8.206e-2    //Gas constant L.atm/(mol.K)
4 T = 298.15      //Temperature of the solution, K
5
6 //Calculations
7 pii = csolute*R*T
8
9 //Results
```

```
10 printf("\n Osmotic pressure %4.2f atm",pii)
```

Scilab code Exa 9.8 Activity coefficinet of CS2

```
1 ////Variable Declaration
2 xCS2 = 0.3502    //Mol fraction of CS2, g/L
3 pCS2 = 358.3     //Partial pressure of CS2, torr
4 p0CS2 = 512.3    //Total pressure, torr
5
6 //Calculations
7 alpha = pCS2/p0CS2
8 gama = alpha/xCS2
9
10 //Results
11 printf("\n Activity of CS2 %5.4f atm",alpha)
12
13 printf("\n Activity coefficient of CS2 %5.4f atm",
   gama)
```

Scilab code Exa 9.9 Activity coefficinet of CS2

```
1 ////Variable Declaration
2 xCS2 = 0.3502    //Mol fraction of CS2, g/L
3 pCS2 = 358.3     //Partial pressure of CS2, torr
4 kHCS2 = 2010.    //Total pressure, torr
5
6 //Calculations
7 alpha = pCS2/kHCS2
8 gama = alpha/xCS2
9
10 //Results
11 printf("\n Activity of CS2 %5.4f atm",alpha)
12
```

```
13 printf("\n Activity coefficinet of CS2 %5.4f atm",
      gama)
```

Scilab code Exa 9.10 Henrys constant

```
1 ////Variable Declaration
2 rho = 789.9       //Density of acetone, g/L
3 n = 1.0           //moles of acetone, mol
4 M = 58.08         //Molecular wt of acetone, g/mol
5 kHacetone = 1950 //Henrys law constant, torr
6 //Calculations
7 H = n*M*kHacetone/rho
8
9 //Results
10 printf("\n Henrys constant = %5.2f torr",H)
```

Scilab code Exa 9.11 Activity coefficient

```
1 ////Variable Declaration
2 m = 0.5           //Mass of water, kg
3 ms = 24.0         //Mass of solute, g
4 Ms = 241.0        //Molecular wt of solute, g/mol
5 Tfd = 0.359       //Freezinf point depression, C or
      K
6 kf = 1.86         //Constants for freezing point
      depression for water, K kg/mol
7
8 //Calculations
9 msolute = ms/(Ms*m)
10 gama = Tfd/(kf*msolute)
11
12 //Results
13 printf("\n Activity coefficient = %4.3f",gama)
```

Scilab code Exa 9.12 Volume of nitrogen released from blood at reduced pressure

```
1 ////Variable Declaration
2 m = 70.0          //Mass of human body, kg
3 V = 5.00          //Volume of blood, L
4 HN2 = 9.04e4      //Henry law constant for N2
     solubility in blood, bar
5 T = 298.0         //Temperature, K
6 rho = 1.00        //density of blood, kg/L
7 Mw = 18.02        //Molecualr wt of water, g/mol
8 X = 80            //Percent of N2 at sea level
9 p1= 1.0  //Pressures, bar
10  p2 = 50.0
11 R = 8.314e-2         //Ideal Gas constant, L.bar/(mol.
     K)
12 //Calculations
13 nN21  = (V*rho*1e3/Mw)*(p1*X/100)/HN2
14 nN22  = (V*rho*1e3/Mw)*(p2*X/100)/HN2
15 V = (nN22-nN21)*R*T/p1
16 //Results
17 printf("\n Number of moles of nitrogen in blood at 1
     and 50 bar are %3.2e,%3.3f mol",nN21,nN22)
18
19 printf("\n Volume of nitrogen released from blood at
     reduced pressure %4.3f L",V)
```

Chapter 10

Electrolyte Solutions

Scilab code Exa 10.2 Ionic streangth for NaCl solution

```
1 ////Variable Declaration
2 M = 0.050        //Molarity for NaCl and Na2SO4
     solution, mol/kg
3 [npa,zpa] = (1,1)
4 [nma,zma] = (1,1)
5 [npb,zpb] = (2,1)
6 [nmb,zmb] = (1,2)
7
8 //Calculations
9 Ia = M*(npa*zpa**2 + nma*zma**2)/2
10 Ib = M*(npb*zpb**2 + nmb*zmb**2)/2
11
12 //Results
13 printf("\n Ionic streangth for NaCl solution is %4.3
     f and for Na2SO4 solution is %4.3f, mol/kg",Ia,Ib
     )
```

Chapter 11

Electrochemical Cells Batteries and Fuel Cells

Scilab code Exa 11.1 The potential of H2 half cell

```
////
//Variable Declaration
aH = 0.770          //Activity of
fH2 = 1.13          //Fugacity of Hydrogen gas
E0 = 0.0            //Std. electrode potential, V
n = 1.0             //Number of electrons transfered

//Calculations
E = E0 - (0.05916/n)*log(aH/sqrt(fH2))

//Results
printf("\n The potential of H+/H2 half cell %5.4f V"
   ,E)
```

Scilab code Exa 11.2 E0a E0b

```
////Variable Declaration
E0r1 = -0.877        //Std Electrod potential for Rx2
    : Al3+ + 3e- ------> Al (s)
E0r2 = -1.660        //Std Electrod potential for Rx2
    : Al3+ + 3e- ------> Al (s)
E0r3 = +0.071        //Std Electrod potential for Rx3
    : AgBr (s) + e- ------> Ag(s) +Br- (aq.)

//Calculations
//3Fe(OH)2 (s)+ 2Al (s)  <----------> 3Fe (s)  + 6(OH
   -) + 2Al3+
E0a = 3*E0r1 + (-2)*E0r2
//Fe (s) + 2OH- +   2AgBr (s) --------> Fe(OH)2 (s)
   + 2Ag(s) +  2Br- (aq.)
E0b = -E0r1 + (2)*E0r3

//Results
printf("\n %5.3f %5.3f",E0a,E0b)
```

Scilab code Exa 11.3 E0 for overall reaction

```
////Variable Declaration
E01 = 0.771       //Rx1 : Fe3+ + e- -----> Fe2+
E02 = -0.447      //Rx2 : Fe2+ + 2e- -----> Fe
F = 96485         //Faraday constant, C/mol
[n1,n2,n3] = (1.,2.,3.)

//Calculations
dG01 = -n1*F*E01
dG02 = -n2*F*E02
                  //For overall reaction
dG0 = dG01 + dG02
E0Fe3byFe = -dG0/(n3*F)

//Results
```

```
15 printf("\n E0 for overall reaction is %5.3f V",
     E0Fe3byFe)
```

Scilab code Exa 11.4 Std entropy change of reaction from dE0bydT

```
1  ////Variable Declaration
2  E01 = +1.36                //Std. electrode potential
     for Cl2/Cl
3  dE0bydT = -1.20e-3         //V/K
4  F = 96485                  //Faraday constant, C/mol
5  n = 2.
6  S0H = 0.0                  //Std. entropy J/(K.mol) for
     H+ ,Cl-,H2, Cl2
7  S0Cl = 56.5
8  S0H2 = 130.7
9  S0Cl2 = 223.1
10 [nH,nCl,nH2,nCl2] = (2,2,-1,-1)
11 //Calculations
12 dS01 = n*F*dE0bydT
13 dS02 =nH*S0H + nCl*S0Cl + nH2*S0H2 + nCl2*S0Cl2
14
15 //Results
16 printf("\n Std. entropy change of reaction from
     dE0bydT is %4.2e and\nStd entropy values is %4.2e
     V",dS01,dS02)
```

Scilab code Exa 11.5 Equilibrium constant for reaction

```
1  ////
2  //Variable Declaration
3  E0 = +1.10                 //Std. electrode potential
     for Danniel cell, V
```

```
4                                   //Zn(s) + Cu++ ------>  Zn2+
                                        +  Cu
5 T = 298.15                        //V/K
6 F = 96485                         //Faraday constant, C/mol
7 n = 2.
8 R = 8.314                         //Gas constant, J/(mol.K)
9
10 //Calculations
11 K = exp(n*F*E0/(R*T))
12
13 //Results
14 printf("\n Equilibrium constant for reaction is %4.2
      e",K)
```

Scilab code Exa 11.6 Equilibrium constant for reaction

```
1 ////Variable Declaration
2 E = +0.29                         //Cell emf, V
3 n = 2.
4
5 //Calculations
6 Ksp = 10**(-n*E/0.05916)
7
8 //Results
9 printf("\n Equilibrium constant for reaction is %4.2
      e",Ksp)
```

Scilab code Exa 11.8 Au has positive cell potential of

```
1 ////Variable Declaration
2 E = +1.51                         //EMF for reduction of
      permangnet, V
```

```
E01 = -0.7618            //Zn2+ + 2e- ---------> Zn (s)
E02 = +0.7996            //Ag+ + e- ---------> Ag (s)
E03 = +1.6920            //Au+ + e- ---------> Au (s)

//Calculations
EZn = E - E01
EAg = E - E02
EAu = E - E03

[Er] = ({EZn,EAg,EAu})
//Results
printf("\n Cell potentials for Zn, Ag, Au are %4.2f V, %4.2f V, and %4.2f V",EZn, EAg,EAu)
printf("\n Zn has positive cell potential of %4.3f V and Can be oxidized bypermangnate ion",EZn)
printf("\n Ag has positive cell potential of %4.3f V and Can be oxidized bypermangnate ion",EAg)
printf("\n Au has positive cell potential of %4.3f V and Can be oxidized bypermangnate ion",EAu)
```

Chapter 12

Probability

Scilab code Exa 12.1 Probability of picking up any one ball

```
1 ////Varible declaration
2
3 Prob = 0
4 for x = 1:51
5     Prob = 1/(x) + Prob
6 end
7 Prob1=1.0
8 //Results
9 printf("\n Probability of picking up any one ball is
    %3.1f",Prob1)
```

Scilab code Exa 12.2 Probability of one heart card picked from a std stack of card

```
1 ////
2 //Variable Declaration
3 n = 52          //Total cards
4 nheart = 13     //Number of cards with hearts
5
```

```
6 //Calculations
7 Pe = (nheart/n)
8
9 //Results
10 printf("\n Probability of one (heart)card picked
   from a std. stack of %d cards is %f",n,Pe)
```

Scilab code Exa 12.3 Total number of Five card arrangment from a deck of 52 cards

```
1 ////Variable Declaration
2 n = 52          //Total cards
3
4 //Calculations
5 TotalM = n*(n-1)*(n-2)*(n-3)*(n-4)
6 //Results
7 printf("\n Total number of Five card arrangment from
    a deck of 52 cards is %d",TotalM)
```

Scilab code Exa 12.4 Possible spin states for excited state

```
1 ////Variable Declaration
2 n1 = 2          //Two spin states for 1st electron in
    orbit 1
3 n2 = 2          //Two spin states for 2nd electron in
    orbit 2
4
5 //Calculation
6 M = n1*n1
7
8 //Results
9 printf("\n Possible spin states for excited state
   are %2d",M)
```

Scilab code Exa 12.5 Maximum Possible permutations for 5 player to play

```
////
//Variable Declaration
n = 12          //Total Number of players
j = 5           //Number player those can play match

//Calculation
P = factorial(n)/factorial(n-j)

//Results
printf("\n Maximum Possible permutations for 5
   player to play are %8d",P)
```

Scilab code Exa 12.6 Maximum Possible 5 card combinations

```
////
//Variable Declaration
n = 52          //Number of cards in std . pack
j = 5           //Number of cards in subset

//Calculation
C = factorial(n)/(factorial(j)*factorial(n-j))

//Results
printf("\n Maximum Possible 5-card combinations are
   %8d",C)
```

Scilab code Exa 12.7 Total number of quantum states

```
////
//Variable Declaration
x = 6           //Number of electrons
n = 2           //Number of states

//Calculation
P = factorial(x)/(factorial(n)*factorial(x-n))

//Results
printf("\n Total number of quantum states are %3d",P
   )
```

Scilab code Exa 12.8 Probability of getting 10 head out of 50 tossing

```
//////
//Variable Declaration
n = 50          //Number of separate experiments
j1 = 25          //Number of sucessful expt with
   heads up
j2 = 10          //Number of sucessful expt with
   heads up

//Calculation
C25 = factorial(n)/(factorial(j1)*factorial(n-j1))
PE25 = (1/2)**j1
PEC25 = (1-(1/2))**(n-j1)
P25 = C25*PE25*PEC25

C10 = factorial(n)/(factorial(j2)*factorial(n-j2))
PE10 = (1/2)**j2
PEC10 = (1-(1/2))**(n-j2)
P10 = C10*PE10*PEC10

//Results
printf("\n Probability of getting 25 head out of 50
```

```
    tossing is %4.3f",P25)
20
21 printf("\n Probability of getting 10 head out of 50
    tossing is %4.3e",P10)
```

Scilab code Exa 12.9 sterling

```
1  //////Variable Declaration
2  N = [10,50,100]          //Valures for N
3
4  //Calculations
5  printf("\n    N          ln(N!)          ln(N!)sterling
          Error')
6  for i =10
7
8   lnN = log(factorial(i))
9      lnNs = i*log(i)-i
10     err = abs(lnN-lnNs)
11     printf('\n%3d          %5.2f          %5.2f
                        %4.2f',i,lnN,lnNs, err)
12 end
13 for i =50
14
15  lnN = log(factorial(i))
16     lnNs = i*log(i)-i
17     err = abs(lnN-lnNs)
18     printf('\n%3d          %5.2f          %5.2f
                        %4.2f',i,lnN,lnNs, err)
19 end
20 for i =100
21
22  lnN = log(factorial(i))
23     lnNs = i*log(i)-i
24     err = abs(lnN-lnNs)
25     printf('\n%3d          %5.2f          %5.2f
```

```
                    %4.2f',i,lnN,lnNs, err)
26 end
```

Scilab code Exa 12.10 Probability of receiving any card

```
1 ////
2 //Variable Declaration
3 fi = 1          //Probability of receiving any card
4 n = 52          //Number od Cards
5
6 //Calculations
7 sum = 0
8 for i = 1:52
9     sum = sum + fi
10 end
11 Pxi = (fi/sum)
12
13 //Results
14 printf("\n Probability of receiving any card is %f",
       Pxi)
```

Scilab code Exa 12.12 Average payout

```
1 ////
2 //Variable Declaration
3 //r = Symbol('r')       //Radius of inner circle
4 C = list(5,2,0)
5 //Calculations
6 A1 = %pi
7 A2 = %pi*(2)**2 - A1
8 A3 = %pi*(3)**2 - (A1 + A2)
9 At = A1 + A2 + A3
10 f1 = A1/At
```

```
11 f2 = A2/At
12 f3 = A3/At
13 sf = f1 + f2 + f3
14
15 ns = (f1*C(1)+f2*C(2)+f3*C(3))/sf
16
17 //Results
18 printf("\n A1, A2, A3:%f*r**2 %f*r**2 %f*r**2   ", A1
      , A2, A3)
19 printf("\n f1, f2, f3: %f %f %f", f1,f2,f3)
20 printf("\n Average payout $ %f   ",((ns)))
```

Chapter 13

Boltzmann Distribution

Scilab code Exa 13.1 The observed weight

```
1 ////
2 //Variable Declaration
3
4 aH = 40          //Number of heads
5 N = 100          //Total events
6
7 //Calculations
8 aT = 100 - aH
9 We = factorial(N)/(factorial(aT)*factorial(aH))
10 Wexpected = factorial(N)/(factorial(N/2)*factorial(N
   /2))
11
12 //Results
13 printf("\n The observed weight %5.2e compared to %5
   .2e",We,Wexpected)
```

Scilab code Exa 13.3 Probability of finding an oscillator at energy level of n 3

```
////Variable Declaration
p0 = 0.633          //Probabilities of Energy level
   1,2,3
p1 = 0.233
p2 = 0.086

//Calculation
p4 = 1. -(p0+p1+p2)

//Results
printf("\n Probability of finding an oscillator at
   energy level of n>3 is %4.3f i.e.%4.1f percent",
   p4,p4*100)
```

Scilab code Exa 13.4 Probability of finding an oscillator at energy level of n 3

```
////Variable Declaration
p0 = 0.394          //Probabilities of Energy level
   1,2,3
p1by2 = 0.239
p2 = 0.145

//Calculation
p4 = 1. -(p0+p1by2+p2)

//Results
printf("\n Probability of finding an oscillator at
   energy level of n>3 is %4.3f",p4)
```

Scilab code Exa 13.5 Probability of occupying the second vibrational state n equal

```
////
//Variable Declaration
```

```
I2 = 208          //Vibrational frequency, cm-1
T = 298           //Molecular Temperature, K
c = 3.00e10       //speed of light, cm/s
h = 6.626e-34     //Planks constant, J/K
k = 1.38e-23      //Boltzman constant, J/K
//Calculation
q = 1./(1.-exp(-h*c*I2/(k*T)))
p2 = exp(-2*h*c*I2/(k*T))/q

//Results
printf("\n Partition function is %4.3f",q)

printf("\n Probability of occupying the second
   vibrational state n=2 is %4.3f",p2)
```

Scilab code Exa 13.6 Occupation Number

```
////Variable Declaration
B = 1.45          //Magnetic field streangth, Teslas
T = 298           //Molecular Temperature, K
c = 3.00e10       //speed of light, cm/s
h = 6.626e-34     //Planks constant, J/K
k = 1.38e-23      //Boltzman constant, J/K
gnbn = 2.82e-26   //J/T
//Calculation
ahpbyahm = exp(-gnbn*B/(k*T))

//Results
printf("\n Occupation Number is %7.6f",ahpbyahm)
```

Chapter 14

Ensemble and Molecular Partition Function

Scilab code Exa 14.1 Difference in energy levels

```
1 ////Variable Declarations
2 h = 6.626e-34       //Planks constant, J.s
3 k = 1.38e-23        //Boltzman constant, J/K
4 c = 3.0e8           //speed of light, m/s
5 l = 0.01            //Box length, m
6 n2 =1
7 n1 = 2          //Energy levels states
8 m = 5.31e-26        //mass of oxygen molecule, kg
9
10 //Calculations
11 dE = (n1+n2)*h**2/(8*m*l**2)
12 dEcm = dE/(h*c*1e2)
13 //Results
14 printf("\n Difference in energy levels is %3.2e J or
    %3.2e 1/cm",dE,dEcm)
```

Scilab code Exa 14.2 Thermal wave length

```
////
//Variable Declarations
h = 6.626e-34       //Planks constant, J.s
k = 1.38e-23        //Boltzman constant, J/K
c = 3.0e8           //speed of light, m/s
v = 1.0             //Volume, L
T = 298.0           //Temeprature of Ar, K
m = 6.63e-26        //Mass of Argon molecule, kg

//Calculations
GAMA = h/sqrt(2*%pi*m*k*T)
v = v*1e-3
qT3D = v/GAMA**3

//Results
printf("\n Thermal wave length is %3.2e m and\
   nTranslational partition function is %3.2e",GAMA,
   qT3D)
```

Scilab code Exa 14.4 Spectrum will be observed at

```
////Variable Declarations
h = 6.626e-34       //Planks constant, J.s
k = 1.38e-23        //Boltzman constant, J/K
c = 3.0e8           //speed of light, m/s

J = 4               //Rotational energy level
B = 8.46            //Spectrum, 1/cm

//Calculations
T = (2*J+1)**2*h*c*100*B/(2*k)
//Results
printf("\n Spectrum will be observed at %4.0f K",T)
```

Scilab code Exa 14.5 Rotation partition function of H2

```
////
//Variable Declarations
h = 6.626e-34       //Planks constant, J.s
k = 1.38e-23        //Boltzman constant, J/K
c = 3.0e8           //speed of light, m/s

B = 60.589          //Spectrum for H2, 1/cm
T = 1000            //Temperture of Hydrogen, K
//Calculations
qR = k*T/(2*h*c*100*B)
qRs = 0.0
//for J in range(101):
//    print J
//    if (J%2 == 0):
//        qRs = qRs + (2*J+1)*exp(-h*c*100*B*J*(J+1)
   /(k*T)
//    else:
//        qRs = qRs + 3*(2*J+1)*exp(-h*c*100*B*J*(J
   +1)/(k*T))
//print qRs/4

//Results
printf("\n Rotation partition function of H2 at %4.0
   f is %4.3f",T,qR)
```

Scilab code Exa 14.6 Rotation partition function of H2

```
////Variable Declarations
h = 6.626e-34       //Planks constant, J.s
```

```
3 k = 1.38e-23          //Boltzman constant, J/K
4 c = 3.0e8             //speed of light, m/s
5 B = 0.0374            //Spectrum for H2, 1/cm
6 T = 100.0             //Temperture of Hydrogen, K
7 sigma = 2.
8
9 //Calculations
10 ThetaR = h*c*100*B/k
11 qR = T/(sigma*ThetaR)
12
13 //Results
14 printf("\n Rotation partition function of H2 at %4.0
     f K is %4.3f",T,qR)
```

Scilab code Exa 14.7 Rotation partition function for OCS ONCI CH2O

```
1 ////
2 //Variable Declarations
3 h = 6.626e-34         //Planks constant, J.s
4 k = 1.38e-23          //Boltzman constant, J/K
5 c = 3.0e8             //speed of light, m/s
6 Ba = 1.48                          //Spectrum for OCS,
      1/cm
7 Bb = list(2.84,0.191,0.179)           //Spectrum for
     ONCI, 1/c
8 Bc = list(9.40,1.29,1.13)             //Spectrum for
     CH2O, 1/cm
9 T = 298.0                          //Temperture of
     Hydrogen, K
10 sigmab = 1
11 sigmac = 2
12
13 //Calculations
14 qRa = k*T/(h*c*100*Ba)
15 qRb = (sqrt(%pi)/sigmab)*(k*T/(h*c*100))**(3./2)*
```

```
    sqrt(1/Bb(1))*sqrt(1/Bb(2))*sqrt(1/Bb(3))
16 qRc = (sqrt(%pi)/sigmac)*(k*T/(h*c*100))**(3./2)*
    sqrt(1/Bc(1))*sqrt(1/Bc(2))*sqrt(1/Bc(3))
17
18 //Results
19 printf("\n Rotation partition function for OCS, ONCI
    , CH2O at %4.0f K are %4.0f, %4.0f, and %4.0f
    respectively",T,qRa,qRb,qRc)
```

Scilab code Exa 14.8 Vibrational partition function for I2

```
1 ////
2 //Variable Declarations
3 h = 6.626e-34       //Planks constant, J.s
4 k = 1.38e-23        //Boltzman constant, J/K
5 c = 3.0e8           //speed of light, m/s
6
7 Ba = 1.48                          //Frequency for OCS
    , 1/cm
8 Bb = [2.84,0.191,0.179]            //Frequency for
    ONCI, 1/cm
9 Bc = [9.40,1.29,1.13]              //Frequency for
    CH2O, 1/cm
10 T298 = 298.0                      //Temperture of
    Hydrogen, K
11 T1000 = 1000                      //Temperture of
    Hydrogen, K
12 nubar  =  208
13
14 //Calculations
15 qv298 = 1./(1.-exp(-h*c*100*nubar/(k*T298)))
16 qv1000 = 1./(1.-exp(-h*c*100*nubar/(k*T1000)))
17
18 //Results
19 printf("\n Vibrational partition function for I2 at
```

```
%4d and %4d are %4.2f K and %4.2f respectively",
T298, T1000,qv298, qv1000)
```

Scilab code Exa 14.9 Total Vibrational partition function for OClO

```
////
//Variable Declarations
h = 6.626e-34       //Planks constant, J.s
k = 1.38e-23        //Boltzman constant, J/K
c = 3.0e8           //speed of light, m/s

T = 298             //Temeprature, K
nubar = list(450, 945, 1100)    //Vibrational mode
   frequencies for OClO, 1/cm

//Calculations
Qv = 1.
for i = nubar
    qv = 1/(1.-exp(-h*c*100*i/(k*T)))
    printf("\nAt %4.0f 1/cm the q = %4.3f',i,qv)
    Qv = Qv*qv
   end
//Results
printf("\n Total Vibrational partition function for
   OClO at %4.1f K is %4.3f",T, Qv)
```

Scilab code Exa 14.10 Vibrational partition function for F2

```
////
//Variable Declarations
h = 6.626e-34       //Planks constant, J.s
k = 1.38e-23        //Boltzman constant, J/K
c = 3.0e8           //speed of light, m/s
```

```
6 T = 298                //Temeprature, K
7 nubar = 917            //Vibrational mode frequencies
     for F2, 1/cm
8
9 //Calculations
10 ThetaV = h*c*100*nubar/k
11 Th = 10*ThetaV
12 qv = 1/(1.-exp(-ThetaV/Th))
13
14 //Results
15 printf("\n Vibrational partition function for F2 at
     %4.1f K is %4.3f",T, qv)
```

Scilab code Exa 14.11 Total Vibrational partition function for OClO

```
1 ////
2 //Variable Declarations
3 h = 6.626e-34          //Planks constant, J.s
4 k = 1.38e-23           //Boltzman constant, J/K
5 c = 3.0e8              //speed of light, m/s
6 T = 1000               //Temeprature, K
7 nubar = [1388, 667.4,667.4,2349]   //Vibrational
     mode frequencies for CO2, 1/cm
8
9 //Calculations
10 Qv = 1
11 for i = [1388, 667.4,667.4,2349]
12     qv = 1/(1.-exp(-h*c*100*i/(k*T)))
13   printf("\nAt %4.0f 1/cm the q = %4.3f",i,qv)
14     Qv = Qv*qv
15 //Results
16 end
17 printf("\n Total Vibrational partition function for
     OClO at %4.1f K is %4.3f",T, Qv)
```

Scilab code Exa 14.12 Electronic partition function for F2

```
////
//Variable Declarations
h = 6.626e-34       //Planks constant, J.s
k = 1.38e-23        //Boltzman constant, J/K
c = 3.0e8           //speed of light, m/s
T = 298.            //Temeprature, K
n = [0,1,2,3,4,5,6,7,8]    //Energy levels
E0 = list
   (0,137.38,323.46,552.96,2112.28,2153.21,2220.11,2311.36,2424.78)
       //Energies, 1/cm
g0 = list(4,6,8,10,2,4,6,8,10)

//Calculations
qE = 0.0
for i = 1:9
    a =g0(i)*exp(-h*c*100*E0(i)/(k*T))
    qE = qE + a
end
//Results
printf("\n Electronic partition function for F2 at
   %4.1f K is %4.2f",T, qE)
```

Chapter 15

Statistical Thermodyanamics

Scilab code Exa 15.2 For Internal energy to be

```
1 ////
2 //Variable Declaration
3 U = 1.00e3                //Total internal energy, J
4 hnu = 1.00e-20            //Energy level separation, J
5 NA = 6.022e23             //Avagadro's Number, 1/mol
6 k = 1.38e-23              //Boltzmann constant, J/K
7 n = 1                     //Number of moles, mol
8
9 //Calcualtions
10 T = hnu/(k*log(n*NA*hnu/U-1.))
11
12 //Results
13 printf("\n For Internal energy to be %4.1f J
     temperature will be %4.1f K",U,T)
```

Scilab code Exa 15.3 Electronic contribution to internal enrgy

```
1 ////
```

```
2 //Variable Declaration
3 g0 = 3.0
4 labda = 1263e-9          //Wave length in nm
5 T = 500.                 //Temperature, K
6 c = 3.00e8               //Speed of light, m/s
7 NA = 6.022e23            //Avagadro's Number, 1/mol
8 k = 1.38e-23             //Boltzmann constant, J/K
9 n = 1.0                  //Number of moles, mol
10 h = 6.626e-34           //Planks's Constant, J.s
11
12 //Calcualtions
13 beta = 1/(k*T)
14 eps = h*c/labda
15 qE = g0 + exp(-beta*eps)
16 UE = n*NA*eps*exp(-beta*eps)/qE
17
18 //Results
19 printf("\n Energy of excited state is %4.2e J",eps)
20
21 printf("\n Electronic partition function qE is %4.3e
     ",qE)
22
23 printf("\n Electronic contribution to internal enrgy
     is %4.3e J",UE)
```

Scilab code Exa 15.5 Std Molar entropy for Kr

```
1 ////
2 //Variable Declaration
3 Mne = 0.0201797          //Molecular wt of ne, kg/mol
4 Mkr = 0.0837980          //Molecular wt of kr, kg/mol
5 Vmne = 0.0224            //Std. state molar volume of
     ne, m3
6 Vmkr = 0.0223            //Std. state molar volume of
     kr, m3
```

```
7  h = 6.626e-34          //Planks's Constant, J.s
8  NA = 6.022e23          //Avagadro's Number, 1/mol
9  k = 1.38e-23           //Boltzmann constant, J/K
10 T = 298                //Std. state temeprature,K
11 R = 8.314              //Ideal gas constant, J/(mol.K
       )
12 n = 1.0                //Number of mole, mol
13
14 //Calcualtions
15 mne = Mne/NA
16 mkr = Mkr/NA
17 Labdane = sqrt(h**2/(2*%pi*mne*k*T))
18 Labdakr = sqrt(h**2/(2*%pi*mkr*k*T))
19 Sne = 5.*R/2 + R*log(Vmne/Labdane**3)-R*log(NA)
20 Skr = 5.*R/2 + R*log(Vmkr/Labdakr**3)-R*log(NA)
21
22 //Results
23 printf("\n Thermal wave lengths for Ne is %4.2e m3",
       Labdane)
24
25 printf("\n Std. Molar entropy for Ne is %4.2f J/(mol
       .K)",Sne)
26
27 printf("\n Thermal wave lengths for Kr is %4.2e m3",
       Labdakr)
28
29 printf("\n Std. Molar entropy for Kr is %4.2f J/(mol
       .K)",Skr)
```

Scilab code Exa 15.8 The Gibbs energy for 1 mol of Ar

```
1 ////
2 //Variable Declaration
3 M = 0.040               //Moleculat wt of Ar, kg/mol
4 h = 6.626e-34           //Planks's Constant, J.s
```

```
5  NA = 6.022e23           //Avagadro's Number, 1/mol
6  k = 1.38e-23            //Boltzmann constant, J/K
7  T = 298.15              //Std. state temeprature,K
8  P = 1e5                 //Std. state pressure, Pa
9  R = 8.314               //Ideal gas constant, J/(mol.K
       )
10 n = 1.0                 //Number of mole, mol
11
12 //Calcualtions
13 m = M/NA
14 Labda3 = (h**2/(2*%pi*m*k*T))**(3./2)
15 G0 = -n*R*T*log(k*T/(P*Labda3))
16
17 //Results
18 printf("\n Thermal wave lengths for Ne is %4.2e m3",
       Labda3)
19
20 printf("\n The Gibbs energy for 1 mol of Ar is %6.2f
       kJ",G0/1000)
```

Chapter 16

Kinetic Theory of Gases

Scilab code Exa 16.2 Maximum average root mean square speed of Ar

```
1 ////
2 //Variable Declaration
3 R = 8.314           //Ideal Gas Constant, J/(mol.K)
4 T = 298             //Temperatureof Gas, K
5 M = 0.040           //Molecular wt of Ar, kg/mol
6
7
8 //Calculations
9 vmp = sqrt(2*R*T/M)
10 vave = sqrt(8*R*T/(M*%pi))
11 vrms = sqrt(3*R*T/M)
12
13 //Results
14 printf("\n Maximum, average, root mean square speed
   of Ar\nat 298 K are %4.0f, %4.0f, %4.0f m/s",vmp,
   vave,vrms)
```

Scilab code Exa 16.4 Number of Collisions

```
////
//Variable Declaration
R = 8.314           //Ideal Gas Constant, J/(mol.K)
T = 298             //Temperature of Gas, K
M = 0.040           //Molecular wt of Ar, kg/mol
P = 101325          //Pressure, N/m2
NA = 6.022e23       //Number of particles per mol
V = 1.0             //Volume of Container, L

//Calculations
Zc = P*NA/sqrt(2*%pi*R*T*M)
Nc = Zc
//Results
printf("\n Number of Collisions %4.2e  per s",Nc)
```

Scilab code Exa 16.5 Pressure after 1 hr of effusion

```
////
//Variable Declaration
R = 8.314           //Ideal Gas Constant, J/(mol.K)
T = 298             //Temperature of Gas, K
M = 0.040           //Molecular wt of Ar, kg/mol
P0 = 1013.25        //Pressure, N/m2
NA = 6.022e23       //Number of particles per mol
V = 1.0             //Volume of Container, L
k = 1.38e-23        //Boltzmann constant, J/K
t = 3600            //time of effusion, s
A = 0.01            //Area, um2

//Calculations
A = A*1e-12
V = V*1e-3
expo = (A*t/V)*(k*T/(2*%pi*M/NA))
P = P0*exp(-expo)
//Results
```

```
19 printf("\n Pressure after 1 hr of effusion is %4.3e
     Pa",P/101325)
```

Scilab code Exa 16.6 Single particle collisional frequency

```
1 ////
2 //Variable Declaration
3 R = 8.314           //Ideal Gas Constant, J/(mol.K)
4 T = 298             //Temperature of Gas, K
5 M = 0.044           //Molecular wt of CO2, kg/mol
6 P  = 101325         //Pressure, N/m2
7 NA = 6.022e23       //Number of particles per mol
8 sigm = 5.2e-19      //m2
9
10 //Calculations
11 zCO2 = (P*NA/(R*T))*sigm*sqrt(2)*sqrt(8*R*T/(%pi*M))
12 //Results
13 printf("\n Single particle collisional frequency is
     %4.1e per s",zCO2)
```

Scilab code Exa 16.7 Collisional frequency

```
1 ////
2 //Variable Declaration
3 R = 8.314           //Ideal Gas Constant, J/(mol.K)
4 T = 298             //Temperature of Gas, K
5 MAr = 0.04          //Molecular wt of Ar, kg/mol
6 MKr = 0.084         //Molecular wt of Kr, kg/mol
7 pAr  = 360          //Partial Pressure Ar, torr
8 pKr  = 400          //Partial Pressure Kr, torr
9 rAr = 0.17e-9       //Hard sphere radius of Ar, m
10 rKr = 0.20e-9      //Hard sphere radius of Kr, m
11 NA = 6.022e23      //Number of particles per mol
```

```
12 k = 1.38e-23       //Boltzmann constant, J/K
13
14 //Calculations
15 pAr = pAr*101325/760
16 pKr = pKr*101325/760
17 p1 = pAr*NA/(R*T)
18 p2 = pKr*NA/(R*T)
19 sigm = %pi*(rAr+rKr)**2
20 mu = MAr*MKr/((MAr+MKr)*NA)
21 p3 = sqrt(8*k*T/(%pi*mu))
22 zArKr = p1*p2*sigm*p3
23
24 //Results
25 printf("\n Collisional frequency is %4.2e m-3s-1",
      zArKr)
```

Chapter 17

Transport Phenomena

Scilab code Exa 17.1 Diffusion coefficient of Argon

```
1 //////
2 //Variable Declaration
3 M = 0.040                    //Molecualar wt of Argon,
      kh/mol
4 P = 101325.0     //Pressure and Temperature, Pa, K
5  T = 298.0
6 sigm = 3.6e-19  //
7 R = 8.314                    //Molar Gas constant,
     mol^-1 K^-1
8 N_A = 6.02214129e+23         //mol^-1
9 //Calculations
10 DAr = (1./3)*sqrt(8*R*T/(%pi*M))*(R*T/(P*N_A*sqrt(2)
     *sigm))
11
12 //Results
13 printf("\n Diffusion coefficient of Argon %3.1e m2/s
     ",DAr)
```

Scilab code Exa 17.2 Ratio of collision cross sections of Helium to Argon

```
////
//Variable Declaration
DHebyAr = 4.0
MAr = 39.9          //Molecualar wt of Argon and Neon,
   kg/mol
MHe = 4.0
P   = 101325.0      //Pressure and Temperature, Pa, K
T   =  298.0
sigm = 3.6e-19  //
R = 8.314                       //Molar Gas constant,
   molˆ-1 Kˆ-1
N_A = 6.02214129e+23            //molˆ-1
//Calculations
sigHebyAr = (1./DHebyAr)*sqrt(MAr/MHe)

//Results
printf("\n Ratio of collision cross sections of
   Helium to Argon %4.3f",sigHebyAr)
```

Scilab code Exa 17.3 rms displacement

```
////
//Variable Declaration
D = 1.0e-5                      //Diffusion coefficient,
   m2/s
t1 = 1000                       //Time, s
t10 = 10000                     //Time, s

//Calculations
xrms1 = sqrt(2*D*t1)
xrms10 = sqrt(2*D*t10)

//Results
printf("\n rms displacement at %4d and %4d is  %4.3f
    and %4.3f m respectively",t1,t10,xrms1,xrms10)
```

Scilab code Exa 17.4 Time per random walk

```
1 ////Variable Declaration
2 D = 2.2e-5                    //Diffusion coefficient
     of benzene, cm2/s
3 x0 = 0.3                      //molecular diameter of
     benzene, nm
4
5 //Calculations
6 t = (x0*1e-9)**2/(2*D*1e-4)
7
8 //Results
9 printf("\n Time per random walk is %4.3e s or %4.2f
     ps",t,t/1e-12)
```

Scilab code Exa 17.5 Mean free path

```
1 ////
2 //Variable Declaration
3 P = 101325                     //Pressure, Pa
4 kt = 0.0177                    //Thermal conductivity,
     J/(K.m.s)
5 T = 300.0                      //Temperature, K
6 k = 1.3806488e-23              //Boltzmanconstant,J K
     ^-1
7 sigm = 3.6e-19  //
8 R = 8.314                      //Molar Gas constant,
     mol^-1 K^-1
9 NA = 6.02214129e+23            //mol^-1
10 M = 39.9                      //Molecualar wt of Argon
     and Neon, kg/mol
```

```
11
12 //Calculations
13 CvmbyNA = 3.*k/2
14 nuavg = sqrt(8*R*T/(%pi*M*1e-3))
15 N = NA*P/(R*T)
16 labda = 3*kt/(CvmbyNA*nuavg*N)
17 sigm = 1/(sqrt(2)*N*labda)
18
19 //Results
20 printf("\n Mean free path %4.3e m and collisional
      cross section %4.2e m2",labda, sigm)
```

Scilab code Exa 17.6 Collisional cross section

```
1 ////
2 //Variable Declaration
3 eta = 227.                    //Viscosity of Ar, muP
4 P = 101325                    //Pressure, Pa
5 kt = 0.0177                   //Thermal conductivity,
      J/(K.m.s)
6 T = 300.0                     //Temperature, K
7 k = 1.3806488e-23             //Boltzmanconstant,J K
      ^-1
8 R = 8.314                     //Molar Gas constant,
      mol^-1 K^-1
9 NA = 6.02214129e+23           //mol^-1
10 M = 39.9                     //Molecualar wt of Argon
      and Neon, kg/mol
11
12 //Calculations
13 nuavg = sqrt(8*R*T/(%pi*M*1e-3))
14 N = NA*P/(R*T)
15 m = M*1e-3/NA
16 labda = 3.*eta*1e-7/(nuavg*N*m)          //viscosity
      in kg m s units
```

```
17 sigm = 1./(sqrt(2)*N*labda)
18
19 //Results
20 printf("\n Collisional cross section %4.2e m2",sigm)
```

Scilab code Exa 17.7 Cylinder can be used for

```
1 ////
2 //Variable Declaration
3 m = 22.7                          //Mass of CO2, kg
4 T = 293.0                         //Temperature, K
5 L = 1.0                           //length of the tube, m
6 d = 0.75                          //Diameter of the tube,
     mm
7 eta = 146                         //Viscosity of CO2, muP
8 p1 = 1.05                         //Inlet pressure, atm
9 p2 = 1.00                         //Outlet pressure, atm
10 atm2pa = 101325                  //Conversion for
     pressure from atm to Pa
11 M = 0.044                        //Molecular wt of CO2,
     kg/mol
12 R = 8.314                        //Molar Gas constant, J
     mol^-1 K^-1
13
14 //Calculations
15 p1 = p1*atm2pa
16 p2 = p2*atm2pa
17 F = %pi*(d*1e-3/2)**4*(p1**2-p2**2)/(16.*eta/1.e7*L*
     p2)
18 nCO2 = m/M
19 v = nCO2*R*T/((p1+p2)/2)
20 t = v/F
21
22 //Results
23 printf("\n Flow rate is %4.3e m3/s",F)
```

```
24
25 printf("\n Cylinder can be used for %4.3e s nearly
     %3.1f days",t, t/(24*3600))
```

Scilab code Exa 17.8 Radius of protein

```
1 ////
2 //Variable Declaration
3 eta = 0.891                    //Viscosity of
     hemoglobin in water, cP
4 T = 298.0                      //Temperature, K
5 k = 1.3806488e-23              //Boltzmanconstant,J K
     ^-1
6 R = 8.314                      //Molar Gas constant,
     mol^-1 K^-1
7 D = 6.9e-11                    //Diffusion coefficient,
      m2/s
8
9 //Calculations
10 r = k*T/(6*%pi*eta*1e-3*D)
11
12 //Results
13 printf("\n Radius of protein is %4.3f nm",r/1e-9)
```

Scilab code Exa 17.9 Radius of Lysozyme particle

```
1 ////
2 //Variable Declaration
3 s = 1.91e-13                   //Sedimentation constant
     , s
4 NA = 6.02214129e+23            //mol^-1
5 M = 14100.0                    //Molecualr wt of
     lysozyme, g/mol
```

```
rho = 0.998                    //Density of water, kg/
   m3
eta = 1.002                    //Viscosity lysozyme in
   water, cP
T = 293.15                     //Temperature, K
vbar = 0.703                   //Specific volume of cm3
   /g

//Calculations
m = M/NA
f = m*(1.-vbar*rho)/s
r = f/(6*%pi*eta)

//Results
printf("\n Radius of Lysozyme particle is %4.3f nm",
   r/1e-9)
```

Scilab code Exa 17.11 Molar conductivity of MgCl2 on infinite dilution

```
////Variable Declaration
LMg = 0.0106                   //Ionic conductance for
   Mg, S.m2/mol
LCl = 0.0076                   //Ionic conductance for
   Cl, S.m2/mol
[nMg,nCl] = (1,2)

//Calculations
LMgCl2 = nMg*LMg + nCl*LCl

//Results
printf("\n Molar conductivity of MgCl2 on infinite
   dilution is %5.4f S.m2/mol",LMgCl2)
```

Chapter 18

Elementary Chemical Kinetics

Scilab code Exa 18.2 Rate constant of the reaction

```
1 ////
2 //Variable Declaration
3 Ca0 = list(2.3e-4,4.6e-4,9.2e-4)        //Initial
     Concentration of A, M
4 Cb0 = list(3.1e-5,6.2e-5,6.2e-5)        //Initial
     Concentration of B, M
5 Ri  = list(5.25e-4,4.2e-3,1.68e-2)      //Initial
     rate of reaction, M
6
7 //Calculations
8 alp = log(Ri(2)/Ri(3))/log(Ca0(2)/Ca0(3))
9 beta = (log(Ri(1)/Ri(2)) - 2*log((Ca0(1)/Ca0(2))))/(
     log(Cb0(1)/Cb0(2)))
10 k = Ri(3)/(Ca0(3)**2*Cb0(3)**beta)
11
12 //REsults
13 printf("\n Order of reaction with respect to
     reactant A: %3.2f",alp)
14
15 printf("\n Order of reaction with respect to
     reactant A: %3.2f",beta)
```

```
16
17 printf("\n Rate constant of the reaction: %4.3e 1./(
     M.s)",k)
```

Scilab code Exa 18.3 Timerequire for 60 percent decay of N2O5

```
1 ////
2 //Variable Declaration
3 t1by2 = 2.05e4          //Half life for first order
     decomposition of N2O5, s
4 x = 60.                 //percentage decay of N2O5
5
6 //Calculations
7 k = log(2)/t1by2
8 t = -log(x/100)/k
9
10 //REsults
11 printf("\n Rate constant of the reaction: %4.3e 1/s"
     ,k)
12
13 printf("\n Timerequire for 60 percent decay of N2O5:
     %4.3e s",t)
```

Scilab code Exa 18.4 Timerequire for 60 percent decay of N2O5

```
1 ////
2 //Variable Declaration
3 t1by2 = 2.05e4          //Half life for first order
     decomposition of N2O5, s
4 x = 60.                 //percentage decay of N2O5
5
6 //Calculations
7 k = log(2)/t1by2
```

```
8 t = -log(x/100)/k
9
10 //REsults
11 printf("\n Rate constant of the reaction: %4.3e 1/s"
      ,k)
12
13 printf("\n Time required for 60 percent decay of
      N2O5: %4.3e s",t)
```

Scilab code Exa 18.5 Time required for maximum concentration of A

```
1 ////
2 //Variable Declaration
3 kAbykI = 2.0          //Ratio of rate constants
4 kA = 0.1              //First order rate constant for
      rxn 1, 1/s
5 kI = 0.05             //First order rate constant for
      rxn 2, 1/s
6 //Calculations
7 tmax = 1/(kA-kI)*log(kA/kI)
8
9 //Results
10 printf("\n Time required for maximum concentration
      of A: %4.2f s",tmax)
```

Scilab code Exa 18.7 Percentage of Benzyl Penicillin that under acid catalyzed rea

```
1 ////
2 //Variable Declaration
3 T = 22.0              //Temperature of the reaction, C
4 k1 = 7.0e-4           //Rate constants for rxn 1, 1/s
5 k2 = 4.1e-3           //Rate constant for rxn 2, 1/s
6 k3 = 5.7e-3           //Rate constant for rxn 3, 1/s
```

```
7 //Calculations
8 phiP1 = k1/(k1+k2+k3)
9
10 //Results
11 printf("\n Percentage of Benzyl Penicillin that
   under acid catalyzed reaction by path 1: %4.2f ",
   phiP1*100)
```

Scilab code Exa 18.9 Apperent Rate constant

```
1 ////
2 //Variable Declaration
3 Ea = 42.e3        //Activation energy for reaction, J/
   mol
4 A  = 1.e12        //Pre-exponential factor for
   reaction, 1/s
5 T = 298.0         //Temeprature, K
6 Kc = 1.0e4        //Equilibrium constant for reaction
7 R = 8.314         //Ideal gas constant, J/(mol.K)
8 //Calculations
9 kB = A*exp(-Ea/(R*T))
10 kA = kB*Kc
11 kApp = kA + kB
12
13 //Results
14 printf("\n Forward Rate constant is %4.2e 1/s",kA)
15
16 printf("\n Backward Rate constant is %4.2e 1/s",kB)
17
18 printf("\n Apperent Rate constant is %4.2e 1/s",kApp
   )
```

Scilab code Exa 18.10 Estimated rate

```
//////Variable Declaration
Dh = 7.6e-7      //Diffusion coefficient of
   Hemoglobin, cm2/s
Do2 = 2.2e-5     //Diffusion coefficient of oxygen,
   cm2/s
rh = 35.         //Radius of Hemoglobin, A
ro2 = 2.0        //Radius of Oxygen, A
k = 4e7          //Rate constant for binding of O2 to
    Hemoglobin, 1/(M.s)
NA =6.022e23     //Avagadro Number
//Calculations
DA = Dh + Do2
kd = 4*%pi*NA*(rh+ro2)*1e-8*DA

//Results
printf("\n Estimated rate %4.1e 1/(M.s is far grater
    than experimental value of %4.1e 1/(M.s, \nhence
    the reaction is not diffusion controlled",kd,k)
```

Scilab code Exa 18.11 Backward Rate constant

```
//////Variable Declaration
Ea = 104e3       //Activation energy for reaction, J/
   mol
A  = 1.e13       //Pre-exponential factor for
   reaction, 1/s
T = 300.0        //Temeprature, K
R = 8.314        //Ideal gas constant, J/(mol.K)
h = 6.626e-34    //Plnak constant, Js
c = 1.0          //Std. State concentration, M
k = 1.38e-23     //,J/K

//Calculations
dH = Ea - 2*R*T
dS = R*log(A*h*c/(k*T*%e**2))
```

```
13
14 //Results
15 printf("\n Forward Rate constant is %4.2e 1/s",dH)
16
17 printf("\n Backward Rate constant is %4.2f 1/s",dS)
```

Chapter 19

Complex Reaction Mechanism

Scilab code Exa 19.6 Overall quantum yield

```
1 ////Variable Declarations
2 mr = 2.5e-3          //Moles reacted, mol
3 P = 100.0            //Irradiation Power, J/s
4 t = 27               //Time of irradiation, s
5 h = 6.626e-34        //Planks constant, Js
6 c = 3.0e8            //Speed of light, m/s
7 labda = 280e-9       //Wavelength of light, m
8
9 //Calculation
10 Eabs = P*t
11 Eph = h*c/labda
12 nph = Eabs/Eph      //moles of photone
13 phi = mr/6.31e-3
14
15 //Results
16 printf("\n Total photon energy absorbed by sample %3
      .1e J",Eabs)
17
18 printf("\n Photon energy absorbed at 280 nm is %3.1e
      J",Eph)
19
```

```
20 printf("\n Total number of photon absorbed by sample
       %3.1e photones",nph)
21
22 printf("\n Overall quantum yield %4.2f",phi)
```

Scilab code Exa 19.7 Rate constant with barrier to electron transfer

```
1 //////Variable Declarations
2 r = 2.0e9           //Rate constant for electron
     transfer, per s
3 labda = 1.2         //Gibss energy change, eV
4 DG = -1.93          //Gibss energy change for 2-
     naphthoquinoyl, eV
5 k = 1.38e-23        //Boltzman constant, J/K
6 T = 298.0           //Temeprature, K
7 //Calculation
8 DGS = (DG+labda)**2/(4*labda)
9 k193 = r*exp(-DGS*1.6e-19/(k*T))
10 //Results
11 printf("\n DGS = %5.3f eV",DGS)
12
13 printf("\n Rate constant with barrier to electron
     transfer %3.2e per s",k193)
```