Scilab Textbook Companion for Chemical Engineering Thermodynamics by P. Ahuja[1]

Created by
Shashwat
B.Tech
Chemical Engineering
IIT BHU Varanasi
College Teacher
Prakash Kotecha
Cross-Checked by

August 10, 2013

[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: Chemical Engineering Thermodynamics

Author: P. Ahuja

Publisher: PHI Learning Private Limited, New Delhi

Edition: 1

Year: 2009

ISBN: 9788120336377

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

Introduction

Scilab code Exa 1.1 Calculation of pressure and heat transfer in piston cylinder assembly

```
clear;
clc;

//Example - 1.1
//Page number - 6
printf("Example - 1.1 and Page number - 6\n\n");

//(a)
// The pressure in the cylinder is due to the weight
    of the piston and due to surroundings pressure
m = 50;//[kg] - Mass of piston
A = 0.05;//[m^(2)] - Area of piston
g = 9.81;//[m/s^(2)] - Acceleration due to gravity
Po = 101325;//[N/m^(2)] - Atmospheric pressure
P = (m*g/A)+Po;//[N/m^(2)]
P = P/100000;//[bar]
printf(" (a).Pressure = %f bar\n",P);

//(b)
printf(" (b).Since the piston weight and
    surroundings pressure are the same,the gas
    pressure in the piston-cylinder assembly remains
    %f bar",P);
```

Scilab code Exa 1.2 Calculation of mass of air contained in a room

```
clear;
clc;

//Example - 1.2
//Page number - 8
printf("Example - 1.2 and Page number - 8\n\n");

// Given
P = 1;//[atm] - Atmospheric pressure
P = 101325;//[N/m^(2)]
R = 8.314;//[J/mol*K] - Universal gas constant
T = 30;//[C] - Temperature of air
T = 30+273.15;//[K]
V = 5*5*5;//[m^(3)] - Volume of the room

//The number of moles of air is given by
n = (P*V)/(R*T);//[mol]

//Molecular weight of air(21 vol% O2 and 79 vol% N2)
    =(0.21*32)+(0.79*28)= 28.84 g/mol
m = n*28.84;//[g]
m = m/1000;//[kg]
printf("The mass of air is , m = %f kg",m);
```

Scilab code Exa 1.3 Determination of work done

```
clear;
```

```
clc;

//Example - 1.3
//Page number - 13
printf("Example - 1.3 and Page number - 13\n\n");

// Given
P1 = 3;// [bar] - initial pressure
V1 = 0.5;// [m^(3)] - initial volume
V2 = 1.0;// [m^(3)] - final volume

//(a)
n = 1.5;

//Let P*V^(n)=C //Given relation
//W (work done per mole)= (integrate('P','V',V1,V2))
//W = (integrate('(C/V^(n))','V',V1,V2)) = (C*((V2
    ^*(1-n))-(V1^*(1-n))))/(1-n)
//Where  C=P*V^(n)=P1*V1^(n)=P2*V2^(n)
//Thus  w=((P2*V2^(n)*V2^(1-n))-(P1*V1^(n)*V1^(1-n))
    )/(1-n)
//w = ((P2*V2^(n))-(P1*V1^(n)))/(1-n)
//and thus   W=((P2*V2)-(P1*V1))/(1-n)
//The above expression is valid for all values of n,
    except n=1.0
P2 = (P1*((V1/V2)^(n)));//[bar] //pressure at state
    2

//we have,(V1/V2)=(V1t/(V2t),since the number  of
    moles are constant.Thus
W = ((P2*V2)-(P1*V1))/(1-n)*10^(5);//[J]
W = W/1000;//[kJ]
printf(" (a).The work done (for n=1.5) is %f kJ\n",W
    );

//(b)
//For n=1.0,we have, PV=C.
// w(wok done per mol)= (integrate('P','V',V1,V2)) =
   (integrate('C/V','V',V1,V2)) = C*ln(V2/V1)=P1*V1
  *ln(V2/V1)
W1 = P1*V1*log(V2/V1)*10^(5);//[J]
W1 = W1/1000;//[kJ]
printf(" (b).The work done (for n=1.0) is %f kJ\n",
  W1);

//(c)
//For n=0,we get P=Constant and thus
P = P1;//[bar]
// w =(integrate('P','V',V1,V2)) = P*(V2-V1)
W2 = P*(V2-V1)*10^(5);//[J]
W2 = W2/1000;//[kJ]
printf(" (c).The work done (for n=0) is %f kJ\n\n",
  W2);
```

Scilab code Exa 1.4 Determination of wind energy per unit mass and diameter of the wind turbine

```
clear;
clc;

//Example - 1.4
//Page number - 17
printf("Example - 1.4 and Page number - 17\n\n");

//(a)
//Given
V = 9;// [m/s] - velocity
d = 1;//[m] - diameter
A = 3.14*(d/2)^(2);//[m^(2)] - area
P = 1;//[atm] - pressure
P = 101325;// [N/m^(2)]
T = 300;//[K] - Temperature
R =  8.314;//[J/mol*K] - Universal gas constant
```

```

E = (V^(2))/2;//[J/kg]
printf(" (a).The wind energy per unit mass of air is
    %f J/kg\n",E);

//(b)
// Molecular weight of air(21 vol% O2 and 79 vol% N2
   )=(0.21*32)+(0.79*28)=  28.84 g/mol
M = 28.84*10^(-3);//[kg/mol]
r = (P*M)/(R*T);//[kg/m^(3)] - density
m = r*V*A;// [kg/s] - mass flow rate of air
pi = m*E;//[Watt] - power input
printf(" (b).The wind power input to the turbine is
   %f Watt",pi);
```

Scilab code Exa 1.5 Determination of temperature

```
clear;
clc;

//Example - 1.5
//Page number - 23
printf("Example - 1.5 and Page number - 23\n\n");

// Given
P = 1;// [bar] - atospheric pressure
P1guz = 0.75;// [bar] - gauze pressure in 1st
   evaporator
P2Vguz = 0.25;// [bar] - vaccum gauze pressure in 2
   nd evaporator
P1abs = P + P1guz;// [bar] - absolute pressure in 1
   st evaporator
P2abs = P - P2Vguz;// [bar] -absolute pressure in 2
   nd evaporator

```

```
//From saturated steam table as reported in the book
printf(" For P1abs (absolute pressure) = %f bar\n",
   P1abs);
printf(" The saturation temperature in first
   evaporator  is 116.04 C\n\n");
printf(" For P2abs (absolute pressure) = %f bar\n",
   P2abs);
printf(" The saturation temperature in second
   evaporator  is 91.76 C\n");
```

Scilab code Exa 1.6 Calculation of dryness fraction of steam

```
clear;
clc;

//Example - 1.6
//Page number - 23
printf("Example - 1.6 and Page number - 23\n\n");

// Given
V = 1;// [kg] - volume of tank
P = 10;// [bar] - pressure

//Here degree of freedom =1(C=1,P=2,threfore F=1)
//From steam table at 10 bar as reported in the book
V_liq = 0.001127;// [m^(3)/kg] - volume in liquid
   phase
V_vap = 0.19444;// [m^(3)/kg] - volume in vapour
   phase

//x*Vv=(1-x)*Vl // since two volumes are equal
x = (V_liq/(V_liq+V_vap));// [kg]
y = (1-x);//[kg]

printf(" Mass of saturated vapour is %f kg\n",x);
printf(" Mass of saturated liquid is %f kg",y);
```

Scilab code Exa 1.7 Determination of pressure mass and volume

```
clear;
clc;

//Example - 1.7
//Page number - 23
printf("Example - 1.7 and Page number - 23\n\n");

// Given
V = 1;// [m^(3)] - volume of tank
M = 10;// [m^(3)] - total mass
T = (90+273.15);//[K] - temperature

//From steam table at 90 C as reported in the book
//vapour pressure(pressure of rigid tank) = 70.14[
    kPa] = 0.7014[bar]
printf(" Pressure of tank = 0.7014 bar\n");

V_liq_sat=0.001036;// [m^(3)/kg] - saturated liquid
    specific volume
V_vap_sat=2.36056;// [m^(3)/kg] - saturated vapour
    specific volume

//1=(V_liq_sat*(10-x))+(V_vap_sat*x)
x = (1-(10*V_liq_sat))/(V_vap_sat-V_liq_sat);//[kg]
y = (10-x);//[kg]

printf(" The amount of saturated liquid is %f kg\n",
    y);
printf(" The amount of saturated vapour is %f kg \n"
    ,x);

```

```
z = y*V_liq_sat;//[m^(3)] - Volume of saturated
    liquid
w = x*V_vap_sat;//[m^(3)] - Volume of saturated
    vapour

printf(" Total volume of saturated liquid is %f m
    ^(3)\n",z);
printf(" Total volume of saturated vapour is %f m
    ^(3)",w);
```

Scilab code Exa 1.8 Determination of heat supplied

```
clear;
clc;

//Example - 1.8
//Page number - 24
printf("Example - 1.8 and Page number - 24\n\n");

// Given
V = 10;// [m^(3)] - volume of vessel
P_1 = 1;// [bar] - initial pressure
V_liq_sat = 0.05;// [m^(3)] - saturated liquid
    volume
V_gas_sat = 9.95;// [m^(3)] - saturated vapour
    volume

//At 1 bar pressure
V_liq_1 = 0.001043;// [m^(3/kg)] - specific
    saturated liquid volume
U_liq_1 = 417.33;// [kJ/kg] - specific internal
    energy
V_gas_1 = 1.69400;// [m^(3/kg)] - specific saturated
     vapour volume
U_gas_1 = 2506.06;// [kJ/kg]
```

```

M_liq_1 = V_liq_sat/V_liq_1;// [kg] - mass of
    saturated liqid
M_gas_1 = V_gas_sat/V_gas_1;// [kg] - mass of
    saturated vapour
M = (M_liq_1+M_gas_1);// [kg] - total mass
U_1t = (M_liq_1*U_liq_1)+(M_gas_1*U_gas_1);// [kJ] -
     initial internal energy
V_gas_2 = (V/M);//[m^(3/kg)]

//from steam table at 10 bar pressure as reported in
     the book
V_vap_2 = 0.19444;// [m^(3/kg)]
U_vap_2 = 2583.64;// [kJ/kg]

//from steam table at 11 bar pressure as reported in
     the book
V_vap_3 = 0.17753;//[m^(3/kg)]
U_vap_3 = 2586.40;//[kJ/kg]

//Now computing pressure when molar volume of
    saturated vapour=Vg_2
//By interpolation (P2-10)/(11-10)=(Vg_2-Vv_2)/(Vv_3
    -Vv_2)
P_2 = (((V_gas_2 - V_vap_2)/(V_vap_3 - V_vap_2)*1)
    +10);// [bar] - final pressure

//By interpolation calculating internal energy at
    state 2
//(P2-10)/(11-10)=(U2-Uv_2)/(Uv_3-Uv_2)
U_2 = (((P_2-10)/(11-10))*(U_vap_3 - U_vap_2))+
    U_vap_2;//[kJ/kg]
U_2t = U_2*M;//[kJ]
H = U_2t - U_1t;//[kJ] - Heat supplied
H = H/1000;//[MJ]

printf(" Total heat supplied is %f MJ',H);
// since volume is constant,no work is done by the
```

```
system and heat supplied is used in increasing
the internal energy of the system.
```

Scilab code Exa 1.9 Calculation of saturation temperature

```
clear;
clc;

//Example - 1.9
//Page number - 26
printf("Example - 1.9 and Page number - 26\n\n");

//Given
//Antoine equation for water   ln(Psat)
   =16.262-(3799.89/(T_sat + 226.35))
P = 2;//[atm] - Pressure
P = (2*101325)/1000;//[kPa]

P_sat = P;// Saturation pressure
T_sat = (3799.89/(16.262-log(P_sat)))-226.35;//[C] -
    Saturation temperature
//Thus boiling at 2 atm occurs at Tsat = 120.66 C.

//From steam tables, at 2 bar, Tsat = 120.23 C and at
   2.25 bar, Tsat = 124.0 C
//From interpolation for T_sat = 120.66 C,P = 2.0265
    bar
//For P_= 2.0265 bar, T_sat, from steam table by
   interpolation is given by
//((2.0265-2)/(2.25-2))=((Tsat-120.23)
   /(124.0-120.23))
T_sat_0 = (((2.0265-2)/(2.25-2))*(124.0-120.23))
   +120.23;//[C]

printf(" Saturation temperature (Tsat) = %f C which
   is close to %f C as determined from Antoine
   equation",T_sat_0,T_sat);
```

Scilab code Exa 1.10 Calculation of pressure and temperature at triple point

```
clear;
clc;

//Example - 1.10
//Page number - 27
printf("Example - 1.10 and Page number - 27\n\n");

//Given
// log(P)=-(1640/T)+10.56 (solid)
// log(P)=-(1159/T)+7.769 (liquid),where T is in K
// F+P=C+2, at triple point F+3=1+2 or,F=0 i.e,
    vapour pressure of liquid and solid at triple
    point are same,we get
// -(1640/T)+10.56 = -(1159/T)+7.769

T = (1640-1159)/(10.56-7.769);//[K]
P = 10^((-1640/T)+10.56);//[torr]

printf(" The temperature is %f K\n",T);
printf(" The pressure is %f torr (or mm Hg)",P);
```

Scilab code Exa 1.11 Determination of value of R Cp0 and Cv0

```
clear;
clc;

//Example - 1.11
```

```
//Page number - 29
printf("Example - 1.11 and Page number - 29\n\n");

//Given
M_O2 = 31.999;//molecular weight of oxygen
M_N2 = 28.014;//molecular weight of nitrogen
Y = 1.4;//molar heat capacities ratio for air

//Molecular weight of air(21 vol% O2 and 79 vol% N2)
   is given by
M_air = (0.21*M_O2)+(0.79*M_N2);//(vol% = mol%)

R = 8.314;//[J/mol*K] - Universal gas constant
R = (R*1/M_air);//[kJ/kg*K]

printf(" The value of universal gas constant (R) =
   %f kJ/kg-K \n",R);

//Y=Cp0/Cv0 and Cp0-Cv0=R
Cv_0 = R/(Y-1);//[kJ/kg*K]
Cp_0 = Y*Cv_0;//[kJ/kg*K]
printf(" The value of Cp_0 for air is %f kJ/kg-K\n',
   Cp_0);
printf(" The value of Cv_0 for air is %f kJ/kg-K',
   Cv_0);
```

Scilab code Exa 1.12 Calculation of molar heat capacity

```
clear;
clc;

//Example - 1.12
//Page number - 30
printf("Example - 1.12 and Page number - 30\n\n");

```

```
//Given
Y = 1.4;//molar heat capacities ratio for air
R = 8.314;// [J/mol*K] - Universal gas constant
Cv_0 = R/(Y-1);// [J/mol*K]
Cp_0 = Y*Cv_0;// [J/mol*K]

printf(" The molar heat capacity at constant volume
    (Cv_0) is %f J/mol-K\n',Cv_0);
printf(" The molar heat capacity at constant
    pressure (Cp_0) is %f J/mol-K',Cp_0);
```

Scilab code Exa 1.13 Determination of mean heat capacity

```
clear;
clc;

//Example - 1.13
//Page number - 30
printf("Example - 1.13 and Page number - 30\n\n");

//Given
// Cp0=7.7+(0.04594*10^(-2)*T)+(0.2521*10^(-5)*T^(2)
    )-(0.8587*10^(-9)*T^(3))
T_1 = 400;//[K]
T_2 = 500;//[K]

//(C)avg = q/(T_2 - T_1) = 1/(T_2 - T_1)*{(integrate
    ('C','T',T_1,T_2))}
//(Cp0)avg = 1/(T_2 - T_1)*{(integrate('Cp0','T',T_1
    ,T_2))}
Cp0_avg = (1/(T_2 - T_1))*integrate('
    7.7+(0.04594*10^(-2)*T)+(0.2521*10^(-5)*T^(2))
    -(0.8587*10^(-9)*T^(3))','T',T_1,T_2);

printf(" The mean heat capacity (Cp0_avg) for
    temerature range of 400 to 500 K is %f cal/mol-K"
    ,Cp0_avg);
```

Scilab code Exa 1.14 Calculation of enthalpy of water

```
clear;
clc;

//Example - 1.14
//Page number - 31
printf("Example - 1.14 and Page number - 31\n\n");

//Given
//(a)
P_1 = 0.2;// [MPa] - pressure
x_1 = 0.59;// mole fraction

//From saturated steam tables at 0.2 MPa
H_liq_1 = 504.7;// [kJ/kg] - Enthalpy of saturated
   liquid
H_vap_1 = 2706.7;// [kJ/kg]- Enthalpy of saturated
   vapour
H_1 = (H_liq_1*(1-x_1))+(x_1*H_vap_1);// [kJ/kg]
printf(" (a).Enthalpy of 1 kg of water in tank is %f
    kJ/kg\n",H_1);

//(b)
T_2 = 120.23;// [C] - temperature
V_2 = 0.6;// [m^(3)/kg] - specific volume

//From saturated steam tables at 120.23 C, as
   reported in the book
V_liq_2=0.001061;// [m^(3)/kg]
V_vap_2=0.8857;// [m^(3)/kg]
//since V_2 < Vv_2,dryness factor will be given by,
   V = ((1-x)*V_liq)+(x*V_vap)
x_2 = (V_2- V_liq_2)/(V_vap_2 - V_liq_2);

//From steam table,at 120.2 C,the vapour pressure of
    water is 0.2 MPa.So,enthalpy is given by
H_2 = (H_liq_1*(1-x_2))+(H_vap_1*x_2);//kJ/kg]
printf(" (b).Enthalpy of saturated steam is %f kJ/kg
   \n",H_2);

//(c)
P_3 = 2.5;//[MPa]
T_3 = 350;//[C]
//From steam tables at 2.5 MPa,T_sat = 223.99 C,as
   reported in the book
//since,T_3 > Tsat, steam is superheated
printf(" (c).As steam is superheated,from steam
   table,enthalpy (H) is 3126.3 kJ/kg\n");

//(d)
T_4 = 350;//[C]
V_4 = 0.13857;//[m^(3)/kg]
//From steam table,at 350 C, V_liq = 0.001740 m^(3)/
   kg and V_vap = 0.008813 m^(3)/kg.Since,V > V_vap,
   therefore it is superheated.
//From steam table at 350 C and 1.6 MPa, V = 0.17456
    m^(3)/kg
//At 350 C and 2.0 MPa, V = 0.13857 m^(3)/kg. So,
printf(" (d).The enthalpy of superheated steam (H)
   is 3137.0 kJ/kg\n");

//(e)
P_4 = 2.0;//[MPa]
U_4 = 2900;// [kJ/kg] - internal energy
//From saturated table at 2.0 MPa, U_liq = 906.44kJ
   and U_vap = 2600.3 kJ/kg
//scince,U_4 > Uv, it is saturated.
//From superheated steam table at 2.0 MPa and 350 C,
    as reported in the book
```

```
U_1 = 2859.8;//[kJ/kg]
H_1 = 3137.0;//[kJ/kg]
//At 2.0 MPa and 400 C,
U_2 = 2945.2;//[kJ/kg]
H_2 = 3247.6;//[kJ/kg]
T = (((U_4 - U_1)/(U_2 - U_1))*(400 - 350)) + 350;//
    [C] - By interpolation
H = (((T - 350)/(400 - 350))*(H_2 - H_1)) + H_1;//[
    kJ/kg]
printf(" (e).The enthalpy value (of superheated
    steam) obtained after interpolation is %f kJ/kg\n
    ",H);

//(f)
P_5 = 2.5;//[MPa]
T_5 = 100;//[C]
//At 100 C,P_sat=101350 N/m^(2). Since P_5 > P_sat,
    it is compressed liquid
P_sat = 0.101350;//[MPa]
H_liq = 419.04;//[kJ/kg] - At 100 C and 0.10135 MPa
V_liq = 0.001044;//[m^(3)/kg] - At 100 C and 0.10135
    MPa
H_0 = H_liq + (V_liq*(P_5 - P_sat))*1000;//kJ/kg]
printf(" (f).The enthalpy of compressed liquid is %f
    kJ/kg\n",H_0);
```

Chapter 2

Equations of state

Scilab code Exa 2.1 Relations in virial coefficients

```
clear;
clc;

//Example - 2.1
//Page number - 40
printf("Example - 2.1 and Page number - 40\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 2.1 on page number
    40 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 2.1 on page
    number 40 of the book.");
```

Scilab code Exa 2.2 Determination of acentric factor

```
clear;
clc;

//Example - 2.2
//Page number - 42
printf("Example - 2.2 and Page number - 42\n\n");

//Given
Tc = 647.1;//[K] - Critical temperature
Pc = 220.55;//[bar] - Critical pressure
Tr = 0.7;// Reduced temperature

T = Tr*Tc;//[K]
//From steam table,vapour pressure of H2O at T is
   10.02 [bar], as reported in the book
P = 10.02;//[bar]
w = -1-log10((P/Pc));
printf(" The acentric factor (w) of water at given
   condition is %f ",w);
```

Scilab code Exa 2.3 Calculation of acentric factor

```
clear;
clc;

//Example - 2.3
//Page number - 42
printf("Example - 2.3 and Page number - 42\n\n");

//Given
//log10(Psat)=8.1122-(1592.864/(t+226.184))// 'Psat'
    in [mm Hg] and 't' in [c]
Tc = 513.9;//[K] - Critical temperature
Pc = 61.48;//[bar] - Critical pressure
Pc = Pc*10^(5);//[N/m^(2)]
```

```
Tr = 0.7;// Reduced temperature

T = Tr*Tc;//[K] - Temperature
T = T - 273.15;//[C]
P_sat = 10^(8.1122 - (1592.864/(T + 226.184)));//[mm
    Hg]
P_sat = (P_sat/760)*101325;//[N/m^(2)]
Pr_sat = P_sat/Pc;
w = -1-log10(Pr_sat);// Acentric factor
printf(" The acentric factor (w) for ethanol at
   given condition is %f",w);
```

Scilab code Exa 2.4 Calculation of virial coefficients

```
clear;
clc;

//Example - 2.4
//Page number - 45
printf("Example - 2.4 and Page number - 45\n\n");

//Given
T = 380;//[K] - Temperature
Tc = 562.1;//[K] - Critical temperature
P = 7;//[atm] - Pressure
P = P*101325;//[N/m^(2)]
Pc = 48.3;//[atm] - Critical pressure
Pc = Pc*101325;//[N/m^(2)]
R = 8.314;//[J/mol*K] - Universal gas constant
w = 0.212;// acentric factor
Tr = T/Tc;// Reduced temperature

B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));

```

```
//We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B = ((B_0+(w*B_1))*(R*Tc))/Pc;//[m^(3)/mol]
printf(" The second virial coefficient for benzene
   is %e m^(3)/mol\n",B);

//Compressibility factor is given by
Z = 1 + ((B*P)/(R*T));
printf(" The compressibility factor at 380 K is %f\n
   ",Z);

//We know thar Z=(P*V)/(R/*T),therfore
V = (Z*R*T)/P;//[m^(3)/mol]
printf(" The molar volume is %e m^(3)/mol\n",V);
```

Scilab code Exa 2.5 Calculation of mass using virial equation of state

```
clear;
clc;

//Example - 2.5
//Page number - 46
printf("Example - 2.5 and Page number - 46\n\n");

//Given
V_1 = 0.3;//[m^(3)]//volume of cylinder
T = 60+273.15;//[K] - Temperature
P = 130*10^(5);//[N/m^(2)] - Pressure
Tc = 305.3;//[K] - Critical temperature
Pc = 48.72*10^(5);//[N/m^(2)] - Critical pressure
w = 0.100;//acentric factor
M = 30.07;//molecular weight of ethane
Tr = T/Tc;// Reduced temperature
R = 8.314;//[J/mol*K] - Universal gas constant

B_0 = 0.083-(0.422/(Tr)^(1.6));
```

```
B_1 = 0.139-(0.172/(Tr)^(4.2));

//We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B = ((B_0 + (w*B_1))*(R*Tc))/Pc;//[m^(3)/mol] -
    Second virial coefficient
Z = 1 + ((B*P)/(R*T));//Compressibility factor
V = (Z*R*T)/P;//[m^(3)/mol] - Molar volume

//No.of moles in 0.3 m^(3) cylinder is given by
n1 = V_1/V;//[mol]

//Mass of gas in cylinder is given by
m1 = (n1*M)/1000;//[kg]
printf(" Under actual conditions,the mass of ethane
    is, %f kg\n",m1);

//Under ideal condition, taking Z = 1,
V_ideal = (R*T)/P;//[m^(3)/mol]
n2 = V_1/V_ideal;//[mol]
m2 = (n2*M)/1000;//[kg]
printf(" Under ideal conditions,the mass of ethane
    is, %f kg\n",m2);
```

Scilab code Exa 2.6 Calculation of molar volume

```
clear;
clc;
funcprot(0);

//Example - 2.6
//Page number - 47
printf("Example - 2.6 and Page number - 47\n\n");

//Given
T = 373.15;//[K] - Temperature
```

```
P = 101325;//[N/m^(2)] - Pressure
Tc = 647.1;//[K] - Critical temperature
Pc = 220.55*10^(5);//[N/m^(2)] - Critical pressure
w = 0.345;//acentric factor
Tr = T/Tc;// Reduced temperature
R = 8.314;//[J/mol*K] - UNiversal gas constant

B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));

//We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B = ((B_0+(w*B_1))*(R*Tc))/Pc;//[m^(3)/mol] - Second
    virial coefficient

//We have, Z = 1+(B/V) and Z = (P*V)/(R*T).
   Substituting the value of Z,we get
// V^(2)-((R*T)/P)*V-((B*R*T)/P)=0 .Solving the
   quadratic equation by shreedharcharya rule
V1 = (((R*T)/P) + (((R*T)/P)^(2) + 4*1*((B*R*T)/P))
   ^(1/2))/2*1;

printf(" The molar volume of water vapour is %f m
   ^(3)/mol",V1);

//The roots are,V1 = 0.0003670 [m^(3)/mol] and V2 =
   0.0302510 [m^(3)/mol].
//As 'V2' is near to ideal volume (0.030618 [m^(3)/
   mol]),it is taken as the molar volume
//The other root 'V1' hss no physical significance
```

Scilab code Exa 2.7 Calculation of molar volume and virial coefficients

```
clear;
clc;
funcprot(0);
```

```

//Example - 2.7
// Page number - 47
printf("Example - 2.7 and Page number - 47\n\n");

// Given
T = 50+273.15;//[K] - Temperature
P = 15*10^(5);//[N/m^(2)] - Pressure
Tc = 305.3;//[K] - Critical temperature
Pc = 48.72*10^(5);//[N/m^(2)] - Critical pressure
w = 0.100;// Acentric factor
B = -157.31;//[cm^(3)/mol] - second virial
    coefficient
B = B*10^(-6);//[m^(3)/mol]
C = 9650;//[cm^(6)/mol^(2)] - third virial
    coefficient
C = C*10^(-12);//[cm^(6)/mol^(2)]
R = 8.314;//[J/mol*K] - Universal gas constant

// (1)
V_1 = (R*T)/P;//[m^(3)/mol] - Molar volume
printf(" (1).The molar volume for ideal equation of
    state is %e m^(3)/mol\n",V_1);

// (2)
Tr = T/Tc;// Reduced temperature
// At this temperature
B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));

// We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B_2 = ((B_0 + (w*B_1))*(R*Tc))/Pc;//[m^(3)/mol]//
    second virial coefficient
printf(" (2).The second virial coefficent using
    Pitzer correlation is found to be %e m^(3)/mol
    which is same as given value\n",B_2);

// (3)
// Given (virial equation),Z=1+(B/V)
V_3 = B + (R*T)/P;//[m^(3)/mol] - Molar volume
printf(" (3).The molar volume using virial equation
    of state is %e m^(3)/mol\n",V_3);

// (4)
// Given (virial equation),Z = 1 + ((B*P)/(R*T)) +
    ((C - B^(2))/(R*T)^(2))*P^(2)
V_4 = B + (R*T)/P + ((C - B^(2))/(R*T))*P;// [m^(3)/
    mol]
printf(" (4).The molar volume using given virial
    equation of state is %e m^(3)/mol\n",V_4);

// (5)
// Given,Z = 1 + (B/V)
// Also,Z = (P*V)/(R*T). Substituting the value of Z
    ,we get
// V^(2)-((R*T)/P)*V-((B*R*T)/P)=0.Solving the
    quadratic equation
deff('[y]=f(V)','y=V^(2)-((R*T)/P)*V-((B*R*T)/P)');
V_5_1 = fsolve(0,f);
V_5_2 = fsolve(1,f);

printf(" (5).The molar volume using given virial
    equation of state is %e m^(3)/mol\n",V_5_2);

// The roots are,V_5_1=0.0001743 [m^(3)/mol] and
    V_5_2=0.0016168 [m^(3)/mol].
// As 'V_2' is near to ideal volume (0.0017911 [m
    ^(3)/mol]),it is taken as the molar volume

// (6)
// Given,Z = 1 + (B/V) + (C/V^(2))
// Also,Z = (P*V)/(R*T). Substituting the value of Z
    ,we get
// V^(3)-((R*T)/P)*V^(2)-((B*R*T)/P)*V-((C*R*T)/P)
    =0. Solving the cubic equation
deff('[y]=f1(V)','y=V^(3)-((R*T)/P)*V^(2)-((B*R*T)/P
    )*V-((C*R*T)/P)');
V_6_3=fsolve(-1,f1);
V_6_4=fsolve(0,f1);
V_6_5=fsolve(1,f1);
//The above equation has 1 real and 2 imaginary
    roots. We consider only real root.
printf(" (6).The molar volume using given virial
    equation of state is %e m^(3)/mol\n",V_6_5);
```

Scilab code Exa 2.8 Determination of second and third virial coefficients

```
clear;
clc;
funcprot(0);

//Example - 2.8
// Page number - 49
printf("Example - 2.8 and Page number - 49\n\n");

//Given
T = 0 + 273.15;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant

//Virial equation of state, Z=1+(B/V)+(C/V^(2))
//From above equation we get (Z-1)*V=B+(C/V)

P=[50,100,200,400,600,1000];
Z=[0.9846,1.0000,1.0365,1.2557,1.7559,2.0645];
V=zeros(6);
k=zeros(6);
t=zeros(6);
for i=1:6;
    V(i)=(Z(i)*R*T)/(P(i)*101325);//[m^(3)/mol]
    k(i)=(Z(i)-1)*V(i);
    t(i)=1/V(i);
```

```
end
[C,B,sig]=reglin(t',k');

//From the regression , we get intercept=B and slope=
   C, and thus ,
printf(" The value of second virial coefficient (B)
   is %e m^(3)/mol\n",B);
printf(" The value of third virial coefficient (C)
   is %e m^(6)/mol^(2)",C);
```

Scilab code Exa 2.9 Estimation of second virial coefficient

```
clear;
clc;

//Example - 2.9
//Page number - 51
printf("Example - 2.9 and Page number - 51\n\n");

//Given
T = 444.3;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant
B_11 = -8.1;//[cm^(3)/mol]
B_11 = -8.1*10^(-6);//[m^(3)/mol]
B_22 = -293.4*10^(-6);//[m^(3)/mol]
y1 = 0.5;// mole fraction // equimolar mixture
y2 = 0.5;

// For component 1 (methane)
Tc_1 = 190.6;//[K] - cricitical temperature
Vc_1 = 99.2;//[cm^(3)/mol] - cricitical molar volume
Zc_1 = 0.288;// critical compressibility factor
w_1 = 0.012;// acentric factor

// For component 2 (n-butane)
```

```
Tc_2 = 425.2;//[K]
Vc_2 = 255.0;//[cm^(3)/mol]
Zc_2 = 0.274;
w_2 = 0.199;

//Using virial mixing rule,we get
Tc_12 = (Tc_1*Tc_2)^(1/2);//[K]
w_12 = (w_1 + w_2)/2;
Zc_12 = (Zc_1+Zc_2)/2;
Vc_12 = (((Vc_1)^(1/3) + (Vc_2)^(1/3))/2)^(3);//[cm
    ^(3)/mol]
Vc_12 = Vc_12*10^(-6);//[cm^(3)/mol]
Pc_12 = (Zc_12*R*Tc_12)/Vc_12;//[N/m^(2)]
Tr_12 = T/Tc_12;//Reduced temperature
B_0 = 0.083 - (0.422/(Tr_12)^(1.6));
B_1 = 0.139 - (0.172/(Tr_12)^(4.2));

//We know,(B_12*Pc_12)/(R*Tc_12) = B_0 + (w_12*B_1)
B_12 = ((B_0+(w_12*B_1))*(R*Tc_12))/Pc_12;//[m^(3)/
    mol] - Cross coefficient
B = y1^(2)*B_11+2*y1*y2*B_12+y2^(2)*B_22;//[m^(3)/
    mol] - Second virial coefficient for mixture
B = B*10^(6);//[cm^(3)/mol]
printf(" The second virial coefficient,(B) for the
    mixture of gas is %f cm^(3)/mol",B);
```

Scilab code Exa 2.10 Estimation of molar volume

```
clear;
clc;

//Example - 2.10
//Page number - 52
printf("Example - 2.10 and Page number - 52\n\n");

```

```
//Given
T = 71+273.15;//[K] - Temperature
P = 69*10^(5);//[N/m^(2)] - Pressure
y1 = 0.5;//[mol] - mole fraction of equimolar
   mixture
y2 = 0.5;
R = 8.314;//[J/mol*K] - Universal gas constant

//For component 1 (methane)
Tc_1  =190.6;//[K] - Critical temperature
Pc_1 = 45.99*10^(5);//[N/m^(2)] - Critical pressure
Vc_1 = 98.6;//[cm^(3)/mol] - Critical volume
Zc_1 = 0.286;// Critical compressibility factor
w_1 = 0.012;// acentric factor

//For component 2 (hydrogen sulphide)
Tc_2 = 373.5;//[K]
Pc_2 = 89.63*10^(5);//[N/m^(2)]
Vc_2 = 98.5;//[cm^(3)/mol]
Zc_2 = 0.284;
w_2 = 0.094;

//For component 1
Tr_1 = T/Tc_1;//Reduced temperature
//At reduced temperature
B1_0 = 0.083-(0.422/(Tr_1)^(1.6));
B1_1 = 0.139-(0.172/(Tr_1)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B_11 = ((B1_0+(w_1*B1_1))*(R*Tc_1))/Pc_1;//[m^(3)/
   mol]

//Similarly for component 2
Tr_2 = T/Tc_2;//Reduced temperature
//At reduced temperature Tr_2,
B2_0 = 0.083 - (0.422/(Tr_2)^(1.6));
B2_1 = 0.139 - (0.172/(Tr_2)^(4.2));
B_22 = ((B2_0+(w_2*B2_1))*(R*Tc_2))/Pc_2;//[m^(3)/
   mol]
```

```

//For cross coeffcient
Tc_12 = (Tc_1*Tc_2)^(1/2);//[K]
w_12 = (w_1 + w_2)/2;
Zc_12 = (Zc_1 + Zc_2)/2;
Vc_12 = (((Vc_1)^(1/3) + (Vc_2)^(1/3))/2)^(3);//[cm
    ^(3)/mol]
Vc_12 = Vc_12*10^(-6);//[m^(3)/mol]
Pc_12 = (Zc_12*R*Tc_12)/Vc_12;//[N/m^(2)]

//Now we have,(B_12*Pc_12)/(R*Tc_12) = B_0+(w_12*B_1
    )
//where B_0 and B_1 are to be evaluated at Tr_12
Tr_12 = T/Tc_12;
//At reduced temperature Tr_12
B_0 = 0.083 - (0.422/(Tr_12)^(1.6));
B_1 = 0.139 - (0.172/(Tr_12)^(4.2));
B_12=((B_0 + (w_12*B_1))*R*Tc_12)/Pc_12;//[m^(3)/mol
    ]

//For the mixture
B = y1^(2)*B_11+2*y1*y2*B_12 + y2^(2)*B_22;//[m^(3)/
    mol]

//Now given virial equation is, Z=1+(B*P)/(R*T)
Z = 1 + (B*P)/(R*T);

//Also Z = (P*V)/(R*T).Therefore,
V = (Z*R*T)/P;//[m^(3)/mol]

printf(" The molar volume of the mixture is %e m^(3)
    /mol",V);
//The value obtained is near the experimental value
    of V_exp = 3.38*10^(-4) m^(3)/mol
```

Scilab code Exa 2.11 Calculation of maximum temperature

```
clear;
clc;
funcprot(0);

// Example - 2.11
// Page number - 53
printf("Example - 2.11 and Page number - 53\n\n");

// Given
P = 6*10^(6);// [Pa] - Pressure
P_max = 12*10^(6);// [Pa] - Max pressure to which
    cylinder may be exposed
T = 280;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant

//(1).Assuming ideal gas behaviour,
V_ideal = (R*T)/P;//[m^(3)/mol]
//Now when temperature and pressure are increased,
    the molar volume remains same,as total volume and
     number of moles are same.
//For max pressure of 12 MPa,temperature is
T_max_ideal = (P_max*V_ideal)/R;
printf(" (1).The maximum temperature assuming ideal
    behaviour is %f K\n",T_max_ideal);

// (2).Assuming virial equation of state
// For component 1 (methane),at 280 K
Tc_1 = 190.6;//[K]
Pc_1 = 45.99*10^(5);//[N/m^(2)]
Vc_1 = 98.6;//[cm^(3)/mol]
Zc_1 = 0.286;
w_1 = 0.012;
Tr_1 = T/Tc_1;//Reduced temperature
B1_0 = 0.083 - (0.422/(Tr_1)^(1.6));
B1_1 = 0.139 - (0.172/(Tr_1)^(4.2));

```

```
//We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B_11 = ((B1_0 + (w_1*B1_1))*(R*Tc_1))/Pc_1;//[m^(3)/
    mol]

//For component 2 (Propane)
Tc_2 = 369.8;//[K]
Pc_2 = 42.48*10^(5);//[N/m^(2)]
Vc_2 = 200;//[cm^(3)/mol]
Zc_2 = 0.276;
w_2 = 0.152;
Tr_2 = T/Tc_2;// Reduced temperature
B2_0 = 0.083 - (0.422/(Tr_2)^(1.6));
B2_1 = 0.139 - (0.172/(Tr_2)^(4.2));
B_22 = ((B2_0 + (w_2*B2_1))*(R*Tc_2))/Pc_2;//[m^(3)/
    mol]

//For cross coeffcient
y1 = 0.8;//mole fraction of component 1
y2 = 0.2;//mole fraction of component 2
Tc_12 = (Tc_1*Tc_2)^(1/2);//[K]
w_12 = (w_1 + w_2)/2;
Zc_12 = (Zc_1 + Zc_2)/2;
Vc_12 = (((Vc_1)^(1/3) + (Vc_2)^(1/3))/2)^(3);//[cm
    ^(3)/mol]
Vc_12 = Vc_12*10^(-6);//[m^(3)/mol]
Pc_12 = (Zc_12*R*Tc_12)/Vc_12;//[N/m^(2)]
Tr_12 = T/Tc_12;

//At reduced temperature,Tr_12,
B_0 = 0.083 - (0.422/(Tr_12)^(1.6));
B_1 = 0.139 - (0.172/(Tr_12)^(4.2));
B_12 = ((B_0 + (w_12*B_1))*R*Tc_12)/Pc_12;//[m^(3)/
    mol]

//For the mixture
B = y1^(2)*B_11+2*y1*y2*B_12 + y2^(2)*B_22;//[m^(3)/
    mol]

```

```
//Now given virial equation is , Z=1+(B*P)/(R*T)
Z = 1 + (B*P)/(R*T);
//Also Z = (P*V)/(R*T). Therefore ,
V_real = (Z*R*T)/P;//[m^(3)/mol]

// This molar volume remains the same as the volume
   and number of moles remains fixed.
// Sice Z is a function of presure and temperature ,
   we shall assume a temperature,calculate Z and
   again calculate temperature,till convergence is
   obtained.
// We will use the concept of iteration to compute
   the convergent value of temperature
// Let us start with the temperature at ideal
   conditions i.e T = 560 K,

T_prime = 560;//[K]
fault = 10;

while(fault > 1)
T_prime_r1 = T_prime/Tc_1;
B_prime1_0 = 7.7674*10^(-3);
B_prime1_1 = 0.13714;
B_prime_11 = ((B_prime1_0 + (w_1*B_prime1_1))*(R*
   Tc_1))/Pc_1;//[m^(3)/mol]

//Similarly for component 2,
T_prime_r2 = T_prime/Tc_2;
B_prime2_0 = -0.1343;
B_prime2_1 = 0.10887;
B_prime_22 = ((B_prime2_0 + (w_2*B_prime2_1))*(R*
   Tc_2))/Pc_2;//[m^(3)/mol]

//For cross coefficient (assuming k12=0)
//Tc_12 , w_12 , Zc_12 , Vc_12 and Pc_12 have
   already been calculated above,now
T_prime_r12 = T_prime/Tc_12;//
//At reduced temperature,T_prime_r12 ,
B_prime_0 = 0.083 - (0.422/(T_prime_r12)^(1.6));
B_prime_1 = 0.139 - (0.172/(T_prime_r12)^(4.2));
B_prime_12 = ((B_prime_0+(w_12*B_prime_1))*R*Tc_12)/
   Pc_12;//[m^(3)/mol]

//For the mixture
B_prime = y1^(2)*B_prime_11 + 2*y1*y2*B_prime_12 +
   y2^(2)*B_prime_22;//[m^(3)/mol]
Z_prime = 1 + (B_prime*P_max)/(R*T_prime);
T_new = (P_max*V_real)/(Z_prime*R);
fault = abs(T_prime - T_new);
T_prime = T_new;
end

printf(" (2).The maximum temperature assuming the
   gas to follow virial equation of stste is %f K\n"
   ,T_new);
```

Scilab code Exa 2.12 Calculation of pressure

```
clear;
clc;

// Example - 2.12
// Page number - 64
printf("Example - 2.12 and Page number - 64\n\n");

//Given

V_vessel = 0.1;//[m^(3)]// Volume of vessel
T = 25 + 273.15;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant
m = 25*1000;//[g]// Mass of ethylene
Tc = 282.3;//[K] - Critical temperature
Pc = 50.40;//[bar] - Critical pressure
Pc = Pc*10^(5);//[N/m^(2)]
Zc = 0.281;// Critical compressibility factor
Vc = 131;//[cm^(3)/mol] - Critical volume
Vc = Vc*10^(-6);//[m^(3)/mol]
w = 0.087;// Acentric factor
M = 28.054;// Molecular weight of ethylene

n = m/M;//[mole] - No. of moles of ethylene
V = V_vessel/n;//[m^(3)/mol] - Molar volume

//Under Redlich Kwong equation of state, we have
a = (0.42748*(R^(2))*(Tc^(2.5)))/Pc;//[Pa*m^(6)*K
   ^(1/2)/mol]
b = (0.08664*R*Tc)/Pc;//[m^(3)/mol]
P = ((R*T)/(V-b))-(a/(T^(1/2)*V*(V+b)));//[N/m^(2)]
printf(" The required pressure using Redlich Kwong
   equation of state is %e N/m^(2)\n",P);

//For ideal gas equation of state,
P_ideal = (R*T)/V;//[N/m^(2)]
printf(" For ideal gas equation of state,the
   required pressure is %e N/m^(2)\n",P_ideal);
```

Scilab code Exa 2.13 Calculation of pressure

```
clear;
clc;

// Example - 2.13
// Page number - 65
printf("Example - 2.13 and Page number - 65\n\n");

//Given

V_vessel = 360*10^(-3);//[m^(3)] - volume of vessel
```

```
T = 62+273.15;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant
m = 70*1000;//[g]/ - Mass of carbon dioxide

//For carbon dioxide
Tc = 304.2;//[K] - Cricitical temperature
Pc = 73.83;//[bar] - Cricitical pressure
Pc = Pc*10^(5);// [N/m^(2)]
Zc = 0.274;// Critical compressibility factor
Vc = 94.0;//[cm^(3)/mol]
Vc = Vc*10^(-6);//[m^(3)/mol]
w = 0.224;// Acentric factor
M = 44.01;// Molecular weight of carbon dioxide

n = m/M;//[mol] - No. of moles
V = V_vessel/n;//[m^(3)/mol]//molar volume

// (1)
// Ideal gas behaviour
P_1 = (R*T)/V;//[N/m^(2)]
printf(" (1).The required pressure using ideal
    equation of state is %e N/m^(2)\n",P_1);

// (2)
// Virial equation of state, Z = 1 + (B*P)/(R*T)
// (P*V)/(R*T) = 1 + (B*P)/(R*T), and thus P = (R*T)
    /(V - B). Now
Tr = T/Tc;//Reduced temperature
// At reduced temperature Tr,
B_0 = 0.083 - (0.422/(Tr)^(1.6));
B_1 = 0.139 - (0.172/(Tr)^(4.2));
B = ((B_0 + (w*B_1))*(R*Tc))/Pc;//[m^(3)/mol]
P_2 = (R*T)/(V - B);//[N/m^(2)]
printf(" (2).The required pressure using given
    virial equation of state is %e N/m^(2)\n",P_2);

// (3)
// Virial equation of state, Z = 1 + (B/V)
// (P*V)/(R*T) = 1 + (B/V)
P_3 = ((R*T)/V) + (B*R*T)/(V^(2));//[N/m^(2)]
printf(" (3).The required pressure using given
    virial equation of state is %e N/m^(2)\n",P_3);

// (4)
// Van der Walls equation of state,P = ((R*T)/(V-b))
    - a/(V^(2))
a = (27*(R^(2))*(Tc^(2)))/(64*Pc);//[Pa*m^(6)/mol
    ^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]
P_4 = ((R*T)/(V-b)) - a/(V^(2));//[N/m^(2)]
printf(" (4).The required pressure using van der
    Walls equation of state is %e N/m^(2)\n",P_4);

//(5)
// Redlich Kwong equation of state,
a_1 = (0.42748*(R^(2))*(Tc^(2.5)))/Pc;//[Pa*m^(6)*K
    ^(1/2)/mol]
b_1 = (0.08664*R*Tc)/Pc;//[m^(3)/mol]
P_5 = ((R*T)/(V - b_1)) - (a_1/(T^(1/2)*V*(V + b_1))
    );//[N/m^(2)]
printf(" (5).The required pressure using Redlich
    Kwong equation of state is %e N/m^(2)\n",P_5);
```

Scilab code Exa 2.14 Determination of compressibility factor

```
clear;
clc;
funcprot(0);

// Example - 2.14
// Page number - 66
printf("Example - 2.14 and Page number - 66\n\n");

```

```
//Given
T = 500+273.15;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant
P = 325*1000;//[Pa] - Pressure
Tc = 647.1;//[K] - Cricitical temperature
Pc = 220.55;//[bar] - Cricitical pressure
Pc = Pc*10^(5);//[N/m^(2)]

//(1)
// Van der Walls equation of state,
a = (27*(R^(2))*(Tc^(2)))/(64*Pc);//[Pa*m^(6)/mol
   ^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]
// The cubic form of van der Walls equation of state
    is given by,
// V^(3)-(b+(R*T)/P)*V^(2)+(a/P)*V-(a*b)/P=0
// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)-(b+(R*T)/P)*V^(2)+(a/P)*V-(
   a*b)/P');
V_1 = fsolve(1,f);
V_2 = fsolve(10,f);
V_3 = fsolve(100,f);
// The above equation has 1 real and 2 imaginary
   roots. We consider only real root,
Z_1 = (P*V_1)/(R*T);//compressibility factor
printf(" (1).The compressibility factor of steam
   using van der Walls equation of state is %f\n",
   Z_1);

//(2)

//Redlich Kwong equation of state,
a_1 = (0.42748*(R^(2))*(Tc^(2.5)))/Pc;//[Pa*m^(6)*K
   ^(1/2)/mol]
b_1 = (0.08664*R*Tc)/Pc;//[m^(3)/mol]
// The cubic form of Redlich Kwong equation of state
    is given by,
// V^(3)-((R*T)/P)*V^(2)-((b_1^(2))+((b_1*R*T)/P)-(a
    /(T^(1/2)*P))*V-(a*b)/(T^(1/2)*P)=0
//Solving the cubic equation
deff('[y]=f1(V)','y=V^(3)-((R*T)/P)*V^(2)-((b_1^(2))
    +((b_1*R*T)/P)-(a_1/(T^(1/2)*P)))*V-(a_1*b_1)/(T
    ^(1/2)*P)');
V_4=fsolve(1,f1);
V_5=fsolve(10,f1);
V_6=fsolve(100,f1);
// The above equation has 1 real and 2 imaginary
    roots. We consider only real root,
// Thus compressibility factor is
Z_2 = (P*V_4)/(R*T);//compressibility factor
printf(" (2).The compressibility factor of steam
    using Redlich Kwong equation of state is %f\n",
    Z_2);
```

Scilab code Exa 2.15 Determination of molar volume

```
clear;
clc;

// Example - 2.15
// Page number - 67
printf("Example - 2.15 and Page number - 67\n\n");

//Given
T = 250+273.15;//[K]
R = 8.314;//[J/mol*K]
P = 39.76;//[bar] Vapour pressure of water at T
P = P*10^(5);//[N/m^(2)]
Tc = 647.1;//[K] - Cricitical temperature
Pc = 220.55*10^(5);//[N/m^(2)] - Cricitical pressure
w = 0.345;//Acentric factor
M = 18.015;// Molecular weight of water

```

```
// Using peng-Robinson equation of stste
m = 0.37464 + 1.54226*w - 0.26992*w^(2);
Tr = T/Tc;
alpha = (1 + m*(1 - Tr^(1/2)))^(2);
a = ((0.45724*(R*Tc)^(2))/Pc)*alpha;//[Pa*m^(6)/mol^(2)]
b = (0.07780*R*Tc)/Pc;//[m^(3)/mol]
// Cubuc form of Peng-Robinson equation of stste is given by
// V^(3) + (b-(R*T)/P)*V^(2) - ((3*b^(2)) + ((2*R*T*b)/P) - (a/P))*V+b^(3) + ((R*T*(b^(2))/P) - ((a*b)/P) = 0;
// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)+(b-(R*T)/P)*V^(2)-((3*b^(2))+((2*R*T*b)/P)-(a/P))*V+b^(3)+((R*T*(b^(2)))/P)-((a*b)/P)');
V_1 = fsolve(-1,f);
V_2 = fsolve(0,f);
V_3 = fsolve(1,f);
//The largest root is for vapour phase,
V_vap = V_3;//[m^(3)/mol] - Molar volume (saturated vapour)
V_vap = V_vap*10^(6)/M;//[cm^(3)/g]

printf(" The moar volume of saturated water in the vapour phase (V_vap) is %f cm^(3)/g\n",V_vap);

//The smallest root is for liquid phase,
V_liq = V_1;//[m^(3)/mol] - molar volume (saturated liquid)
V_liq = V_liq*10^(6)/M;//[cm^(3)/g]
printf(" The moar volume of saturated water in the liquid phase (V_liq) is %f cm^(3)/g\n",V_liq);

//From steam table at 250 C, V_vap = 50.13 [cm^(3)/g] and V_liq = 1.251 [cm^(3)/g].
printf(" From steam table at 250 C, V_vap = 50.13 [cm^(3)/g] and V_liq = 1.251 [cm^(3)/g]");
```

Scilab code Exa 2.16 Calculation of volume

```
clear;
clc;
funcprot(0);

// Example - 2.16
// Page number - 68
printf("Example - 2.16 and Page number - 68\n\n");

//Given
T = 500+273.15;//[K] - Temperature
P = 15;//[atm] - Pressure
P = P*101325;//[N/m^(2)]
R = 8.314;//[J/mol*K] - Universal gas constant
Tc = 190.6;//[K] - Cricitical temperature
Pc = 45.99*10^(5);//[N/m^(2)] - Cricitical pressure
Vc = 98.6;//[cm^(3)/mol] - Cricitical molar volume
Zc = 0.286;// Critical compressibility factor
w = 0.012;// Acentric factor

//(1)
//Virial equation of state,Z = 1 + (B*P)/(R*T)
Tr_1 = T/Tc;//Reduced temperature
B_0 = 0.083-(0.422/(Tr_1)^(1.6));
B_1 = 0.139-(0.172/(Tr_1)^(4.2));
// We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B = ((B_0+(w*B_1))*(R*Tc))/Pc;//[m^(3)/mol]//second
    virial coefficient
Z = 1 + (B*P)/(R*T);//compressibility factor
//(P*V)/(R*T)=1+(B*P)/(R*T),and thus,
V_1  = (Z*R*T)/P;//[m^(3)/mol]
printf(" (1).The molar volume of methane using given
    virial equation is %e m^(3)/mol\n",V_1);
```

```

//(2).
//Virial equation of state,Z = 1 + (B/V)
//Also,Z = (P*V)/(R*T). Substituting the value of Z,
   we get
// V^(2) - ((R*T)/P)*V - ((B*R*T)/P) = 0.Solving the
    quadratic equation
deff('[y]=f(V)','y=V^(2)-((R*T)/P)*V-((B*R*T)/P)');
V2_1=fsolve(0,f);
V2_2=fsolve(1,f);
// Out of two roots,we will consider only positive
   root
printf(" (2).The molar volume of methane using given
    virial equation is %e m^(3)/mol\n",V2_2);

// (3)
// Van der Walls equation of state,
// (P + (a/V^(2)))*(V - b) = R*T
a_3 = (27*(R^(2))*(Tc^(2)))/(64*Pc);//[Pa*m^(6)/mol
   ^(2)]
b_3 = (R*Tc)/(8*Pc);//[m^(3)/mol]
// The cubic form of van der Walls equation of state
    is given by,
// V^(3) - (b + (R*T)/P)*V^(2) + (a/P)*V - (a*b)/P =
    0
// Solving the cubic equation
deff('[y]=f1(V)','y=V^(3)-(b_3+(R*T)/P)*V^(2)+(a_3/P
   )*V-(a_3*b_3)/P');
V3_1=fsolve(1,f1);
V3_2=fsolve(10,f1);
V3_3=fsolve(100,f1);
// The above equation has 1 real and 2 imaginary
   roots. We consider only real root.
printf(" (3).The molar volume of methane using van
   der Walls equation of state is %e m^(3)/mol\n",
   V3_1);

// (4)
// Redlich Kwong equation of state
a_4 = (0.42748*(R^(2))*(Tc^(2.5)))/Pc;//[Pa*m^(6)*K
    ^(1/2)/mol]
b_4 = (0.08664*R*Tc)/Pc;//[m^(3)/mol]
// The cubic form of Redlich Kwong equation of state
     is given by,
// V^(3) - ((R*T)/P)*V^(2) - ((b_1^(2)) + ((b_1*R*T)
    /P) - (a/(T^(1/2)*P))*V - (a*b)/(T^(1/2)*P) = 0
// Solving the cubic equation
deff('[y]=f2(V)','y=V^(3)-((R*T)/P)*V^(2)-((b_4^(2))
    +((b_4*R*T)/P)-(a_4/(T^(1/2)*P)))*V-(a_4*b_4)/(T
    ^(1/2)*P)');
V4_1=fsolve(1,f2);
V4_2=fsolve(10,f2);
V4_3=fsolve(100,f2);
//The above equation has 1 real and 2 imaginary
    roots. We consider only real root.
printf(" (4).The molar volume of methane using
    Redlich Kwong equation of state is %e m^(3)/mol\n
    ",V4_1);

// (5)
// Using Peng-Robinson equation of state
m = 0.37464 + 1.54226*w - 0.26992*w^(2);
Tr_5 = T/Tc;
alpha = (1 + m*(1 - Tr_5^(1/2)))^(2);
a = ((0.45724*(R*Tc)^(2))/Pc)*alpha;//[Pa*m^(6)/mol
    ^(2)]
b = (0.07780*R*Tc)/Pc;//[m^(3)/mol]
// Cubic form of Peng-Robinson equation of stste is
    given by
// V^(3)+(b-(R*T)/P)*V^(2)-((3*b^(2))+((2*R*T*b)/P)
    -(a/P))*V+b^(3)+((R*T*(b^(2))/P)-((a*b)/P)=0;
// Solving the cubic equation
deff('[y]=f3(V)','y=V^(3)+(b-(R*T)/P)*V^(2)-((3*b
    ^(2))+((2*R*T*b)/P)-(a/P))*V+b^(3)+((R*T*(b^(2))
    /P)-((a*b)/P)');
V5_1=fsolve(-1,f3);
```

```
V5_2=fsolve(0,f3);
V5_3=fsolve(1,f3);
//The largest root is for vapour phase,
//The largest root is only considered as the
   systemis gas
printf(" (5).The molar volume of methane using Peng−
   Robinson equation of state is %e m^(3)/mol\n",
   V5_3);
```

Scilab code Exa 2.17 Estimation of compressibility factor

```
clear;
clc;
funcprot(0);

// Example − 2.17
// Page number − 70
printf("Example − 2.17 and Page number − 70\n\n");

//Given
T = 310.93;//[K] − Temperature
P = 2.76*10^(6);//[N/m^(2)] − Pressure
R = 8.314;//[J/mol*K] − Universal gas constant
y1 = 0.8942;// Mole fraction of component 1 (methane
   )
y2 = 1-y1;// Mole fraction of component 2 (n−butane)

//For component 1 (methane)
Tc_1 = 190.58;//[K] − Cricitical temperature
Pc_1 = 46.05;//[bar] − Cricitical pressure
Pc_1 = Pc_1*10^(5);//[N/m^(2)]
Zc_1 = 0.288;// Critical compressibility factor
Vc_1 = 99.1;//[cm^(3)/mol]
Vc_1 = Vc_1*10^(-6);//[m^(3)/mol]
w_1 = 0.011;// Acentric factor
```

```

//For component 2 (n-butane)
Tc_2 = 425.18;//[K] - Cricitical temperature
Pc_2 = 37.97;//[bar] - Cricitical pressure
Pc_2 = Pc_2*10^(5);// [N/m^(2)]
Zc_2 = 0.274;// Critical compressibility factor
Vc_2 = 255.1;// [cm^(3)/mol]
Vc_2 = Vc_2*10^(-6);// [m^(3)/mol]
w_2 = 0.193;// Acentric factor

// (1)
// Virial equation of state , Z = 1 + (B*P)/(R*T)
// For component 1 (methane)
Tr_1 = T/Tc_1;//Reduced temperature
// At reduced temperature
B1_0 = 0.083 - (0.422/(Tr_1)^(1.6));
B1_1 = 0.139 - (0.172/(Tr_1)^(4.2));
// We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B_11 = ((B1_0+(w_1*B1_1))*(R*Tc_1))/Pc_1;//[m^(3)/
    mol]

//Similarly for component 2
Tr_2 = T/Tc_2;//Reduced temperature
//At reduced temperature Tr_2,
B2_0 = 0.083 - (0.422/(Tr_2)^(1.6));
B2_1 = 0.139 - (0.172/(Tr_2)^(4.2));
B_22 = ((B2_0 + (w_2*B2_1))*(R*Tc_2))/Pc_2;//[m^(3)/
    mol]

//For cross coeffcient
Tc_12 = (Tc_1*Tc_2)^(1/2);//[K]
w_12 = (w_1 + w_2)/2;
Zc_12 = (Zc_1 + Zc_2)/2;
Vc_12 = (((Vc_1)^(1/3)+(Vc_2)^(1/3))/2)^(3);//[m^(3)
    /mol]
Pc_12 =(Zc_12*R*Tc_12)/Vc_12;//[N/m^(2)]

//Now we have,(B_12*Pc_12)/(R*Tc_12) = B_0+(w_12*B_1
    )
//where B_0 and B_1 are to be evaluated at Tr_12
Tr_12 = T/Tc_12;
//At reduced temperature Tr_12
B_0 = 0.083 - (0.422/(Tr_12)^(1.6));
B_1 = 0.139 - (0.172/(Tr_12)^(4.2));
B_12 = ((B_0+(w_12*B_1))*R*Tc_12)/Pc_12;//[m^(3)/mol
    ]

//For the mixture
B = y1^(2)*B_11+2*y1*y2*B_12+y2^(2)*B_22;//[m^(3)/
    mol]
Z_1 = 1+(B*P)/(R*T);//compressibility factor
printf(" (1).The compressibility factor of mixture
    using Virial equation of state is %f\n",Z_1);

// (2)
// Pseudo reduced method.
T_pc = (y1*Tc_1)+(y2*Tc_2);//[K] - Cricitical
    temperature
P_pc = (y1*Pc_1)+(y2*Pc_2);//[N/m^(2)] - Cricitical
    pressure
w = (y1*w_1)+(y2*w_2);// Acentric factor
T_pr = T/T_pc;// Reduced temperature
P_pr = P/P_pc;// Reduced pressure
//At this value of Tpr,
B0 = 0.083 - (0.422/(T_pr)^(1.6));
B1 = 0.139 - (0.172/(T_pr)^(4.2));
Z0 = 1 + B0*(P_pr/T_pr);
Z1 = B1*(P_pr/T_pr);
Z = Z0 + w*Z1;
printf(" (2).The compressibility factor of mixture
    using pseudo reduced method is %f\n",Z);

// (3)
// Redlich Kwong equation of state is given by
// P = ((R*T)/(V-b)) - (a/(T^(1/2)*V*(V+b)))
// For methane,component 1
```

```
a_1 = (0.42748*(R^(2))*(Tc_1^(2.5)))/Pc_1;//[Pa*m
    ^(6)*K^(1/2)/mol]
b_1 = (0.08664*R*Tc_1)/Pc_1;//[m^(3)/mol]
//For n-butane,component 2
a_2 = (0.42748*(R^(2))*(Tc_2^(2.5)))/Pc_2;//[Pa*m
    ^(6)*K^(1/2)/mol]
b_2 = (0.08664*R*Tc_2)/Pc_2;//[m^(3)/mol]
//For the mixture
a_12 = (a_1*a_2)^(1/2);//[Pa*m^(6)*K^(1/2)/mol]
a = y1^(2)*a_1 + 2*y1*y2*a_12 + y2^(2)*a_2;//[Pa*m
    ^(6)*K^(1/2)/mol]
b = (y1*b_1) + (y2*b_2);//[m^(3)/mol]
// The cubic form of Redlich Kwong equation of state
     is given by,
// V^(3) - ((R*T)/P)*V^(2) - ((b_1^(2)) + ((b_1*R*T)
    /P) - (a/(T^(1/2)*P))*V - (a*b)/(T^(1/2)*P) = 0
// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)-((R*T)/P)*V^(2)-((b^(2))+((
    b*R*T)/P)-(a/(T^(1/2)*P)))*V-(a*b)/(T^(1/2)*P)');
V_1=fsolve(1,f);
V_2=fsolve(10,f);
V_3=fsolve(100,f);
// Thus compressibility factor is
Z_3 = (P*V_1)/(R*T);//compressibility factor
printf(" (3).The compressibility factor of mixture
    using Redlich Kwong equation of state is %f\n",
    Z_3);
```

Chapter 3

The First Law and Its Applications

Scilab code Exa 3.1 Calculation of temperature

```
clear;
clc;
funcprot(0);

//Example - 3.1
//Page number - 80
printf("Example - 3.1 and Page number - 80\n\n");

// Given
V_vessel = 4*10^(-3);//[m^(-3)] - Volume of vessel
T = 200+273.15;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal fas constant
P = 1.5*10^(6);//[Pa] - Pressure
Q = 40*1000;//[J] - Heat input
// From steam table at 200 C,Psat=1.55549 MPa,
   therefore the steam is superheated.

// (1)
// Using steam table,at 1.5 MPa and 200 C,
V_1 = 0.1325;//[m^(3)/mol] - Specific volume
U_1 = 2598.1;//[kJ/kg] - Specific internal energy
// From first law under constant pressure,
// Q - m*P*(V2 - V1) = m*(U2 - U1)
m = V_vessel/V_1;//[kg] - Mass of system
// Putting the values,the above equation becomes
// 45283*(V2 - 0.1325) + 30.1887*(U2 - 2598.1) =
   40000
// From steam table at 700 C LHS is 33917.0 and at
   800 C,it is 40925.3.
// Therefore the final temperature lies between 700
   and 800 C
printf(" (1).From steam table the final temperature
   lies between 700 and 800 C\n");

// Alternate method
// Here we use first law at constant pressure,
// Q = m*(H_2 - H_1)
H_1 = 2796.8;//[kJ/kg]
// Substituting the values,
// 40 = 0.0301887*(H_2 - 2796.8)
H_2 = (40/0.0301887) + 2796.9;//[kJ/kg]
// Threfore,final enthalpy is (H2) 4121.8 [kJ/kg]
   and pressure is 1.5 [MPa].
// From steam table at 1.5 [MPa]and 700 C,enthalpy
   is 3920.3 [kj/kg] and at 1.5 [MPa]and 800 C,
   enthalpy is 4152.6 [kj/kg]
printf("\tAlternate method\n");
printf("\tBy linear interpolation we get the
   temperature at which enthlpy is 4121.8 kJ/kg to
   be 786.74 C\n\n");

// (2)
// Assuming ideal behaviour.
n = (P*V_vessel)/(R*T);//[mol] - No of moles
M = 18.015;// Molecular weight of water
m_2 = n*M;//[g] - mass of moles
Cp_1 = 7.7 + 0.04594*10^(-2)*T + 0.2521*10^(-5)*T
```

```
    ^(2) - 0.8587*10^(-9)*T^(3);//[cal/mol*K] - Heat
    capacity at constant presure
R0 = 1.987;//[cal/mol*K] - universal gas constant
Cv_1 = Cp_1 - R0;//[cal/mol*K] - Heat capacity at
    constant volume
Cv_1 = Cv_1*4.184/M;//[J/g*K]
T1 = T;
// From 1st law energy balance for constant pressure
    , we have Q-W= m*(delta_U)
// Q = P*(V2 - V1)*m = m*Cv*(T2 - T1)
// Q = P*((T2/T1)-1)*V1*m = m*Cv*(T2-T1)
// But, (V1*Cv)=initial total volume of system =
    V_vessel
// Q-P((T2/T1)-1)*V_vessel = m_2*Cv_0*(T2-T1);
deff('[y]=f(T2)','y=Q-P*((T2/T1)-1)*V_vessel-m_2*
    Cv_1*(T2-T1)');
T2_1 = fsolve(1,f);
//The heat capacity should be evaluted at mean
    temperature
T_mean = (T1 + T2_1)/2;
Cp_2 = 7.7 + 0.04594*10^(-2)*T_mean+0.2521*10^(-5)*
    T_mean^(2) - 0.8587*10^(-9)*T_mean^(3);//[cal/mol
    *K] - Heat capacity at constant presure
Cv_2 = Cp_2-R0;//[cal/mol*K] - Heat capacity at
    constant volume
Cv_2 = Cv_2*4.184/M;//[J/g*K]
//Now again solving the equation Q=P*((T2/T1)-1)*V1*
   m = m*Cv*(T2-T1),for Cv=Cv_2
deff('[y]=f1(T2)','y=Q-P*((T2/T1)-1)*V_vessel-m_2*
    Cv_2*(T2-T1)');
T2_2 = fsolve(1,f1);
printf(" (2).The temperature assuming ideal
    behaviour is %f K\n",T2_2);

// Alternate method
// From 1st law at constant pressure, we have Q = m*
   Cp(T2-T1)
T2_3 = Q/(m_2*(Cp_1*4.184/M))+T1;
```

```
//We can calculate the mean temperature as done
   above
T_mean1 = (T1 + T2_3)/2;//[J/g*K]
//The heat capacity should be evaluted at mean
   temperature
Cp_3 = 7.7 + 0.04594*10^(-2)*T_mean1 +
   0.2521*10^(-5)*T_mean1^(2)-0.8587*10^(-9)*T_mean1
   ^(3);//[cal/mol*K] - Heat capacity at constant
   presure
Cp_3 = Cp_3*4.184/M;//[J/g*K]
// Again solving the equation Q = m*Cp(T2-T1), for
   Cp=Cp_3
T2_4 = Q/(m_2*Cp_3) + T1;
printf("\tAlternate method\n");
printf("\tThe temperature assuming ideal behaviour (
   alternate method) is %f K\n",T2_4);
```

Scilab code Exa 3.2 Calculation of heat required

```
clear;
clc;

//Example - 3.2
//Page number - 83
printf("Example - 3.2 and Page number - 83\n\n");

//Given
V_tank = 1;//[m^(3)] - Volume of the tank
V_liq = 0.05;//[m^(3)] - Volume of saturated water
V_vap = 0.95;//[m^(3)] - Volume of saturated vapour
P = 1;//[bar] - Pressure
V_liq_sat = 0.001043;//[m^(3)/kg] - Specific volume
   of saturated water
V_vap_sat = 1.6940;//[m^(3)/kg] - Specific volume of
    saturated vapour
```

```
U_liq_sat = 417.4;//[kJ/kg] - Saturated liquid
    internal energy
U_vap_sat = 2506.1;//[kJ/kg] - Saturated vapour
    internal energy
m = (V_liq/V_liq_sat) + (V_vap/V_vap_sat);//[kg] -
    Total mass of water
U_1 = (V_liq/V_liq_sat)*U_liq_sat + (V_vap/V_vap_sat
    )*U_vap_sat;//[kJ] - Total internal energy

// At final state,which is saturated vapour
V = V_tank/m;//[m^(3)/kg] - Molar volume
// From saturated steam table at 8 MPa,as reported
    in the book V_vap = 0.02352[m^(3)/kg] and U_vap =
     2569.8[kJ/kg]
// At 9 MPa, Vv = 0.02048[m^(3)/kg] and Uv = 2557.8[
    kJ/kg]
// Therefore final state pressure of the system (
    from interpolation) is 8.954 [MPa] and internal
    energy of saturated vapour is 2558.35 [kJ/kg]
U_2 = m*2558.35;//[kJ] - Final total internal energy
del_Ut = U_2 - U_1;//[kJ]
//we have, del_U = Q - W
//Here work done is zero because volume is rigid.
Q = del_Ut;//[kJ]
Q = del_Ut*10^(-3);//[MJ]
printf(" The amount of heat to be added is %f MJ", Q
    );
```

Scilab code Exa 3.3 Calculation of temperature internal energy and enthalpy

```
clear;
clc;

//Example - 3.3
```

```
//Page number - 83
printf("Example - 3.3 and Page number - 83\n\n");

//Given
M_vap_sat = 0.22;//[kg] - mass of saturated vapour
M_liq_sat = 1.78;//[kg] - mass of saturated liquid
P = 700;//[kPa] - Pressure

//At P=700 kPa,the systen is saturated,from steam
   table as reported in the book
T_sat1 = 164.97;//[C]
V_liq_1 = 0.001108;//[m^(3)/kg]
V_vap_1 = 0.2729;//[m^(3)/kg]
Vt_1 = V_liq_1*M_liq_sat + V_vap_1*M_vap_sat;//[m
   ^(3)] - total volume

//At final state,P = 8 MPa
T_sat2 = 295.06;//[C]
V_liq_2 = 0.001384;//[m^(3)/kg]
V_vap_2=0.02352;//[m^(3)/kg]
Vt_2 = Vt_1;// Since the volume is rigid.
// Since the volume of 2 kg of vapour is 0.062 [m
   ^(3)]
V = Vt_2/2;//[m^(3)/kg] - specific volume

// (a)
// From steam table at 8 [MPa]and 350 [C],V=0.02995[
   m^(3)/kg];
V_1 = 0.02995;//[m^(3)/kg]
// And at 8 [MPa]and 400 [C],
V_2 = 0.03432;//[m^(3)/kg]
// By interpolation,
T = ((V-V_1)/(V_2 - V_1))*(400-350)+350;
printf(" (a).The final temperature is %f c\n",T);

// (b)
// From steam table
U_1 = 2747.7;//[kJ/kg]
```

```
H_1 = 2987.3;//[kJ/kg]
// And at 8 [MPa]and 400 C,
U_2 = 2863.8;//[kJ/kg]
H_2 = 3138.3;//[kJ/kg]
// Therefore at T = 362.01 C
U = U_1+((U_2 - U_1)/(400 - 350))*(T - 350);
printf(" (b).The internal energy is %f kJ/kg\n",U);

//(c)
H = H_1+((H_2 - H_1)/(400 - 350))*(T - 350);
printf(" (b).The enthalpy is %f kJ/kg\n",H);
```

Scilab code Exa 3.4 Calculation of work done

```
clear;
clc;

//Example - 3.4
//Page number - 85
printf("Example - 3.4 and Page number - 85\n\n");

// Given
T = 300;//[K] - Temperature
P1 = 1;//[bar] - Initial pressure
P1 = P1*10^(5);//[N/m^(2)]
P2 = 8;//[bar] - Final pressure
P2 = P2*10^(5);//[N/m^(2)]
R = 8.314;//[J/mol*K] - Universal gas constant
Tc = 126.2;//[K] - Critical temperature
Pc = 34;//[bar] - Critical pressure
Pc = Pc*10^(5);//[N/m^(2)]
w = 0.038;//  Acentric factor

// w = integral(Pdv)
// Z = 1 + (B/V)
// (P*V)/(R*T) = 1 + (B/V)
// P = (R*T)/V + (B*R*T)/V^(2)
// w = integrate('(R*T/V) + (B*R*T)/V^(2)','V',V1,V2
    )
// Under isothermal conditions,
// w = R*T*log(V2/V1) - B*R*T*((1/V2) - (1/V1));
// The second virial coefficient at state 1 is same
    as at state 2,as the temperature is the same i.e,
     T=300 [K]
Tr = T/Tc;
B_0 = 0.083 - (0.422/(Tr)^(1.6));
B_1 = 0.139 - (0.172/(Tr)^(4.2));
B = ((B_0+(w*B_1))*(R*Tc))/Pc;//[m^(3)/mol]

// Now we have to calculate molar volume i.e V1 and
    V2 at given conditions
// At state 1,
deff('[y]=f(V)','y=V^(2)-(R*T/P1)*V-(B*R*T)/P1');
V_1 = fsolve(-1,f);
V_2 = fsolve(1,f);
// We will take root near to (R*T)/P, i.e V_2
V1 = V_2;

// At state 2,
deff('[y]=f1(V)','y=V^(2)-(R*T/P2)*V-(B*R*T)/P2');
V_3=fsolve(-1,f1);
V_4=fsolve(1,f1);
V2 = V_4;
// The work done is thus,
w = R*T*log(V2/V1) - B*R*T*((1/V2) - (1/V1));//[J]
w = w*10^(-3);//[kJ]

printf(" The work done is %f kJ/mol\n",w);
printf(" Negative sign indicates that work is done
    on the gas");
```

Scilab code Exa 3.5 Calculation of work done

```
clear;
clc;

//Example - 3.5
//Page number - 86
printf("Example - 3.5 and Page number - 86\n\n");

//Given

T = 300;//[K] - Temperature
P1 = 1;//[bar] - Initial pressure
P1 = P1*10^(5);//[N/m^(2)]
P2 = 8;//[bar] - Final pressure
P2 = P2*10^(5);//[N/m^(2)]
R = 8.314;//[J/mol*K] - Universal gas constant
y1 = 0.21;// Mole fraction of component 1 (oxygen)
y2 = 0.79;// Mole fraction of component 1 (nitroen)

// For component 1 (Oxygen)
Tc_1 = 154.6;//[K]
Pc_1 = 50.43*10^(5);//[N/m^(2)]
Vc_1 = 73.4;//[cm^(3)/mol]
Zc_1 = 0.288;
w_1 = 0.022;

//For component 2 (Nitrogen)
Tc_2 = 126.2;//[K]
Pc_2 = 34*10^(5);//[N/m^(2)]
Vc_2 = 89.2;//[cm^(3)/mol]
Zc_2 = 0.289;
w_2 = 0.038;

```

```
//For component 1
Tr_1 = T/Tc_1;//Reduced temperature
//At reduced temperature
B1_0 = 0.083 - (0.422/(Tr_1)^(1.6));
B1_1 = 0.139 - (0.172/(Tr_1)^(4.2));
// We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B_11 = ((B1_0+(w_1*B1_1))*(R*Tc_1))/Pc_1;//[m^(3)/
    mol]

// Similarly for component 2
Tr_2 = T/Tc_2;//Reduced temperature
// At reduced temperature Tr_2,
B2_0 = 0.083 - (0.422/(Tr_2)^(1.6));
B2_1 = 0.139 - (0.172/(Tr_2)^(4.2));
B_22 = ((B2_0 + (w_2*B2_1))*(R*Tc_2))/Pc_2;//[m^(3)/
    mol]

//For cross coeffcient
Tc_12 = (Tc_1*Tc_2)^(1/2);//[K]
w_12 = (w_1 + w_2)/2;
Zc_12 = (Zc_1+Zc_2)/2;
Vc_12 = (((Vc_1)^(1/3)+(Vc_2)^(1/3))/2)^(3);//[cm
    ^(3)/mol]
Vc_12 = Vc_12*10^(-6);//[m^(3)/mol]
Pc_12 = (Zc_12*R*Tc_12)/Vc_12;//[N/m^(2)]

// Now we have,(B_12*Pc_12)/(R*Tc_12) = B_0 + (w_12*
    B_1)
// where B_0 and B_1 are to be evaluated at Tr_12
Tr_12 = T/Tc_12;
// At reduced temperature Tr_12
B_0 = 0.083 - (0.422/(Tr_12)^(1.6));
B_1 = 0.139 - (0.172/(Tr_12)^(4.2));
B_12 = ((B_0+(w_12*B_1))*R*Tc_12)/Pc_12;//[m^(3)/mol
    ]

// For the mixture
B = y1^(2)*B_11 + 2*y1*y2*B_12+y2^(2)*B_22;//[m^(3)/
```

```
    mol]
// Now we have to calculate molar volume i.eV1 and
    V2 at given conditions

// At state 1,
deff('[y]=f(V)','y=V^(2)-(R*T/P1)*V-(B*R*T)/P1');
V_1=fsolve(-1,f);
V_2=fsolve(1,f);
// We will take root near to (R*T)/P, i.e V_2
V1 = V_2;//[m^(3)/mol]

// At state 2,
deff('[y]=f1(V)','y=V^(2)-(R*T/P2)*V-(B*R*T)/P2');
V_3=fsolve(-1,f1);
V_4=fsolve(1,f1);
V2 = V_4;//[m^(3)/mol]

// Work done per mole of air is given by, w=integral
    (Pdv)
// Z = 1 + (B/V)
// (P*V)/(R*T) = 1 +( B/V)
// P = (R*T)/V+(B*R*T)/V^(2)
// w = integrate('(R*T/V)+(B*R*T)/V^(2)','V',V1,V2)
// Under isothermal conditions,
w = R*T*log(V2/V1)-B*R*T*((1/V2)-(1/V1));
w = w*10^(-3);//[kJ/mol]
printf(" The work done is %f kJ/mol",w);
```

Scilab code Exa 3.6 Calculation of work done

```
clear;
clc;
funcprot(0);

//Example - 3.6
```

```
//Page number - 88
printf("Example - 3.6 and Page number - 88\n\n");


//Given
T = 125+273.15;//[K] - Temperature
P1 = 1;//[bar] - Initial pressure
P1 = P1*10^(5);//[N/m^(2)]
P2 = 60;//[bar] - Final pressure
P2 = P2*10^(5);//[N/m^(2)]
R = 8.314;//[J/mol*K] - Universal gas constant
Tc = 416.3;//[K] - Critical temperature
Pc = 66.80*10^(5);//[N/m^(2)] - Critical pressure

// (1)
// Virial equation of state, Z = 1 + (B/V)+(C/V^(2))
// (P*V)/(R*T) = 1 + (B/V)+(C/V^(2))
// P = (R*T)/V+(B*R*T)/V^(2)+(C*R*T)/V^(3)
// w = integral(PdV)=R*T*log(V2/V1)-(B*R*T)*(1/V2-1/
   V1)-(C*R*T/2)*(1/V2^(2)-1/V1^(2))

B = -207.5;//[cm^(3)/mol] - Second virial
   coefficient
B = -207.5*10^(-6);//[m^(3)/mol]
C = 18200;//[cm^(6)/mol^(2)] - Third virial
   coefficient
C = 18200*10^(-12);//[m^(6)/mol^(2)]

// We need to calculate molar volume at state 1 and
   2,
// At state 1,P = P1,
// V^(3)-(R*T/P)*V^(2)-(B*R*T/P)*V-(C*R*T/P)=0
// Solving the cubic equation
deff('[y]=f1(V)','y=V^(3)-(R*T/P1)*V^(2)-(B*R*T/P1)*
   V-(C*R*T/P1)');
V_1=fsolve(-1,f1);
V_2=fsolve(0,f1);
V_3=fsolve(10,f1);
```

```
// The cubic equation has only 1 real root,other two
    roots are imaginary.
V1 = V_3;

// Similarly at state 2,P=P2
// V^(3) - (R*T/P)*V^(2) - (B*R*T/P)*V - (C*R*T/P) =
    0
// Solving the cubic equation
deff('[y]=f2(V)','y=V^(3)-(R*T/P2)*V^(2)-(B*R*T/P2)*
   V-(C*R*T/P2)');
V_4=fsolve(-1,f2);
V_5=fsolve(0,f2);
V_6=fsolve(1,f2);
V2 = V_6;
// Finally work done is given by,
w1 = R*T*log(V2/V1)-(B*R*T)*(1/V2-1/V1)-(C*R*T/2)
   *(1/V2^(2)-1/V1^(2));//[J/mol]
w1 = w1*10^(-3);//[kJ/mol]
printf(" (1).The work done using given virial
   equation of state is %f kJ/mol\n",w1);

// (2)
// Virial equation of state, Z = 1+(B*P)/(R*T)+((C-B
   ^(2))/(R*T)^(2))*P^(2)
// (P*V)/(R*T)= 1+(B*P)/(R*T)+((C-B^(2))/(R*T)^(2))*
   P^(2)
// V = (R*T)/P+B+((C-B^(2))/(R*T))*P
// Differentiating both sides by P and integrating
   we get,
// w = integral(PdV)=-(R*T)*log(P2/P1)+((C-B^(2))
   /(2*R*T))*(P2^(2)-P1^(2))
w2 = -(R*T)*log(P2/P1) + ((C-B^(2))/(2*R*T))*(P2^(2)
   -P1^(2));//[J/mol]
w2 = w2*10^(-3);//[kJ/mol]
printf(" (2).The work done using given virial
   equation of state is %f kJ/mol\n",w2);

// (3)
// Van der Walls equation of state is given by,
a = (27*(R^(2))*(Tc^(2)))/(64*Pc);//[Pa*m^(6)/mol
    ^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]
// P = ((R*T)/(V-b))-a/(V^(2));//[N/m^(2)]
// w = integral(PdV)=R*T*log((V2-b)/(V1-a))+a*(1/V2
    -1/V1)
// The cubic form of van der Walls equation of state
     is given by,
// V^(3) - (b+(R*T)/P)*V^(2) + (a/P)*V - (a*b)/P = 0
// Solving the cubic equation for P=P1
deff('[y]=f3(V)','y=V^(3)-(b+(R*T)/P1)*V^(2)+(a/P1)*
    V-(a*b)/P1');
V2_1=fsolve(1,f3);
V2_2=fsolve(10,f3);
V2_3=fsolve(100,f3);
// The above equation has 1 real and 2 imaginary
    roots. We consider only real root (V2_3).

// Similarly at state 2,for P=P2,
deff('[y]=f4(V)','y=V^(3)-(b+(R*T)/P2)*V^(2)+(a/P2)*
    V-(a*b)/P2');
V2_4=fsolve(1,f4);
V2_5=fsolve(10,f4);
V2_6=fsolve(100,f4);
// The above equation has 1 real and 2 imaginary
    roots. We consider only real root (V2_6).
// Finally work done is given by
w3 = R*T*log((V2_6-b)/(V2_3-b))+a*(1/V2_6-1/V2_3);//
    [J/mol]
w3 = w3*10^(-3);//[kJ/mol]
printf(" (3).The work done using  van der Walls
    equation of state is %f kJ/mol\n",w3);

// (4)
// Redlich Kwong equation of state,
a_1 = (0.42748*(R^(2))*(Tc^(2.5)))/Pc;//[Pa*m^(6)*K
    ^(1/2)/mol]
```

```
b_1 = (0.08664*R*Tc)/Pc;//[m^(3)/mol]
// P = ((R*T)/(V-b_1))-(a_1/(T^(1/2)*V*(V+b_1)));//[N/m^(2)]
// Work done is given by
// w = R*T*log((V2-b)/(V1-b))-a/T^(1/2)*integrate('1/V*(V+b)','V',V1,V2)
// Using the factorization 1/(V*(V+b))=(1/b)*((1/V)-(1/V+b)),we get
// w = R*T*log((V2-b)/(V1-b))-(a/(b*T^(1/2)))*(log(V2/V1)-log((V2+b)/(V1+b))
// Now we have calculate V1 and V2,
// The cubic form of Redlich Kwong equation of state is given by,
// V^(3) - ((R*T)/P)*V^(2) - ((b_1^(2)) + ((b_1*R*T)/P) - (a/(T^(1/2)*P))*V - (a*b)/(T^(1/2)*P) = 0
// Solving the cubic equation at state 1,
deff('[y]=f5(V)','y=V^(3)-((R*T)/P1)*V^(2)-((b_1^(2))+((b_1*R*T)/P1)-(a_1/(T^(1/2)*P1)))*V-(a_1*b_1)/(T^(1/2)*P1)');
V3_1=fsolve(1,f5);
V3_2=fsolve(10,f5);
V3_3=fsolve(100,f5);
// The above equation has 1 real and 2 imaginary roots. We consider only real root (V3_3).

// Similarly at state 2,for P = P2,
deff('[y]=f6(V)','y=V^(3)-((R*T)/P2)*V^(2)-((b_1^(2))+((b_1*R*T)/P2)-(a_1/(T^(1/2)*P2)))*V-(a_1*b_1)/(T^(1/2)*P2)');
V3_4=fsolve(1,f6);
V3_5=fsolve(10,f6);
V3_6=fsolve(100,f6);
// The above equation has 1 real and 2 imaginary roots. We consider only real root (V3_6).
// Finally work done is given by
w4 = R*T*log((V3_6-b_1)/(V3_3-b_1))-(a_1/(b_1*T^(1/2)))*(log(V3_6/V3_3)-log((V3_6+b_1)/(V3_3+b_1)));//[J/mol]
```

```
w4 = w4*10^(-3);//[kJ/mol]
printf(" (3).The work done using Redlich Kwong
   equation of state is %f kJ/mol\n",w4);
```

Scilab code Exa 3.7 Proving a mathematical relation

```
clear;
clc;

//Example - 3.7
//Page number - 92
printf("Example - 3.7 and Page number - 92\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 3.7 on page number
    92 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 3.7 on page
    number 92 of the book.");
```

Scilab code Exa 3.8 Calculation of work done

```
clear;
clc;
funcprot(0);

//Example - 3.8
//Page number - 93
printf("Example - 3.8 and Page number - 93\n\n");

```

```

//Given
T = 20 + 273.15;//[K] - Temperature
P_1 = 140;//[kPa] - Initial pressure
P_1 = P_1*10^(3);//[Pa]
P_2 = 560;//[kPa] - Final pressure
P_2 = P_2*10^(3);//[Pa]
R = 1.987;//[cal/mol*K] - Universal gas constant

// Cp_0 = 1.648+4.124*10^(-2)*T - 1.53*10^(-5)*T^(2)
    + 1.74*10^(-9)*T^(3)
// Using adiabatic compression, P*V^(Y)=constant.
   For ideal gases
// P*(R*T/P)^(Y) = constant
// P^(1-Y)*T^(Y) = constant or,P1^(1-Y)*T1^(Y)=P2
   ^(1-Y)*T2^(Y)
// Now,at state 1, i.e at T=20[C]
Cp_1 = 1.648+4.124*10^(-2)*T-1.53*10^(-5)*T^(2)
   +1.74*10^(-9)*T^(3);//[cal/mol*K] - Heat capacity
    at constant pressure
Cv_1 = Cp_1 - R;//[cal/mol*K] - Heat capacity at
   constant volume
Y1 = Cp_1/Cv_1;// Ratio of heat capacities

// Now calculatung the temperature at state 2 (T2)
// (T2/T1)=(P1/P2)^((1-Y1)/Y1)
T_1 = T;
T_2 = ((P_1/P_2)^((1-Y1)/Y1))*T_1;//[K]

// Now calculating the mean temperature
T_mean = (T_1 + T_2)/2;//[K]
// At mean temperature
Cp_2 = 1.648+4.124*10^(-2)*T_mean - 1.53*10^(-5)*
   T_mean^(2) + 1.74*10^(-9)*T_mean^(3);//[cal/mol*K
   ] - Heat capacity at constant pressure
Cv_2 = Cp_2 - R;//[cal/mol*K] - Heat capacity at
   constant volume
Y2 = Cp_2/Cv_2;
```

```

// Calculating exit temperature
// Again using the realation ,(T2/T1)=(P1/P2)^((1-Y1)
   /Y1)
T_exit = ((P_1/P_2)^((1-Y2)/Y2))*T_1;//[K]
// Since value of mean temperature has not changed
   much the molar heat capacity ratio can be assumed
    to be same.Therefore
// w = -delta(U)=Cv_0*(T2-T1)
w = Cv_2*(T_1 - T_exit);//[cal/mol]
w = w*4.184;//[J/mol]

printf(" The work done for adiabatic compression is
   %f J/mol\n",w);
```

Scilab code Exa 3.9 Calculation of final temperature

```
clear;
clc;

//Example - 3.9
//Page number - 93
printf("Example - 3.9 and Page number - 93\n\n");

//Given
m_ice = 1000;//[g] - Mass of ice
m_water = 1000;//[g] - Mass of water
T_ice = 273.15;//[K] - Temperature of ice
T_water = 373.15;//[K] - Temperature of water
L = 79.71;//[cal/g] - Latent heat of melting of ice.

//(1)
Cp_1 = 1;//[cal/g-K] - Heat capacity at constant
   pressure
// Let the final temperature be T
// We assume that all of the ice melts.Energy taken
   up by ice is
// E1 = L*m_ice + m_ice*Cp_1*(T - T_ice)
// Energy given by hot water is,
// E2 = m_water*Cp_1*(T_water - T)
// No heat exchange with surrounding.Solving for T
T_1 = (m_ice*Cp_1*T_ice + m_water*Cp_1*T_water - L*
   m_ice)/(m_ice*Cp_1 + m_water*Cp_1);//[K]
T_1 = T_1 - 273.15;//[C]

printf(" (1).The final temperature (taking Cp_water
   = 1 cal/g-K) is %f C\n",T_1);
//Since the final temperature is greater than 273.15
    K,so our assumption that all of ice melts is
   correct

// (2)
// Cp_2 = 1.00874-0.7067*10^(-3)*T+15.93*10^(-6)*T
   ^(2)-83.8*10^(-9)*T^(3);
// From energy balance,we get L*m_ice + m_ice*
   integrate('Cp_2','T',0,T) + m_water*integrate('
   Cp_2','T',100,T) = 0;  (where T is in C)
// On putting the values and then simplifying we get
// 2.01748*T - 0.0007067*T^(2) + 1.062*10^(-5)*T^(3)
    - 4.19*10^(-8)*T^(4) - 20.8455 = 0
// Solving the above equation we get
deff('[y]=f1(T)','y = 2.01748*T - 0.0007067*T^(2) +
   1.062*10^(-5)*T^(3) - 4.19*10^(-8)*T^(4) -
   20.8455');
T_0 = fsolve(1,f1);//[C]
printf(" (2).The final temperature using specific
   heat capacity equation is %f C\n",T_0);
```

Scilab code Exa 3.10 Finding expressions for temperature and pressure

```
clear;
clc;

//Example - 3.10
//Page number - 95
printf("Example - 3.10 and Page number - 95\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 3.10 on page
   number 95 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 3.10 on
   page number 95 of the book.");
```

Scilab code Exa 3.11 Calculation of final pressure

```
clear;
clc;

//Example - 3.11
//Page number - 97
printf("Example - 3.11 and Page number - 97\n\n");

//Given
n = 1.5;// - ratio of heat capacities
T_1 = 500;//[K] - Initial temperature
T_2 = 1000;//[K] - Final temperature
P_1 = 1;//[bar] - Initial pressure
P_1 = P_1*10^(5);//[Pa]
R = 8.314;//[J/mol*K] - Universal gas constant

// The compression path is given by, P*V^(1.5) =
    constant
// P*(R*T/P)^(1.5) = constant
// P1^(-0.5)*T1^(1.5) = P2^(-0.5)*T2^(1.5)
P_2 = P_1*(T_1/T_2)^(-3);//[Pa]
P_2_final = P_2*10^(-5);//[bar] - Final pressure in
    bar
printf(" The final pressure is %f bar\n",P_2_final);

// From first law q - w = delta(U).
// First w and delta(U) are calculated and
    thereafter heat exchange is determined.
V_1 = R*T_1/P_1;//[m^(3)/mol] - Initial volume
V_2 = R*T_2/P_2;//[m^(3)/mol] - Final volume
w = ((P_1*V_1)/(n - 1))*(1 - (P_2/P_1)^(1 - 1/n));//
    [J/mol] - work done

// Mean temperature is given by,
T_mean = (T_1 + T_2)/2;//[K]

//Now, heat capacity at T_mean is given by,
Cp_0 = R*(3.3 + 0.63*10^(-3)*T_mean);//[J/mol*K]
Cv_0 = Cp_0 - R;//[J/mol*K]
//Therefore delta(U) is given by
del_U = Cv_0*(T_2 - T_1);//[J/mol] - Change in
    internal energy
q = w + del_U;//[J/mol] - heat change
printf(" The amount of heat supplied to the system
    is %f J/mol\n",q);
```

Scilab code Exa 3.12 Calculation of slope and work done

```
clear;
clc;

//Example - 3.12
```

```
//Page number - 99
printf("Example - 3.12 and Page number - 99\n\n");

//Given
P_1 = 150*10^(3);//[Pa] - Initial pressure
V_1 = 0.001;//[m^(3)] - Initial volume
P_2 = 1000*10^(3);//[Pa] - Final pressure
V_2 = 0.003;//[m^(3)] - Final volume

// At x = 0, Vt(total volume) = 0.001 m^(3),
   therefore x = (V_t - V_1)/A;  where A is area of
   cross section and x is length
// Force exerted b sprig is given by, F = Ps*A = k*x
    = k*(V_t - V_1)/A
// Ps = (k/A^(2))*(V_t - V_1)
// Total pressure = Initial pressre + Pressre due to
    spring
// P = P_1 + (k/A^(2))*(V_t - V_1)
// Let (k/A^(2)) = t (say)
// At state 2, i.e at P2 and V_t = V_2.
deff('[y]=f(t)','y=P_2-P_1 - t*(V_2-V_1)');
t = fsolve(1000,f);
// Therefore,pressure is related to total volume as
   P = P_1-t*(V_t - V_1)

// (a)
//slope = (k/A^(2))
printf(" (a).The slope of the line on P-Vt diagram
   is %e N/m^(5)\n",t);

// (b)
// Work done by the gas is given by  w=integral(
   PdVt)
w = integrate('P_1+t*(V_t-V_1)','V_t',V_1,V_2);//[J]
w = w*10^(-3);//[kJ]
printf(" (b).The work done by gas is %f kJ\n",w);
```

Scilab code Exa 3.13 Calculation of work done and final temperature

```
clear;
clc;

//Example - 3.13
//Page number - 99
printf("Example - 3.13 and Page number - 99\n\n");

//Given
V = 36;//[L] - Vol of gas on each side
P_1 = 1;//[atm] - pressure on left side of the
    piston
P_1 = P_1*101325;//[Pa]
T = 273.15;//[K]
P_2 = 3.375;//[atm] - Pressure on right side of the
    piston
P_2 = P_2*101325;//[Pa]
Y = 1.44;// Ratio of heat capacities
R = 8.314;//[J/mol*K] - Universal gas constnt

// (a)
// For total system, del(U_total) = Q.
// Onto gas on right hand side no heat is supplied,
    as the piston is non conducting. Therefore, for
    gas on the right hand side, del(U) = -W.
// As heat is slowly supplied to the left hand side,
    expansion on right hand side is slow and process
    is adiabatic.
// For gas on right hand side, PV^(Y) = constant.
// T_2/T_1 = (P_2/P_1)^((Y - 1)/Y)
T_right = T*(P_2/P_1)^((Y - 1)/Y);//[K]

Cv_0 = R/(Y-1);//[J/mol*K] - Heat capacity at
```

```
    constant volume.
// Now work done on the gas on right hand side is
    given by
// W = (P_1*V_1 - P_2*V_2)/(Y - 1) = R*(T_2 - T_1)/(
    Y - 1) = Cv_0*(T_1 - T_2)
W_left = Cv_0*(T - T_right);//[J/mol]
// Negative sign for the work done on LHS gas
    implies work is done on the gas

// For right hand side of the gas
// P*Vt = n*R*T
n = P_1*(V*10^(-3))/(R*T);// number of moles
W_right = (-W_left)*n;//[J] - We used negative sign
    for 'W_left' because it is negative in magnitude.
W_right = W_right/1000;//[kJ]
printf(" (a).Total work done on gas on the right
    hand side is %f kJ\n",W_right);

//(b)
printf(" (b).The final temperature of the gas on
    right side is %f K\n",T_right);

// (c)
// Pressure is same on both sides as piston is
    frictionless.
// The number of moles on both sides are also same
    as they have same temperature,presure and volume.
// We have (P_left*V_left)/T_left = (P_right*V_right
    )/T_right.
// Since P_left = P_right, (V_left/T_left) = (
    V_right/T-right) and also P*V^(Y) = constant.
V_right = V*(P_1/P_2)^(1/Y);//[L] - The total volume
     on right side

// The total volume on right side can also be
    calculated using P2*V2 = n*R*T2.
// Since total volume = 72 [L], therefore volume of
    left side is
```

```
V_left = 2*V - V_right;//[L]
T_left = T_right*(V_left/V_right);
printf(" (c).Final temperature of the gas on the
    left side is %f K\n",T_left);

//(d)
//The first law applied to the total system (left
    side and right side) gives.
//Q - W = del(U_left) + del(U_right)
//There is no net work done by the total system as
    the cylinder is closed at both ends.
Q = n*Cv_0*(T_left-T) + n*Cv_0*(T_right-T);//[J]
Q = Q/1000;//[kJ]
printf(" (d).Amount of heat added to the gas on the
    left side is %f kJ",Q);
```

Scilab code Exa 3.14 Calculation of powerand discharge head

```
clear;
clc;

//Example - 3.14
//Page number - 105
printf("Example - 3.14 and Page number - 105\n\n");


//Given
P_2 = 0.2;//[bar]
P_2 = P_2*10^(5);//[Pa]
int_dia_2 = 2.4*10^(-2);//[m] - internal diameter at
     state 2.
Q = 5*10^(-3);//[cubic metre/s] - Flow rate at point
     2.
den = 1000;//[kg/cubic metre] - density
delta_z = 1;//[m] - Difference in height
```

```
g = 9.81;//[m/s^(2)] - Acceleration due to gravity

// (1)
// Pressure at state 1 is atmospheric pressure and
   at state 2 is gauze pressure at state 2 +
   atmospheric pressure,thus
// (delta(P)/den) = (P2-P1)/den = P2/den
Vel_2 = Q/(3.14*(int_dia_2/2)^(2));//[m/s] -
   Velocity of water at state 2.
// Velocity at state 1 i negligible as compared to
   velocity at state 2,because the diameter of
   reservoir is very large as compared to diameter
   of pipe at state 2

// From bernaulli equation we get,
// -w = (delta(P)/den) + delta(v^(2))/2 + g*delta_z
w = -((P_2/den )+ (Vel_2^(2)/2) + (g*delta_z));//[J/
   kg]
// w multiplied by m = (den*Q), will give the fluid
   power.
m = den*Q;//[kg/s]
W_net = m*w;//[Watt]
printf(" (1).The fluid power is %f Watt\n",W_net);

//(2)
// Total discharge head developed by the pump is
   given by
// h = (delta(P)/den*g) + (Vel_2^(2)/2*g) + delta_z
h = (P_2/(den*g)) + (Vel_2^(2)/(2*g)) + delta_z;//[m
   ]
printf(" (2).Total discharge head developed by the
   pump is given by h = %f m',h);
```

Scilab code Exa 3.15 Calculation of discharge velocity

```
clear;
clc;

//Example - 3.15
//Page number - 106
printf("Example - 3.15 and Page number - 106\n\n");

//Given
T_1 = 1000;//[K] - Temperature at entry
P_1 = 0.6;//[MPa] - Pressure at entry
P_2 = 0.2;//[MPa] - Exit pressure
Vel_1 = 50;//[m/s] - Entry velocity
Y = 1.4;// Ratio of heat capacities
Mol_wt = 28;//[g/mol] - Molecular weight of air
Cp = 29.099;//[J/mol-K] - Specific heat capacity at
    constant pressure
Cp = (Cp/Mol_wt)*1000;//[J/kg-K]

// We know that for a flow process
// delta_H + delta_V^(2)/2 + delta_(g*z) = q - w
// Since process is adiabatic,therefore q = 0 and
    since no work is done by the gas, therefore w = 0
// Assuming there is no change in the potenial
    energy between entry and exit, we have
// delta_H + delta_V^(2)/2 = 0

// For a reversible process P*V^(Y) = constant and
    thus (T_2/T_1) = (P_2/P_1)^((Y-1)/Y)
T_2 = T_1*(P_2/P_1)^((Y-1)/Y);//[K] - Exit
    temperature

// delta_H + delta_V^(2)/2 = 0
// Vel_2^(2)/2 - Vel_1^(2)/2 - (H_1 - H_2)= 0
// Vel_2^(2)/2 - Vel_1^(2)/2 - Cp*(T_1 - T_2) = 0
Vel_2_square = 2*(Vel_1^(2)/2 + Cp*(T_1 - T_2));//[m
    ^(2)/s^(2)]
Vel_2 = (Vel_2_square)^(1/2);//[m/s]

```

```
printf(" The discharge velocity is %f m/s\n",Vel_2);
```

Scilab code Exa 3.16 Calculation of change in enthalpy

```
clear;
clc;

//Example - 3.16
//Page number - 107
printf("Example - 3.16 and Page number - 107\n\n");

//Given
P_entry = 10;//[bar] - Pressure at entry
V_entry = 200;//[m/s] - Velocity at entry
P_exit = 1;//[bar] - Vressure at exit
V_exit = 800;//[m/s] - Velocity at exit
g = 9.81;//[m/s^(2)] - Acceleration due to gravity

//Heat balance gives
// delta_H + (delta_V^(2))/2 + g*delta_z = q - w
//delta_H = q - w - (delta_V^(2))/2
//From nozzle no work is extracted,therefore
delta_H = -(V_exit^(2)- V_entry^(2))/2;//[J/kg]
delta_H = delta_H*10^(-3);//[kJ/kg]

printf(" The change in enthalpy per kg of steam is
   %f kJ/kg",delta_H);
```

Scilab code Exa 3.17 Calculation of work done and change in enthalpy

```
clear;
clc;

```

```
//Example - 3.17
//Page number - 111
printf("Example - 3.17 and Page number - 111\n\n");


//Given
T_1 = 280;//[K] - Temperature at entry
P_1 = 100;//[kPa] - Pressure at entry
T_2 = 400;//[K] - Temperature at exit
P_2 = 600;//[kPa] - Pressure at exit
m = 0.02;//[kg/s] - Mass flow rate
m = m*10^(3);//[g/s]
heat_loss = 16;//[kJ/kg]

//Cp_0 = 28.11 + 0.1967*10^(-2)*T + 0.4802*10^(-5)*T
   ^(2) - 1.966*10^(-9)*T^(3)
//delta_H = q - w (neglecting kinetic and potential
   changes)
//delta_H = integral(Cp_0*dT)
delta_H = integrate('28.11 + 0.1967*10^(-2)*T +
   0.4802*10^(-5)*T^(2) - 1.966*10^(-9)*T^(3)','T',
   T_1,T_2);//[J/mol - Enthalpy change
printf(" Change in enthalpy is %f J/mol\n",delta_H);

//Molecular weight of air(21 vol% O2 and 79 vol% N2)
   =(0.21*32)+(0.79*28)= 28.84 g/mol
Mol_wt = 28.84;//[g/mol]
q = - (heat_loss*Mol_wt);//[J/mol]
w = q - delta_H;//[J/mol]
printf(" The work done per mole of air is %f J/mol\n
   ",w);
//the negative sign implies that work is done on the
    compressor.

n = m/Mol_wt;//[mol/s] - Mole flow rate
W_net = delta_H*n;//[W]
W_net = -W_net*10^(-3);//[kW]
printf(" And the necessary power input to the
    compressor is %f kW\n",W_net);
```

Scilab code Exa 3.18 Calculation of work done per unit mass

```
clear;
clc;

//Example - 3.18
//Page number - 112
printf("Example - 3.18 and Page number - 112\n\n");


//Given
T_1 = 300;//[K] - Temperature at entry
P_1 = 100;//[kPa] - Pressure at entry
P_2 = 900;//[kPa] - Pressure at exit
R = 8.314;//[J/mol*K] - Universal gas constant

// (a)
// Reversible adiabatic compression
Y = 1.4;//Ratio of specific heat capacities
// For ideal gas, P*V^(Y)= constant and it takes the
    form of (T_2/T_1) = (P_2/P_1)^((Y-1)/Y)
T_2 = T_1*(P_2/P_1)^((Y - 1)/Y);//[K]
// The work exchange for adiabatic process is given
   by
// W_adia = -delta_H = -Cp*(T2-T1) = Cp*(T1-T2) = ((
   Y*R)/(Y-1))*(T1-T2)
W_adia = ((Y*R)/(Y - 1))*(T_1 - T_2);//[J/mol] -work
    done
// Molecular weight of air(21 vol% O2 and 79 vol% N2
   )=(0.21*32)+(0.79*28)= 28.84 g/mol
Mol_wt = 28.84;//[g/mol]
W_adia = W_adia/Mol_wt;//[J/g]
printf(" (a).The compressor work done for reversible
    adiabatic compession is %f J/g\n",W_adia);

//(b)
//Isothermal compression
//W_iso = -integral(V*dP) = -integral((R*T/P)*dP) =
   R*T*ln(P_2/P_1)
W_iso = -R*T_1*log(P_2/P_1);//[J/mol]
W_iso = W_iso/Mol_wt;//[J/g]
printf(" (b).The compressor work done for isothermal
    compession is %f J/g\n",W_iso);
//Note that in isothermal compression between the
   same states work done is less as compared to
   reversible adiabatic compression.

//(c)
//Ideal two-stage compression
n = 1.3;//Polytropic exponent.
//Minimum work done in two stage compression  is
   given by
//W_comp = ((2*n*R*T_1)/(n-1))*[1-(P_x/P_1)^(n-1)/n]
//where for minimum work, (P_x/P_1) = (P_x/P_2), and
    thus
P_x = (P_1*P_2)^(1/2);//[kPa]
//therefore, work done is given by,
W_comp = ((2*n*R*T_1)/(n-1))*[1-(P_x/P_1)^((n-1)/n)
   ];//[J/mol]
W_comp = W_comp/Mol_wt;//[J/g]
printf(" (c).The compressor work done for ideal two-
   stage compession is %f J/g\n",W_comp);
```

Scilab code Exa 3.19 Calculation of inlet and outlet velocity and power

```
clear;
clc;

```

```
//Example - 3.19
//Page number - 113
printf("Example - 3.19 and Page number - 113\n\n");


//Given
T_1 = 600;//[C] - Temperature at entry
P_1 = 15;//[MPa] - Pressure at entry
T_2 = 400;//[K] - Temperature at exit
P_2 = 100;//[kPa] - Pressure at exit
A_in = 0.045;//[metre square] - flow  in area
A_out = 0.31;//[metre square] - flow out area
m = 30;//[kg/s] - mass flow rate.

//At 15 MPa and 600 C,it has been reported in the
   book that the properties of steam are,
Vol_1 = 0.02491;//[m^(3)/kg] - Specific volume
H_1 = 3582.3;//[kJ/kg] - Enthalpy
// m = den*vel*A = (Vel*A)/Vol, substituting the
   values
vel_1 = (m*Vol_1)/A_in;//[m/s] - Velocity at point
   1.
printf(" The inlet velocity is %f m/s\n",vel_1);

//At 100 MPa (saturated vapour),it has been reported
    in the book that the properties of steam are,
   T_sat = 99.63 C, and
Vol_vap_2 = 1.6940;//[m^(3)/kg] - specific volume of
    saturated vapour.
H_vap_2 = 2675.5;//[kJ/kg] - Enthalpy os saturated
   vapour.
vel_2 = (m*Vol_vap_2)/A_out;//[m/s] - Velocity at
   point 2.
printf(" The exit velocity is %f m/s\n",vel_2);

//From first law we get, q - w =delta_H + delta_V
   ^(2)/2
//q = 0, therefore, -w = delta_H + delta_V^(2)/2
```

```
delta_H = H_vap_2 - H_1;//[kJ/kg] - change in
    enthalpy.
delta_V_square = (vel_2^(2) - vel_1^(2))/2;//[J/kg]
delta_V_square = delta_V_square*10^(-3);//[kJ/kg]
w = -(delta_H + delta_V_square);//[J/kg]
W_net = w*m;//[kW]
W_net = W_net*10^(-3);//[MW] - power produced.
printf(" The power that can be produced by the
    turbine is %f MW",W_net);
```

Scilab code Exa 3.20 Proving a mathematical relation

```
clear;
clc;

//Example - 3.20
//Page number - 117
printf("Example - 3.20 and Page number - 117\n\n");

//(1)
//This part involves no numerical components
//For prove refer to this example 3.20 on page
    number 117 of the book.

//(2)
//Given,
R = 8.314;//[J/mol-K] - Universal gas constant
Cp_0 = 2.5*R;//[J/mol-K] -  Specific heat capacity
    at constant pressure
Cv_0 = 1.5*R;//[J/mol-K] -  Specific heat capacity
    at constant volume
T_L = 300;//[K] - Temperature at which port
    properties are constant.

Y = Cp_0/Cv_0;// Ratio of heat capacities.
```

```
//From part(1) we obtained the relation,
// T_2 = 1/(((P_2-P_1)/(Y*T_L*P_2))+(P_1/(P_2*T_1)))
// Not that when P_2 >> P_1 ,T_2 approaches Y*T_L
   and thus
T_2 = Y*T_L;//[K]
printf(" (b).The final temperature is %f K",T_2);
```

Scilab code Exa 3.21 Determination of equilibrium temperature

```
clear;
clc;

//Example - 3.21
//Page number - 119
printf("Example - 3.21 and Page number - 119\n\n");

//Given
T_1 = 40 + 273.15;//[K] - Initial temperature.
P_1 = 1;//[bar] - Initial pressure.
P_1 = P_1*10^(5);//[Pa]
Vol_1 = 0.01;//[cubic metre] - Initial volume of the
    cylinder.
T_2 = 100 + 273.15;//[K] - Final temperature.
P_2 = 100;//[kPa] - Final pressure.
P_2 = P_2*10^(5);//[Pa]
Vol_2 = 0.02;//[cubic metre] - Final volume of the
   cylinder.
Cp = 1.005;//[J/g-K] - Specific heat capacity at
   constant pressure.
Cv = 0.718;//[J/g-K] - Specific heat capacity at
   constant volume.
Mol_wt = 28.84;//[g/mol] - Molecular weight of air.
R = 8.314;//[J/mol-K] - universal gas constant

delta_Vol = Vol_2 - Vol_1;// [cubic metre] - Change
```

```
    in volume.
// Assuming ideal gas P*V = R*T
V_1 = (R*T_1)/P_1;// [m^(3)/mol] - Initial specific
    volume.
// Therefore,the total number of moles initially in
    the system is,
n_1 = (Vol_1/V_1);// [mol]
m_1 = n_1*Mol_wt;// [g] - Initial mass of the system
    .
Y = Cp/Cv;//Ratio of heat capacities

// The energy balance equation is given by
// -P*delta_Vol + H_liq*(m_2 - m_1) = m_2*Cv*(P*V2)/
    R - m_1*Cv*T_1
// m_2*Cv*(P*V2)/R = (Cv*P_1*Vol_2)/R
// Cv/R = 1/(Y-1)
// Since pressure of the gas in system is assumed
    constant,therefore it remains at 1 bar and thus P
     = P_1,
H_liq = Cp*T_2;// [J/g] - Enthalpy of liquid
m_2 = (P_1*delta_Vol + ((P_1*Vol_2)/(Y-1)) + H_liq*
    m_1 - m_1*Cv*T_1)/H_liq;//[g]

//The mass entering the assembly during the filling
    process is given by
m = m_2 - m_1;//[g]
n_2 = m_2/Mol_wt;//[mol] - Number of moles in the
    final state.
V_2 = Vol_2/n_2;//[m^(3)/mol] - Final specific
    volume.
// Therfore,final temperature is given by,
T_2 = (P_1*V_2)/R;//[K] - Final temperature.

printf(" The final equilibrium temperature is %f K\n
    ",T_2);
printf(" The mass entering through the valve is %f g
    \n",m);
```

Scilab code Exa 3.22 Determination of mass

```
clear;
clc;

//Example - 3.22
//Page number - 122
printf("Example - 3.22 and Page number - 122\n\n");

//Given
V_total = 5;//[L] - Volume of pressure cooker.
V_total = V_total*10^(-3);//m^(3)
P_gauze = 15;//[psi] - Operating pressure (gauze)of
    pressure cooker.
P_gauze = (P_gauze/14.5)*10^(5);//[N/m^(2)]
P_atm = 0.966*10^(5);//[N/m^(2)] - Atmospheric
    pressure.
m_1 = 1;//[kg] - Initial mass.
t = 30*60;//[s] - Total time.
J = 500;//[W] - Rate of heat supply

P_abs = P_gauze + P_atm;//[N/m^(2)] - Absolute
    pressure.
//The energy balance equqtion gives,
// Q = m_e*H_e +(m_2*U_2 - m_1*U_1), where 'm_e' is
    the mass exit  from the system and 'H_e' is
    enthalpy at exit conditions.

//It has been reported in the book that from steam
    table at P_abs,
T_sat = 120.23;//[K]- Saturated temperature
V_liq = 0.001061;//[m^(3)/kg] - specific volume of
    liquid.
V_vap = 0.8857;//[m^(3)/kg] - specific volume of
```

```
    vapour.
U_liq = 504.49;//[kJ/kg] - specific internal energy
    of liquid.
U_vap = 2529.5;//[kJ/kg] - specific internal energy
    of vapour.
H_liq = 504.70;//[kJ/kg] - specific enthalpy of
    liquid.
H_vap = 2706.7;//[kJ/kg] - specific internal energy
    of vapour.

//We know that total volume occupied by 1 kg of
    fluid is
//V_total = (1-x)*V_liq + x*V_vap
x1 = (V_liq - V_total)/(V_liq - V_vap);//[g]

//Internal energy at this state is
U_1 = (1-x1)*U_liq + x1*U_vap;//[kJ/kg] - specific
    internal energy
U_1_net = m_1*U_1;//[kJ] - Internal energy

//The amount of heat suplied is given by,
J_net = J*t;//[J] - Net heat supplied.
J_net = J_net*10^(-3);//[kJ]

//Let the dryness factor at the end of the process
    be x
//The specific properties of the liquid and vapour
    remain same as P and T_sat are the same in the
    cooker.
//Let the total mass of H2O (liquid + vapour) at the
     end of the process be 'm' kg.
// V_total/m = (1-x)*(V_liq) + x*V_vap ......
    equqtion(1)

//the specific internal energy at the end of process
     is
//U = (1-x)*U_liq + x*U_vap
//The total internal energy at the end of the
```

```
    process is
//U_net = m*U = x*[(1-x)*U_liq + x*U_vap]

//The energy balance equqtion gives,
// Q = m_e*H_e +(m_2*U_2 - m_1*U_1), where 'm_e' is
    the mass exit  from the system and 'H_e' is
    enthalpy at exit conditions.
//Since the vapour which exits out have enthalpy
    equal to that of saturated vapour,we get on
    simplification
// 900 = (1-m)*(2706.7) + m*((1-x)*504.49 + x
    *2529.5) - 513.5..........equation(2)
// The second equation on simplification becomes
// x = ((0.005/m) - 0.001061)/0.884639

// Putting the expression of x in first equation and
     then simplifying , we get
// - 1293.2 = -2202.21*m + 11.445 - 2.429*m
m = (11.445+1293.2)/(2202.21+2.429);//[kg]

// Therefore x can be calculated as
x = ((0.005/m) - 0.001061)/0.884639;

// Therfore total water (liquid + vapour) present in
     the pressure cooker at the end of the process is
     m kg.
m_vapour = x*m;//[kg] - Mass of vapour
m_liquid = (1-x)*m;//[kg] - Mass of vapour

printf(" Total water (liquid + vapour) present in
    the pressure cooker at the end of the process is
    %f kg\n",m);
printf(" The mass of vapour is %f kg\n",m_vapour);
printf(" The mass of liquid is %f kg\n",m_liquid);
```

Chapter 4

The Secomd Law and Its Applications

Scilab code Exa 4.1 Calculation of entropy change

```
clear;
clc;

//Example - 4.1
//Page number - 148
printf("Example - 4.1 and Page number - 148\n\n");

//Given
n = 1000;//[mol]
T = 400;//[K]
P_1 = 100;//[kPa]
P_2 = 1000;//[kPa]
R = 8.314;//[J/mol*K] - Universal gas constant

//(a)
T_surr = 400;//[K] - surrounding temperature
// Total change in entropy of the system is given by
// delta_S_sys = n*(Cp_0*log(T_2/T_1) - R*log(P_2/
   P_1))
```

```
// The process being isothermal the first term is
   zero and the total entropy change of the system
   is
delta_S_sys_a = - n*R*log(P_2/P_1);//[J/K]
delta_S_sys_a = delta_S_sys_a*10^(-3);//[kJ/K]

// Since the process is reversible therefore
Q_sys = T_surr*delta_S_sys_a;//[kJ] - Heat change in
    the system
// Negative sign in the value of Q_sys implies that
   heat is released from the system and is released
   to the surroundings,therefore
Q_surr = - Q_sys;//[kJ] - Heat change in the
   surrounding
delta_S_surr_a = Q_surr/T_surr;//[kJ/K]

delta_S_univ_a = delta_S_sys_a + delta_S_surr_a;//[
   kJ/K]
// We get delta_S_univ = 0, which is true for a
   reversible process

printf(" (a).The entropy change of the gas is given
   by  delta_S_sys = %f kJ/K \n",delta_S_sys_a);
printf("      The entropy change of the surrounding
   is, delta_S_surr = %f kJ/K \n",delta_S_surr_a);
printf("      The total entropy change of the gas is,
    delta_S_univ = %f kJ/K \n\n",delta_S_univ_a);

//(b)
T_surr_b = 300;//[K] - surrounding temperature
// Since the initial and final states are fixed
   therefore the entropy change of the system is
   same whether the process is carried out
   reversibly or irreversibly.
delta_S_sys_b = delta_S_sys_a;

// Work done under reversible condition is given by
// W = integral(P*dV) = integral(((R*T)/V)*dV) = R*T
```

```
    *log(V_2/V_1)
// For ideal gas we have, P1*V1/T1 = P2*V2/T2 or, V2
    /V1 = P1/P2 (for isothermal conditions)
W = R*T*log(P_1/P_2);//[J/mol]
W = W*10^(-3);//[kJ/mol]
// 20% extra work has to be done for the system to
    reach the same final state as under reversible
    conditions. Therefore
W = W*(120/100);//[kJ/mol]
W = W*n;//[kJ] - Total work done for n moles

// Using the first law we have  delta_U = Q - W. Now
     under isothermal conditions for ideal gas,
    delta_U = 0. Therefore,
Q = -W;
// It implies that whatever work is done on the
    system is lost as heat to the surroundings.
// Since heat is gained by the surroundings
    therefore
delta_S_surr_b = Q/T_surr_b;//[kJ/K]

delta_S_univ_b = delta_S_sys_b + delta_S_surr_b;//[
    kJ/K]

printf(" (b).The entropy change of the gas is given
    by  delta_S_sys = %f kJ/K \n",delta_S_sys_b);
printf("      The entropy change of the surrounding
    is, delta_S_surr = %f kJ/K \n",delta_S_surr_b);
printf("      The total entropy change of the gas is,
     delta_S_univ = %f kJ/K \n\n",delta_S_univ_b);
```

Scilab code Exa 4.2 Determination of whether the process is reversible or not

```
clear;
```

```
clc;

//Example - 4.1
//Page number - 148
printf("Example - 4.1 and Page number - 148\n\n");

// Given
T = 400;//[K] - Temperature
P_1 = 500*10^(3);//[Pa] - Initial pressure
P_2 = 100*10^(3);//[Pa] -  Final pressure
V_1 = 750*10^(-6);//[m^(3)] - Initial volume
W_actual = 0.55*10^(3);//[J] - Actual work done
R = 8.314;//[J/mol*K] - Universal fas constant

// Suppose that the surroundings are at 400 K.
// Therefore the process is externally reversible as
     temperature of the surroundings is same as
   system temperature.
// The number of moles is given by
n = (P_1*V_1)/(R*T);//[mol]
// The entropy change of ideal gas under isothermal
   condition is given by
delta_S_sys = - n*R*log(P_2/P_1);//[J/mol]

// The heat supplied to the system in the internally
     reversible process is
Q_theot = T*delta_S_sys;//[J]
// Since the process is isothermal therefore,
   workdone is given by
W_theot = Q_theot;//[J] - Theoritical work done
// Since actual work done by the gas is 0.55 kJ
   therefore actual heat supplied is also 0.55 kJ
   because under isothermal conditions delta_U = 0
Q_actual = W_actual;

// Since Q_theot > Q_actual, so the process is
   irreversible
printf(" Since, Q_theot = %f J  is greater than
   Q_actual = %f J\n",Q_theot,Q_actual);
printf(" Therefore,the process is internally
   irreversible");

// Moreover delta_S_sys is same whether the process
   is reversible or irreversible as the initial and
   final states is the same.
// In the reversible process higher amount of heat
   is supplied (as compared to irreversible) due to
   which delta_S_sys take place.
// In the irreversible process the entropy of system
    increases due two reasons : heat supplied and
   entropy generation
// So in the irreversible case amount of heat
   supplied is less as compared to reversible case
   as entropy generation term also adds to the
   entropy change of system
// delta_S_sys = Q/T_b + S_gen
S_gen = delta_S_sys - (Q_theot/T);//[J/K]
// The entropy generated may be due to friction and
   other dissipayive effects or due to non-quasi-
   static expansion
```

Scilab code Exa 4.3 Calculation of final pressure temperature and increase in entropy

```
clear;
clc;

//Example - 4.3
//Page number - 150
printf("Example - 4.3 and Page number - 150\n\n");

// Given
R = 8.314;//[J/mol*K] - Universal gas constant
```

```
// For side A
V_A = 1;//[L] - Volume
V_A = V_A*10^(-3);//[m^(3)]
T_A = 300;//[K] - Temperature
P_A = 2;//[atm] - Pressure
P_A = P_A*101325;//[Pa]

// For side B
V_B = 1;//[L] - volume
V_B = V_B*10^(-3);//[m^(3)]
T_B = 300;//[K] - Temperature
P_B = 1;//[atm] - Pressure
P_B = P_B*101325;//[Pa]

// From first law final temperature and pressure are
    given by (example 3.30)
// T = ((n_A*T_A) + (n_B*T_B))/(n_A + n_B)
// P = ((P_A*V_A) + (P_A*V_B))/(V_A + V_B)

// Since in this case T_A = T_B, therefore final
   pressure is given by
P = ((P_A*V_A) + (P_B*V_B))/(V_A + V_B);//[Pa]
P = P/101325;//[atm]

printf(" The final temperature is %f K\n",T_A);
printf(" The final pressure is %f atm\n",P);

// The number of moles of air on each side are
n_A = (P_A*V_A)/(R*T_A);//[mol]
n_B = (P_B*V_B)/(R*T_B);//[mol]

delta_S_A = -n_A*R*log((P*101325)/P_A);//[J/K] -
   Entropy change on side A
delta_S_B = -n_B*R*log((P*101325)/P_B);//[J/K] -
   Entropy change on side B
delta_S_sys = delta_S_A + delta_S_B;//[J/K] - Total
   entropy change of system

```

```
// Since the system is insulated there is no heat
   exchange with the surroundings, therefore entropy
    change of surrounding is zero
delta_S_surr = 0;//[J/K]
delta_S_univ = delta_S_sys + delta_S_surr;//[J/K]
printf(" The total increase in entropy is %f J/K",
   delta_S_univ);

// The entropy change of the system can also be
   writtten as
// delta_s_sys = Q/T_b + S_gen
// Since there is no heat transfer, therefore
S_gen = delta_S_univ;//[J/K]
// The process is reversible because of entropy
   generation due to spontaneous release of piston.
```

Scilab code Exa 4.4 CAlculation of final temperature heat transfer and change of entropy

```
clear;
clc;

//Example - 4.4
//Page number - 151
printf("Example - 4.4 and Page number - 151\n\n");

// Given
V_vessel = 0.2;//[m^(3)] - Volume of the vessel
P_1 = 10;//[bar] - Initial pressure inside the
   vessel
P_1 = P_1*10^(5);//[Pa]
P_2 = 3.5;//[bar] - Final pressure inside the vessel
P_2 = P_2*10^(5);//Pa
T_1 = 250 + 273.15;//[K] - Initial temperature of
   the vesssel
```

```
R = 8.314;//[J/mol*K] - Universal gas constant

// (a)
// At 10 bar and 250 C the steam is superheated.
   From steam table as reported in book we have
V_1 = 0.2327;//[m^(3)/kg] - specific volume
U_1 = 2709.9;//[kJ/kg] - specific internal energy
H_1 = 2942.6;//[kj/kg] - Specific enthalpy
S_1 = 6.9247;//[kJ/kg-K] - Specific entropy
// the quantity of steam is given by
m = V_vessel/V_1;//[kg]

// At final state
V_2 = 0.2327;//[m^(3)/kg] - Molar volume
V_liq_2 = 0.001079;// [m^(3)/kg]
V_vap_2 = 0.5243;// [m^(3)/kg]
// Since overall volume lies between saturated
   liquid and saturated vapour therefore the steam
   is saturated and its dryness fraction at final
   state is given by
x = (V_2 - V_liq_2)/(V_vap_2 - V_liq_2);
// Final temperature = T_sat (at 3 bar) from steam
   table
T_final = 138.88;//[C]

// At 3.5 bar saturated conditions
S_liq = 1.7275;//[kJ/kg-K] - Entropy of saturated
   liquid
S_vap = 6.9405;//[kJ/kg-K] - Entropy of saturated
   vapour
U_liq = 583.95;//[kJ/kg] - Internal energy of
   saturated liquid
U_vap = 2548.9;//[kJ/kg] - Internal energy of
   saturated vapour
// Therefore at final state
U_2 = U_liq*(1 - x) + x*U_vap;//[kJ/kg]
S_2 = S_liq*(1 - x) + x*S_vap;//[kJ/kg-K]
Q_1 = m*(U_2 - U_1);//[kJ]
```

```
delta_S_1 = m*(S_2 - S_1);//[kJ/kg-K]

printf(" (a).The final temperature is %f C\n",T_final);
printf("      The amount of heat transfer is %f kJ\n",Q_1);
printf("      The change of entropy is %f kJ/kg-K\n\n",delta_S_1);

// (b)
Y = 1.4;// Ratio of heat capacities for air
// (P_1*V_1)/T_1 = (P_2*V_2)/T_2   and since   V_1 = V_2
T_2 = (P_2/P_1)*T_1;//[K]

// Since the volume is fixed therefore work done (W) = 0 and from first law we get
// Q = delta_U = n*Cv_0*(T_2 - T_1)
Cv_0 = R/(Y - 1);//[J/mol-K] - Heat capacity at constant volume
Cp_0 = (Y*R)/(Y - 1);//[J/mol-K] - Heat capacity at constant pressure
n = (P_1*V_vessel)/(R*T_1);//[mol] - No. of moles
Q_2 = n*Cv_0*(T_2 - T_1);//[J] - Heat change
Q_2 = Q_2*10^(-3);//[kJ]

delta_S_2 = Cp_0*log(T_2/T_1) - R*log(P_2/P_1);//[J/mol-K]
delta_S_2 = n*delta_S_2*10^(-3);//[kJ/K]

printf(" (b).The final temperature is %f C\n",T_2);
printf("      The amount of heat transfer is %f kJ\n",Q_2);
printf("      The change of entropy is %f kJ/K\n",delta_S_2);
```

Scilab code Exa 4.5 Calculation of final temperature work and heat transfer

```
clear;
clc;

//Example - 4.5
//Page number - 153
printf(" Example - 4.5 and Page number - 153\n\n");

// Given
m = 1000;//[g] - Mass of fluid
P_1 = 20;//[bar] - Initial pressure
P_1 = P_1*10^(5);//[Pa]
P_2 = 2;//[bar] - Final pressure
P_2 = P_2*10^(5);//Pa
T_1 = 250 + 273.15;//[K] - Initial temperature
n = 1.25;
R = 8.314;//[J/mol*-] - Universal gas constant
Y = 1.4;// Index of expansion
Cv_0 = R/(Y- 1);//[J/mol-K]
Cp_0 = R + Cv_0;//[J/mol-K]

//(a)
// For steam at 20 bar and 250 C, from steam table
   as reported in the book
V_1 = 0.11144;//[m^(3)/kg]
U_1 = 2679.6;//[kJ/kg]
S_1 = 6.5453;//[kJ/kg-K]
// P_1*V_1^(n) = P_2*V_2^(n)
V_2 = ((P_1*V_1^(n))/P_2)^(1/n);//[m^(3)/kg]

// At 2 bar under saturated conditions ,from steam
   table as reported in the book
```

```
V_liq = 0.001061;//[m^(3)/kg]
V_vap = 0.8857;//[m^(3)/kg]
x = (V_2 - V_liq)/(V_vap - V_liq);// Dryness
    fraction
T_sat = 120.23;//[C] - The final temperature
U_liq  = 504.49;//[kJ/kg] - Internal energy of
    saturate liquid
U_vap = 2529.5;//[kJ/kg] - Internal energy of
    saturate vapour
// Therefore, internal energy at state 2 is given by
U_2 = U_liq*(1 - x) + x*U_vap;//[kJ/kg]

// Work transfer is given by
W = (P_1*V_1 - P_2*V_2)/(n - 1);//[J/kg]
W = W*10^(-3);//[kJ/kg]
delta_U = U_2 - U_1;//[kJ/kg]

// From first law, q - W = delta_U
q = W + delta_U;//[kJ/kg]

// At final state (2 bar saturated), as reported in
    the book
S_liq = 1.5301;//[kJ/kg-K] - Entropy of saturated
    liquid
S_vap = 7.1271;//[kJ/kg-K] - Entropy of saturated
    vapour
// Therefore, entropy at state 2 is given by
S_2 = S_liq*(1 - x) + x*S_vap;//[kJ/kg-K]
delta_S = S_2 - S_1;//[kJ/kg-K]

printf(" (a).The final temperature is %f C\n",T_sat)
    ;
printf("      The work done is equal to %f kJ/kg\n",W
    );
printf("      The heat change is equal to %f kJ/kg\n"
    ,q);
printf("      The entropy change is equal to %f kJ/kg
    -K\n\n",delta_S);
```

```

//(b)
// P*V^(n) = constant
// Since the gas behaves as ideal we can write
// P_1^(1-n)*T_1^(n) = P_2^(1-n)*T_2^(n)
T_2 = T_1*(P_1/P_2)^((1-n)/n);//[K]

// Molar volume is given by
V_2_1 = (R*T_1)/P_1;//[m^(3)/mol] - At state 1
V_2_2 = (R*T_2)/P_2;//[m^(3)/mol] - At state 2

// Work transfer is given by
w_2 = ((P_1*V_2_1) - (P_2*V_2_2))/(n-1);//[J/mol]
Mol_wt_air = 0.21*32 + 0.79*28;//[g/mol] - Molecular
    weight of air
n_mole = m/Mol_wt_air;
// Total work transfer is given by
W_2 = w_2*n_mole*10^(-3);//[kJ]
// Internal energy change is given by
delta_U = n_mole*Cv_0*(T_2 - T_1)*10^(-3);//[kJ]

// Heat transfer is given by
Q = W_2 + delta_U;//[kJ]

// Entropy change is given by
delta_S_2 = Cp_0*log(T_2/T_1) - R*log(P_2/P_1);//[J/
   mol-K]
delta_S_2 = delta_S_2*n_mole;//[J/mol]

printf(" (b).The final temperature is %f C\n",T_2);
printf("     The work done is equal to %f kJ/kg\n",
   W_2);
printf("     The total heat change is equal to %f kJ
   \n",Q);
printf("     The entropy change is equal to %f kJ/kg
   -K\n\n",delta_S_2);
```

Scilab code Exa 4.6 Calculation of final temperature and work done

```
clear;
clc;

//Example - 4.6
//Page number - 154
printf("Example - 4.6 and Page number - 154\n\n");

//Given
m = 1000;//[g] - Mass of fluid
P_1 = 20;//[bar] - Initial pressure
P_2 = 2;//[bar] - Final ressure
T_1 = 250 + 273.15;//[K] - Initial tempearture
R = 8.314;//[J/mol*K] - Universal gas constant

// (a).
// At 20 bar and 250 C as reported in the book
V_1 = 0.11144;//[m^(3)/kg] - Specific volume
U_1 = 2679.6;//[kJ/kg] - Specific internal energy
S_1 = 6.5453;//[kJ/kg-K] - Specific entropy
S_2 = S_1;// Isentropic expansion

// At 2 bar under saturated conditions
S_liq = 1.5301;//[kJ/kg-K]
S_vap = 7.1271;//[kJ/kg-K]
U_liq = 504.49;//[kJ/kg-K]
U_vap = 2529.5;//[kJ/kg-K]
// Therefore dryness factor can be determined as
x = (S_1 - S_liq)/(S_vap - S_liq);
U_2 = U_liq*(1 - x) + x*U_vap;//[kJ/kg] - Specific
    internal energy at final state
delta_U = U_2 - U_1;//[kJ/kg] - change in internal
    energy
```

```
W = - delta_U;// - Work done

// The final saturated temperature at 2 bar from
   steam table is
T_2 = 120.23;//[C]

printf(" (a).The final temperature is %f C\n",T_2);
printf("      The work done is equal to %f kJ/kg\n\n"
   ,W);

// (b).
Y = 1.4;// Index of expansion for air
Cv_0 = R/(Y-1);//[J/mol*K] - Specific heat capacity
   at constant volume
// Ideal gas under isentropic expansion    P_1^(1-Y)
   *T_1^(Y) =P_2^(1-Y)*T_2^(Y)
T_2_prime = T_1*(P_1/P_2)^((1-Y)/Y);//[K] - Final
   temperature
delta_U_prime = Cv_0*(T_2_prime - T_1);//[J/mol] -
   change in internal energy

// Number of moles is given by
n = m/28.84;//[mol]
delta_U_prime = delta_U_prime*n*10^(-3);//[kJ]
W_prime = - delta_U_prime;// Work done

printf(" (b).The final temperature is %f C\n",
   T_2_prime);
printf("      The work done is equal to %f kJ/kg\n",
   W_prime);
```

Scilab code Exa 4.7 Determination of index of isentropic expansion

```
clear;
clc;
```

```

//Example - 4.7
//Page number - 155
printf("Example - 4.7 and Page number - 155\n\n");

//Given
P_1 = 15;//[bar] - Initial pressure
P_2 = 0.15;//[bar] - Final pressure

// We know that during isentropic expansion
// W = ((P_1*V_1) - (P_2*V_2))/(Y - 1)

// At 15 bar (saturated vapour), from steam table as
    reported in the book
V_1 = 0.13177;//[m^(3)/kg]
U_1 = 2594.5;//[kJ/kg]
S_1 = 6.4448;//[kJ/kg-K]

// Now at state 2 (P_2 = 0.15 bar),from steam table
   as reported in the book
S_2 = S_1;// Isentropic expansion
S_liq = 0.7549;//[kJ/kg-K]
S_vap = 8.0085;//[kJ/kg-K]
U_liq = 225.92;//[kJ/kg]
U_vap = 2448.7;//[kJ/kg]
V_liq = 0.001014;//[m^(3)/kg]
V_vap = 10.02;//[m^(3)/kg]

// Therefore dryness factor can be calculated as
x = (S_1 - S_liq)/(S_vap - S_liq);
U_2 = U_liq*(1 - x) + x*U_vap;//[kJ/kg] - Specific
   internal energy at final state
delta_U = U_2 - U_1;//[kJ/kg] - change in internal
   energy
W = - delta_U;// - Work done

// The specific volume at the final state is
V_2 = V_liq*(1 - x) + x*V_vap;//[m^(3)/kg]
```

```

// From work done under adiabatic conditions we get
// W = ((P_1*V_1) - (P_2*V_2))/(Y - 1)
Y = (((P_1*V_1) - (P_2*V_2))/W) + 1;

printf(" The index of expansion is given by Y = %f\n
   ",Y);
```

Scilab code Exa 4.8 Determination of entropy production

```
clear;
clc;

//Example - 4.8
//Page number - 157
printf("Example - 4.8 and Page number - 157\n\n");

//Given
P_1 = 40;//[bar] - Initial pressure
T_1 = 500;//[C] - Initial temperature
Vel_1 = 140;//[m/s] - Initial velocity
T_2 = 100;//[C] - Final temperature
Vel_2 = 80;//[m/s] - Final velocity
W = 746.0;//[kJ/kg] - Work output

// (a).
// From steam table as reported in the book
H_1 = 3445.3;//[kJ/kg]
H_2 = 2676.1;//[kJ/kg]
S_1 = 7.0901;//[kJ/kh-K]
S_2 = 7.3549;//[kJ/kg-K]

// The temperature at which heat exchange take place
    is given by
T_b =(T_1 + T_2)/2 + 273.15;//[K]
```

```

// From first law in a control volume
// q − W = delta_H + (delta_V ^(2))/2 , therefore
q = W*10^(3) + (H_2 - H_1)*10^(3) + (Vel_2^(2) -
    Vel_1^(2))/2;//[J/kg]
q = q*10^(-3);//[kJ/kg]

S_gen = (S_2 - S_1) - (q/T_b);//[kJ/kg−K]

printf(" (a).The specific entropy production within
    turbine is %f kJ/kg−K\n",S_gen);

//(b)
// If control volume is too large to include the
    turbine and the environment then T_b becomes
    equal to 289 K. In this case
T_b_prime = 298;//[K]

// The entropy change of the sysytem is given by
//delta_S = q/T_b + S_gen
S_gen = (S_2 - S_1) - (q/T_b_prime);//[kJ/kg−K]

printf(" (b).The specific entropy production within
    turbine is %f kJ/kg−K",S_gen);

// In the first part only irreversibilities within
    the turbine are evaluated
// whereas in part (2) irreversible heat transfer
    between the turbine cover and environment  are
    also included.
```

Scilab code Exa 4.9 Determination of work required and exit temperature

```
clear;
clc;
```

```

//Example - 4.9
//Page number - 160
printf("Example - 4.9 and Page number - 160\n\n");

//Given
P_1 = 1;//[MPa] - Initial pressure
T_1 = 200 + 273.15;//[K] - Initial temperature
P_2 = 8;//[MPa] - Final pressure
Y = 1.4;// Index of expansion of gas
R = 8.314;//[J/mol-K] - Universal gas constant

//(1)
// The exit temperature for ideal gas under
   isentropic conditions is given by
T_2 = T_1*((P_2/P_1)^((Y-1)/Y));//[K] - Exit
   temperature
Cp_0 = Y*R/(Y-1);//[J/mol-K] - Specific heat
   capacity at constant pressure
// For isentropic conditions the enthalpy change for
    ideal gas is given by
delta_H_s = Cp_0*(T_2 - T_1);//[J/mol]
// Therefore work is given by
W = - delta_H_s;//[J/mol]

printf(" (1).The exit temperature of steam is %f K\n
   ",T_2);
printf("      The required work is %f J/mol\n\n",W);

//(2)
eff = 0.8;// Adiabatic efficiency
// delta_H_s/delta_H_a = 0.8
delta_H_a = delta_H_s/eff;//[J/mol] - Actual
   enthalpy change
W_a = - delta_H_a;//[J/mol]

// The ideal gas enthalpy is a function only of
   temperature,therefore actual exit temperature
```

```
    T_2a is given by
// delta_H_a = Cp_0*(T_2a - T_1)
T_2a = (delta_H_a/Cp_0) + T_1;

printf(" (2).The exit temperature of steam is %f K\n
    ",T_2a);
printf("     The required work is %f J/mol\n\n",W_a)
    ;
```

Scilab code Exa 4.10 Determination of work required and exit temperature

```
clear;
clc;

//Example - 4.10
//Page number - 161
printf("Example - 4.10 and Page number - 161\n\n");

//Given
P_1 = 1;//[MPa] - Initial pressure
T_1 = 200 + 273.15;//[K] - Initial temperature
P_2 = 8;//[MPa] - Final pressure
Y = 1.4;// Index of expansion of gas
R = 1.987;//[cal/mol*K] - Universal gas constant
// Cp_0 = 7.7 + 0.04594*10^(-2)*T + 0.2521*10^(-5)*T
    ^(2) - 0.8587*10^(-9)*T^(3), here T is in K and
    Cp_0 is in  cal/mol-K
a = 7.7;
b = 0.04594*10^(-2);
c = 0.2521*10^(-5);
d = - 0.8587*10^(-9);

// delta_S = integral((Cp_0/T)*dT) - R*log(P_2/P_1)
    = 0
```

```
// delta_S = integral(((a + b*T + c*T^(2) + d*T^(3))
    /T)*dT) - R*log(P_2/P_1) = 0
// delta_S = a*log(T_2/T_1) + b*(T_2 - T_1) + (c/2)
    *(T_2^(2) - T_1^(2)) + (d/3)*(T_2^(3) - T_1^(3))
    - R*log(P_2/P_1) = 0
// Solving for T_2 in the above equation we get
deff('[y]=f(T_2)','y=a*log(T_2/T_1)+b*(T_2-T_1)+(c
    /2)*(T_2^(2)-T_1^(2))+(d/3)*(T_2^(3)-T_1^(3))-R*
    log(P_2/P_1)');
T_2 = fsolve(100,f);

// Now let us calculate the enthalpy change under
    these conditions
delta_H_s = integrate('7.7+0.04594*10^(-2)*T
    +0.2521*10^(-5)*T^(2)-0.8587*10^(-9)*T^(3)','T',
    T_1,T_2);//[cal/mol]
delta_H_s = delta_H_s*4.184;//[J/mol]
// Therefore isentropic work done is
W = - delta_H_s;

printf(" (1).The exit temperature of steam is %f K\n
    ",T_2);
printf("     The required work is %f J/mol\n\n",W);

//(2)
eff = 0.8;
delta_H_a = delta_H_s/eff;//[J/mol] - Actual
    enthalpy change
// Therefore actual work done is given by
W_a = - delta_H_a;//[J/mol]

// Now we have to determine the exit temperature
    under actual conditions
// delta_H_a = integral(Cp_0*dT) from limit T_1 =
    473.15 K to T_2
// On putting the values and simplifying we get
// 7.7*T_2 + 0.02297*10^(-2)*T_2^(2) + 0.084*10^(-5)
    *T_2^(3) - 0.214675*10^(-9)*T_2^(4) - 6907.106 =
    0

deff('[y]=f1(T_2_prime)','y=a*(T_2_prime-T_1)+(b/2)
    *(T_2_prime^(2)-T_1^(2))+(c/3)*(T_2_prime^(3)-T_1
    ^(3))+(d/4)*(T_2_prime^(4)-T_1^(4))-(delta_H_a
    /4.184)');
T_2_prime = fsolve(100,f1);

printf(" (2).The exit temperature of steam is %f K\n
    ",T_2_prime);
printf("      The required work is %f J/mol\n\n",W_a)
    ;
```

Scilab code Exa 4.11 Determination of work required and exit temperature

```
clear;
clc;

//Example - 4.11
//Page number - 162
printf("Example - 4.11 and Page number - 162\n\n");

//Given
P_1 = 1;//[MPa] - Initial pressure
T_1 = 200 + 273.15;//[K] - Initial temperature
P_2 = 8;//[MPa] - Final pressure
Y = 1.4;// Index of expansion of gas

// At state 1 (1 MPa and 200 C) from steam table as
    reported in the book
H_1 = 2827.9;//[kJ/kg]
S_1 = 6.694;//[kJ/kg]
// At state 2 (8 MPa)
S_2 = S_1;// Isentropic process
```

```
// From steam table at 8 MPa and 450 C
S_21 = 6.5551;//[kJ/kg-K]
// From steam table at 8 MPa and 500 C
S_22 = 6.7240;//[kJ/kg-K]
// Therefore temperature at which entropy of steam
   is 6.694 kJ/kg-K is given by
T_2 = 450 + (500-450)/(S_22-S_21)*(S_2-S_21);//[C]
T_2 = T_2 + 273.15;//[K]

// Enthalpy of steam at 8 MPa and 450 C from steam
   table as reported in the book
H_21 = 3272.0;//[kJ/kg]
// And at 8 MPA and 500 C
H_22 = 3398.3;//[kJ/kg]
// Therefore enthalpy of steam at 8 MPa and T_2
H_2 = H_21 + ((H_22-H_21)/(500-450))*((T_2-273.15) -
    450);
// Work done is given by
// W = - delta_H_s
W = - (H_2 - H_1);//[J/g]
W = W*18.015;//[J/mol]
delta_H_s = - W;

printf(" (1).The exit temperature of steam is %f K\n
   ",T_2);
printf("      The required work is %f J/mol\n\n",W);

//(2)
eff = 0.8;// Adiabatic efficiency
// delta_H_s/delta_H_a = 0.8
delta_H_a = delta_H_s/eff;//[J/mol] - Actual
   enthalpy change
// Therefore actual work done
W_a = - delta_H_a;//[J/mol]
// Enthalpy at actual exit conditions is
H_2_a = H_1 + delta_H_a/18.015;//[kJ/kg]

// Enthalpy of steam at 8 MPa and 500 C from steam
```

```
   table as reported in the book
H_21_a = 3398.3;//[kJ/kg]
// And at 8 MPA and 550 C
H_22_a = 3521.0;//[kJ/kg]
// Therefore temperature at H_22_a is given by
T_2_a = 500 + ((550-500)*(H_2_a - H_21_a))/(H_22_a -
    H_21_a);//[C]
T_2_a = T_2_a + 273.15;//[K]

printf(" (2).The exit temperature of steam is %f K\n
   ",T_2_a);
printf("     The required work is %f J/mol\n\n",W_a)
   ;
```

Scilab code Exa 4.12 Determination of work required and discharge temperature

```
clear;
clc;

//Example - 4.12
//Page number - 163
printf("Example - 4.12 and Page number - 163\n\n");

//Given
P_1 = 140;//[kPa] - Initial pressure
T_1 = 20 + 273.15;//[K] - Initial temperature
P_2 = 560;//[kPa] - Final pressure
eff = 0.75;// Compressor efficiency
R = 1.987;//[cal/mol*K] - Universal gas constant
// Cp_0 = 4.750 + 1.200*10^(-2)*T + 0.3030*10^(-5)*T
   ^(2) - 2.630*10^(-9)*T^(3), here T is in K and
   Cp_0 is in  cal/mol-K
a = 7.7;
b = 0.04594*10^(-2);
```

```
c = 0.2521*10^(-5);
d = - 0.8587*10^(-9);

// At 20 C,as reported in the book
Cp_0 = 8.46;//[cal/mol-K] - Specific heat capacity
    at constant pressure
Cv_0 = Cp_0 - R;//[cal/mol-K] - Specific heat
    capacity at constant volume
Y = Cp_0/Cv_0;// Index of expansion

// Assuming 100 % efficiency ,for reversible and
    adiabatic process the final temperature is given
    by
// P*V^(Y) = constant or, P*((R*T)/P)^(Y) = constant
T_2 = ((P_1/P_2)^((1-Y)/Y))*T_1;//[K]

// Since at final temperature the value of heat
    capacity ratio would have changed
// So let us determine Y at mean temperature and
    then calculating final temperature
T_mean = (T_1 + T_2)/2;//[K]

// At T_mean,as reported in the book
Cp_0_new = 9.153;//[cal/mol-K]
Cv_0_new = Cp_0_new - R;//[cal/mol-K]
Y_new = Cp_0_new/Cv_0_new;
T_2_new = T_1*((P_1/P_2)^((1-Y_new)/Y_new));//[K]

// The enthalpy change is given by
delta_H = integrate('4.750+1.200*10^(-2)*T
    +0.3030*10^(-5)*T^(2)-2.630*10^(-9)*T^(3)','T',
    T_1,T_2_new);//[cal/mol]

//For adiabatic process
W = - delta_H;//[cal/mol]
// Now actual work done on the system is given by
W_a = W/eff;//[cal/mol]

```

```
// Since the actual process is adiabatic the work
    done is change in negative of enthalpy
// Therefore actual change in enthalpy is - W_a, or
// - W_a = 4.750*(T_2-T_1) + (1.2*10^(-2)/2)*(T_2
    ^(2)-T_1^(2)) + (0.3030*10^(-5)/3)*(T_2^(3)-T_1
    ^(3)) - (2.63*10^(-9)/4)*(T_2^*(4)-T_1^(4));
// Solving for T_2 in the above equation
deff('[y]=f1(T_2_prime)','y=4.750*(T_2_prime-T_1)
    +((1.2*10^(-2))/2)*(T_2_prime^(2)-T_1^(2))
    +((0.3030*10^(-5))/3)*(T_2_prime^(3)-T_1^(3))
    -((2.63*10^(-9))/4)*(T_2_prime^(4)-T_1^(4))+W_a')
    ;
T_2_prime=fsolve(100,f1);

printf(" The required work is %f cal/mol\n",W_a);
printf(" The discharge temperature of methane is %f
    K\n",T_2_prime);
```

Scilab code Exa 4.13 Dtermination of power output entropy and exit temperature

```
clear;
clc;

//Example - 4.13
//Page number - 164
printf("Example - 4.13 and Page number - 164\n\n");

//Given
P_1 = 10;//[bar] - Initial pressure
T_1 = 500 + 273.15;//[K] - Initial temperature
P_2 = 2;//[psia] - Final pressure
P_2 = P_2/14.5;//[bar]
P_2 = P_2*10^(2);//[kPa]
m = 1.8;//[kg/s] - Mass flux
```

```
eff = 0.8;// Efficiency

// At state 1, from steam table
H_1 = 3478.5;//[kJ/kg]
S_1 = 7.7622;//[kJ/kg-K]
S_2 = S_1;// Adiabatic process

// From saturated steam table at 10 kPa
S_liq_1 = 0.6493;//[kJ/kg-K]
S_vap_1 = 8.1502;//[kJ/kg-K]
// From saturated steam table at 15 kPa
S_liq_2 = 0.7549;//[kJ/kg-K]
S_vap_2 = 8.0085;//[kJ/kg-K]
// Threfore at P_2
S_liq = S_liq_1 + ((S_liq_2-S_liq_1)/(15-10))*(P_2
    -10);
S_vap = S_vap_1 + ((S_vap_2-S_vap_1)/(15-10))*(P_2
    -10);

// The dryness fraction at exit state is
x_2 = (S_1-S_liq)/(S_vap-S_liq);
// The enthalpy at exit to be determined. At 10 kPa
H_liq_1 = 191.83;//[kJ/kg]
H_vap_1 = 2584.7;//[kJ/kg]
// At 15 kPa
H_liq_2 = 225.94;//[kJ/kg]
H_vap_2 = 2599.1;//[kJ/kg]
// Therfore at P_2
H_liq = H_liq_1 + ((H_liq_2-H_liq_1)/(15-10))*(P_2
    -10);
H_vap = H_vap_1 + ((H_vap_2-H_vap_1)/(15-10))*(P_2
    -10);

// Enthalpy at state 2
H_2_s = H_liq*(1-x_2) + x_2*H_vap;//[kJ/kg]
W = m*(H_1 - H_2_s);//[kW]

printf(" (1).The power output is %f kW\n\n",W);
```

```

//(2)
// If the process is 80 % efficient the enthalpy
   change is
// H_1 - H_2_a = eff*(H_1 - H_2_s)
H_2_a = H_1 - (0.8*(H_1 - H_2_s));

// Now under these conditions temperature and
   entropy have to be determined. From superheated
   steam tables,as reported in the book
// At 10 kPa and 100 C
H_2_1 = 2687.5;//[kJ/kg]
S_2_1 = 8.4479;//[kJ/kg-k]
// At 10 kPa and 150 C
H_2_2 = 2783.0;//[kJ/kg]
S_2_2 = 8.6882;//[kJ/kg-k]
// At 50 kPa and 100 C
H_3_1 = 2682.5;//[kJ/kg]
S_3_1 = 7.6947;//[kJ/kg-k]
// At 50 kPa and 150 C
H_4_1 = 2780.1;//[kJ/kg]
S_4_1 = 7.9401;//[kJ/kg-k]
// Therefore at P_2 and 100 C
H_prime_1 = H_2_1 + ((H_3_1-H_2_1)/(50-10))*(P_2-10)
   ;//[kJ/kg]
S_prime_1 = S_2_1 + ((S_3_1-S_2_1)/(50-10))*(P_2-10)
   ;//[kJ/kg-K]
// Therefore at P_2 and 150 C
H_prime_2 = H_2_2 + ((H_4_1-H_2_2)/(50-10))*(P_2-10)
   ;//[kJ/kg]
S_prime_2 = S_2_2 + ((S_4_1-S_2_2)/(50-10))*(P_2-10)
   ;//[kJ/kg-K]

// Enthalpy at exit is H_2_a. So at this condition
   temperature can be nom be determined
T_exit = ((H_2_a - H_prime_1)/(H_prime_2 - H_prime_1
   ))/(150-100) + 100;//[C]
// The entropy at exit is
```

```
S_exit =  ((H_2_a - H_prime_1)/(H_prime_2 -
   H_prime_1))/(S_prime_2 - S_prime_1) + S_prime_1;
   //[kJ/kg-K]

printf(" (2).The entropy at exit is %f kJ/kg-K\n",
   S_exit);
printf("     The temperature of the exit state is %f
    C\n\n",T_exit);

printf("     The irreversibility is advatageous
   because the exit steam is superheated and
   therefore,\n");
printf("     the blades of the turbine are not
   eroded by water droplets which get formed when
   the process is isentropic");
```

Scilab code Exa 4.14 Calculation of work output per unit mass

```
clear;
clc;

//Example - 4.14
//Page number - 166
printf("Example - 4.14 and Page number - 166\n\n");

//Given
P_1 = 6;//[MPa] - Initial pressure
T_1 = 500 + 273.15;//[K] - Initial temperature
P_2 = 10;//[kPa] - Final pressure
out_qlty = 0.9;// Output quality

// At 6 MPa and 500 C, from steam table as reported
   in the book
H_1 = 3422.2;//[kJ/kg]
S_1 = 6.8803;//[kJ/kg-K]
```

```
S_2 = S_1;// Adiabatic reversible conditions
// At 10 kPa saturated
H_liq = 191.83;//[kJ/kg]
H_vap = 2584.7;//[kJ/kg]
S_liq = 0.6493;//[kJ/kg-K]
S_vap = 8.1502;//[kJ/kg-K]

// The dryness fraction is given by
x = (S_1-S_liq)/(S_vap-S_liq);

// Now the exit enthalpy is given by
H_2 = H_liq*(1-x) + x*H_vap;//[kJ/kg]
W = - (H_2 - H_1);//[kJ/kg] - Under isentropic
    conditions

// We know that, delta_S = q/T_b + S_gen
// Since delta_S = 0, therefore under isentropic
    conditions
S_gen = 0;//[kJ/kg-K]

// Now for output quality 0.9
H_2_a = H_liq*(1-out_qlty) + out_qlty*H_vap;//[kJ/kg
    ]
S_2_a = S_liq*(1-out_qlty) + out_qlty*S_vap;//[kJ/kg
    ]
W_a = - (H_2_a - H_1);//[kJ/kg]
delta_S_a = S_2_a - S_1;//[kJ/kg-k]
// Again, delta_S = q/T_b + S_gen
// Since q = 0, therefore under isentropic
    conditions
S_gen_a = delta_S_a;//[kJ/kg-K
// Now efficiency is given by eff = delta_H_a/
    delta_H_s
eff = W_a/W;

printf(" For output quality = 0.9\n");
printf(" The work output per unit mass is %f kJ/kg\n
   ",W_a);
```

```
printf(" The entropy generation is given by S_gen =
    %f kJ/kg-K\n",S_gen_a);
printf(" The efficiency with respect to reversible
    adiabatic case is given by eff = %f\n\n",eff);

// Now for output quality 1
out_qlty_1 = 1;
H_2_a_1 = H_liq*(1-out_qlty_1) + out_qlty_1*H_vap;//
    [kJ/kg]
S_2_a_1 = S_liq*(1-out_qlty_1) + out_qlty_1*S_vap;//
    [kJ/kg]
W_a_1 = - (H_2_a_1 - H_1);//[kJ/kg]
delta_S_a_1 = S_2_a_1 - S_1;//[kJ/kg-k]
// Again,  delta_S = q/T_b + S_gen
// Since q = 0, therefore under isentropic
    conditions
S_gen_a_1 = delta_S_a_1;//[kJ/kg-K
// Now efficiency is given by eff = delta_H_a/
    delta_H_s
eff_1 = W_a_1/W;

printf(" For output quality = 1.0\n");
printf(" The work output per unit mass is %f kJ/kg\n
    ",W_a_1);
printf(" The entropy generation is given by S_gen =
    %f kJ/kg-K\n",S_gen_a_1);
printf(" The efficiency with respect to reversible
    adiabatic case is given by eff = %f\n",eff_1);
```

Scilab code Exa 4.15 Estimation of final velocity

```
clear;
clc;

//Example - 4.15
```

```
//Page number - 168
printf("Example - 4.15 and Page number - 168\n\n");

//Given
P_1 = 3;//[bar] - Initial pressure
T_1 = 150 + 273.15;//[K] - Initial temperature
Vel_1 = 90;//[m/s] - Initial velocity
P_2 = 1;//[bar] - Final pressure
eff = 0.95;// Adiabatic effciciency of the nozzle
R = 8.314;//[J/mol*-] - Universal gas constant

// At 3 bar and 150 C, from steam table
S_1 = 7.0778;//[kJ/kg-K]
H_1 = 2761.0;//[kJ/kg]
S_2 = S_1;//

// At 1 bar saturated
S_liq = 1.3026;//[kJ/kg-K]
S_vap = 7.3594;//[kJ/kg-K]
H_liq = 417.46;//[kJ/kg]
H_vap = 2675.5;//[kJ/kg]
// The dryness factor of exit steam can be
    determined as
x = (S_1-S_liq)/(S_vap-S_liq);
// Enthalpy of exit steam is given by
H_2 = H_liq*(1-x) + x*H_vap;//[kJ/kg]
delta_H_s = H_2 - H_1;//[kJ/kg] - Enthalpy change
delta_H_a = eff*delta_H_s;//[kJ/kg]

// Assuming no heat exchange with surroundings and
    since no work is done
// delta_H + (delta_V^(2))/2 = 0
delta_Vel_square = 2*(-delta_H_a)*1000;//[m^(2)/s
    ^(2)]
Vel_2 = (delta_Vel_square + Vel_1^(2))^(1/2);//[m/s]

printf(" (1).The final velocity (when fluid is steam
    ) is %f m/s\n\n",Vel_2);
```

```

// (2)
Y = 1.4;// Index of expansion
Cp_0 = (Y*R)/(Y-1);//[J/mol-K] - Specific heat
   capacity at constant pressure
// The final temperature has to be determined such
   that entropy change is zero. Under isentropic
   conditions
// P_1^(1-Y)*T_1^(Y) = P_2^(1-Y)*T_2^(Y)
T_2 = T_1*(P_1/P_2)^((1-Y)/Y);//[K]
delta_H_s_2 = Cp_0*(T_2 - T_1);//[J/mol]
delta_H_a_2 = eff*delta_H_s_2;//[J/mol]
delta_H_a_2 = (delta_H_a_2*1000)/28.84;//[J/kg]

delta_Vel_square_2 = 2*(-delta_H_a_2);//[m^(2)/s^(2)
   ]
Vel_2_2 = (delta_Vel_square_2 + Vel_1^(2))^(1/2);//[
   m/s]

printf(" (2).The final velocity (when fluid is air
   which behaves as an ideal gas) is %f m/s\n\n",
   Vel_2_2);
```

Scilab code Exa 4.16 Calculation of final velocity and increase in entropy

```
clear;
clc;

//Example - 4.16
//Page number - 169
printf("Example - 4.16 and Page number - 169\n\n");

//Given
P_1 = 300;//[kPa] - Initial pressure
T_1 = 450;//[K] - Initial temperature
```

```
Vel_1 = 90;//[m/s] - Initial velocity
P_2 = 180;//[kPa] - Final pressure
eff = 0.95;// Adiabatic effciciency of the nozzle
R = 8.314;//[J/mol*-] - Universal gas constant
Cp = 5.19;//[kJ/kg-K] - Specific heat capacity at
   constant pressure

//(a)
// Exit velocity is highest when drop in enthalpy is
    highest or when isentropic conditions are
   maintained

Mol_wt_He = 4;//[g/mol] - Molecular weight of helium
R_He = R/Mol_wt_He;// 'R' for helium
Y = Cp/(Cp - R_He);

// Now temperature at exit to be determined
T_2s = T_1*(P_1/P_2)^((1-Y)/Y);//[K]
delta_H_2s = Cp*(T_2s - T_1);//[kJ/kg]

// Since no work is done and heat exchange is zero,
   from first law we get
// delta_H + delta_Vel^(2)/2 = 0
delta_Vel_square = 2*(-delta_H_2s)*1000;//[m^(2)/s
   ^(2)]
Vel_2 = (delta_Vel_square)^(1/2);//[m/s] - ( as
   Vel_1 << Vel_2)

printf(" (a).The maximum exit velocity is %f m/s\n\n
   ",Vel_2);

//(b)
T_2a = 373;//[K] - Measured temperature of helium
delta_H_a = Cp*(T_2a - T_1);//[kJ/kg]
delta_Vel_square_a = 2*(-delta_H_a)*1000;//[m^(2)/s
   ^(2)]
Vel_2a = (delta_Vel_square_a)^(1/2);//[m/s] - ( as
   Vel_1 << Vel_2a)
```

```

printf(" (b).The actual exit velocity is %f m/s\n\n"
   ,Vel_2a);

//(c)
delta_S_sys = Cp*log(T_2a/T_1) - R_He*log(P_2/P_1);
// we know that delta_S_sys = q/T_b + S_gen and
   since q = 0, therfore
S_gen = delta_S_sys;//[kJ/kg-K]

printf(" (c).The increasse in entropy per unit mass
   is %f kJ/kg-K",S_gen);

// The source of irreversibility is friction in the
   nozzle.
```

Scilab code Exa 4.17 Calculation of work done and heat transfer

```
clear;
clc;

//Example - 4.17
//Page number - 170
printf("Example - 4.17 and Page number - 170\n\n");

//Given
P_1 = 1;//[bar] - Initial pressure
T_1 = 150 + 273.15;//[K] - Initial temperature
V_2 = 0.28;//[m^(3)/kg] - Final specific volume
T_2 = T_1;//[K] - Isothermal process
R = 8.314;//[J/mol-K] - Universal gas constant

// At 1 bar and 150 C, from steam table
S_1 = 7.6134;//[kJ/kg-K]
H_1 = 2776.4;//[kJ/kg]
```

```

// At 150 C saturated
V_liq = 0.001091;//[m^(3)/kg]
V_vap = 0.3928;//[m^(3)/kg]
H_liq = 632.2;//[kJ/kg]
H_vap = 2746.5;//[kJ/kg]
S_liq = 1.8418;//[kJ/kg-K]
S_vap = 6.8379;//[kJ/kg-K]

// The dryness factor of exit steam can be
   determined as
x = (V_2 - V_liq)/(V_vap - V_liq);
S_2 = S_liq*(1-x) + x*S_vap;//[kJ/kg-K] -Entropy
H_2 = H_liq*(1-x) + x*H_vap;//[kJ/kg] -Enthalpy
delta_H = H_2 - H_1;//[kJ/kg] - Enthalpy change
delta_S = S_2 - S_1;//[kJ/kg]

// Since the compression is reversible
q = T_1*delta_S;//[kJ/kg] - Heat transfer
// From first law  q - W = delta_H
W = q - delta_H;//[kJ/kg]

printf(" (1).The amount of heat transfer (when fluid
    is steam) is %f kJ/kg\n",q)
printf("     The amount of work transfer (when fluid
    is steam) is %f kJ/kg\n\n",W)

//(2)
V_2 = V_2*(28.84/1000);//[m^(3)/mol] - Molar volume
   at exit
// Pressure at exit is given by
P_2 = ((R*T_2)/V_2);//[N/m^(2)]
P_2 = P_2*10^(-5);//[bar]

// Entropy change is given by, delta_S_2 = Cp*log(
   T_2/T_1) - R*log(P_2/P_1), but since T_1 = T_2,
   therfore
delta_S_2 = - R*log(P_2/P_1);//[J/mol-K]
```

```

q_2 = T_1*delta_S_2;//[J/mol] - Heat change
q_2 = q_2/28.84;//[kJ/kg]

// Enthalpy change is given by,  delta_H_2 = Cp*(T_2
    - T_1)  = 0 (as T_1 = T_2)
delta_H_2 = 0;//[kJ/kg]

// From first law  q - W = delta_H , therefore
W_2 = q_2 - delta_H_2;//[kJ/kg]

printf(" (2).The amount of heat transfer (when fluid
    is ideal gas) is %f kJ/kg\n",q_2)
printf("     The amount of work transfer (when fluid
    is ideal gas) is %f kJ/kg\n",W_2)
```

Scilab code Exa 4.18 Calculation of air velocity and change in entropy

```
clear;
clc;

//Example - 4.18
//Page number - 171
printf("Example - 4.18 and Page number - 171\n\n");

//Given
P_1 = 7*10^(5);//[Pa] - Initial pressure
T_1 = 95 + 273.15;//[K] - Initial temperature
P_2 = 3.5*10^(5);//[Pa] - Final pressure
dia = 15*10^(-2);//[m] - Diameter of pipe
m = 2;//[kg/s] - Mass flow rate
R = 8.314;//[J/mol-K] - Universal gas constant
Y = 1.4;// Index of expansion
Cp_0 = (R*Y)/(Y-1);//[J/mol-K] - Specific heat
    capacity at constant pressure
```

```
Cp_0 = (Cp_0/28.84)*1000;//[J/kg-K]
rho_1 = 6.6;//[kg/m^(3)] - Density

// velocity before throttling is to be determined m
   = rho*Vol*Area
V_1 = (R*T_1)/P_1;//[m^(3)/mol] - Specific volume
V_1 = (V_1/28.84)*1000;//[m^(3)/kg]
Vel_1 = m/(rho_1*3.14*(dia/2)^(2));//[m/s] -
    Velocity before throttling

// Let the temperature after throttling be T_2, then
// V_2 = (((R*T_2)/P_2)/28.84)*1000
// Vel_2 = m/(rho_2*Area) = (m*V_2)/(3.14*(dia/2)
    ^(2))
// From first law, since q = W = 0, we get
// delta_H + (delta_V^(2))/2 = 0
// Cp_0*(T_2 - T_1) + ((Vel_2)^(2) - (Vel_1)^(2))/2
   = 0
//Cp_0*(T_2 - T_1) +  (((m*((((R*T_2)/P_2)/28.84)
    *1000))/(3.14*(dia/2)^(2)))^(2) - (Vel_1)^(2))/2
   = 0
// Solving the above equation for T_2, we get
deff('[y]=f1(T_2)','y=Cp_0*(T_2 - T_1) +  (((m*((((R
    *T_2)/P_2)/28.84)*1000))/(3.14*(dia/2)^(2)))^(2)
   - (Vel_1)^(2))/2');
T_2 =fsolve(100,f1);//[K] - Temperature after
    throttling
// Therefore velocity of air downstream of
    restriction is given by
Vel_2 = ((m*((((R*T_2)/P_2)/28.84)*1000))/(3.14*(dia
    /2)^(2)));//[m/s]

printf(" The velocity of air downstream of
    restriction is %f m/s\n",Vel_2);

delta_T = (T_2 - T_1);
// Since temperature difference (delta_T) is very
    small, therefore enthalpy change is also very
   small

// Entropy change is given by, delta_S = Cp_0*log(
   T_2/T_1) - R*log(P_2/P_1), but since T_1 and T_2
   are almost equal
delta_S = - R*log(P_2/P_1);//[J/mol-K]

printf(" The change in entropy is %f kJ/mol-k",
   delta_S);
```

Chapter 5

Exergy

Scilab code Exa 5.1 Determination of fraction of the availability loss

```
clear;
clc;

//Example - 5.1
//Page number - 184
printf("Example - 5.1 and Page number - 184\n\n");


//Given
T_1 = 500+273.15;//[C] - Condensation temperature
T_2 = 250+273.15;//[C] - Temperature at which
    vaporization takes place.

T_3 = 25+273.15;//[C] - Ambient atmospheric
    temperature.

Q = 1;//We are taking a fictitious value of Q, its
    value is not given.But we need to initialize it
    wid some value,so we are taking its value as Q=1.

//The exergy content of the vapour at 500 C,
Ex_T_1 = Q*(1-(T_3/T_1));
Ex_T_2 = Q*(1-(T_3/T_2));
//Therefore,loss in exergy is given by
Ex_loss = Ex_T_1 - Ex_T_2;
//Fraction of exergy lost due to irreversible
   process is,
Ex_fraction =(Ex_loss/Ex_T_1);
printf(" The fraction of exergy lost due to
   irreversible process is %f",Ex_fraction);
```

Scilab code Exa 5.2 Determination of availability change and irreversibility

```
clear;
clc;

//Example - 5.2
//Page number - 188
printf("Example - 5.2 and Page number - 188\n\n");

//Given
T_1 = 300;//[K] - Initial temperature.
P_1 = 100;//[kPa] - Initial pressure.
T_2 = 500;//[K] - Final temperature.
T_0 = 300;//[K] - Environment temperature.
P_0 = 1;//[atm] - Environment pressure.
R = 8.314;//[J/mol*K]
//(Cp_0/R)= 3.626
Cp_0 = 3.626*R;//[J/mol-K] - Heat capacity at
   constant pressure


//(1).
//The availability change is given by, (phi_1 -
   phi_2) = U_1 - U_2 + P_0*(V_1 - V_2) - T_0*(S_1 -
   S_2)
//Let us determine the change in internal energy
//For ideal gas the molar internal energy change is
   given by  delta_U = Cv_0*(T_2-T_1)
//For ideal gas Cp_0 - Cv_0 = R, and therefore
Cv_0 = ((Cp_0/R)- 1)*R;//[J/mol-K] - Heat capacity
   at constant volume
delta_U = Cv_0*(T_2-T_1);//[J/mol]
//delta_U = -w (from energy balance). Therefore, U1-
   U2 = -delta_U.
//The entropy change of ideal gas is given by
//delta_S = Cp_0*log(T_2/T_1) - R*log(P_2/P_1), but
   ,(P1*V1/T1) = (P1*V1/T1) and therefore (P2/P1) =
   (T2/T1)
delta_S = Cp_0*log(T_2/T_1) - R*log(T_2/T_1);//[J/
   mol-K]
//The exergy change is given by, (phi_1 - phi_2) =
   U_1 - U_2 + P_0*(V_1 - V_2) - T_0*(S_1 - S_2)
//(V_1 - V_2) = 0, because the tank is rigid and so
   the volume is constant
delta_phi = (-delta_U) - T_0*(-delta_S);//[J/mol]
printf(" (1).The change in exergy is %f J/mol\n\n",
   delta_phi);

//(2).
//Entropy change of the system is given by,
   delta_S_sys = q/T_b + S_gen
//Since the system is adiabatic therefore,
   delta_S_sys = S_gen
S_gen = delta_S;
//Irreversibility is given by
i = T_0*S_gen;//[J/mol]
printf(" (2).The value of irreversibility is %f J/
   mol",i);
//Irreversibility can also be determined using
//i = (W_rev_use - W_use)
```

Scilab code Exa 5.3 Determination of availability change and irreversibility

```
clear;
clc;

//Example - 5.3
//Page number - 190
printf("Example - 5.3 and Page number - 190\n\n")

//Given
P_1 = 15;//[bar] - Initial pressure
P_1 = P_1*10^(5);//[Pa]
T_1 = 300+273.15;//[K] - Initial temperature
T_0 = 298.15;//[K]
T_R = 1200;//[K] - Reservoir temerature.
P_0 = 1;//[bar]
P_0 = P_0*10^(5);//[Pa]
n = 1;//[mol] - No of moles
R = 8.314;//[J/mol*K]
Y = 1.4;// - Ratio of heat capacities.
Cv_0 = R/(Y-1);//[J/mol-K] - Heat capacity at
    constant volume
Cp_0 = Cv_0 + R;//[J/mol-K] - Heat capacity at
    constant pressure

//(1)
//V_2 = 2*V_1 and since pressure is constant,we get
    (V_1/T_1) = (2*V_1/T_1), or, T_2 = 2*T_1.
T_2 = 2*T_1;//[K]
W = P_1*(((R*T_2)/P_1)-((R*T_1)/P_1));//[J/mol] -
    Actual work done
delta_U = Cv_0*(T_2-T_1);//[J/mol] - Change in
    internal energy.
```

```
q = W + delta_U;//[J/mol] - Heat change
//Now the availability change is given by, (phi_1 -
   phi_2) = U_1 - U_2 + P_0*(V_1 - V_2) - T_0*(S_1 -
    S_2) + q*(1-(T_0/T_R))
//delta_S = Cp_0*log(T_2/T_1) - R*log(P_2/P_1), and
   P_1 = P_2, Therefore
delta_S = Cp_0*log(T_2/T_1);;//[J/mol-K] - Entropy
   change.
//Substituting expressions for delta_phi calculation
   . Decrease in availability is given by,
delta_phi = (-delta_U) + P_0*(((R*T_1)/P_1)-((R*T_2)
   /P_1)) - T_0*(-delta_S) + q*(1-(T_0/T_R));//[J/
   mol]
//Actual work done is given by, W = P_1*(V2-V1)
//Work done to displace the atmosphere is given by,
   W = P_0*(V_2-V_1)
//Therefore, W_use = (P_1*(V2-V1) - P_0*(V2-V1))
W_use = (P_1-P_0)*(((R*T_2)/P_1)-((R*T_1)/P_1));//[J
   /mol] - useful work done
W_rev_use = delta_phi;// reversible useful work done
//Irreversibility is given by,
i = W_rev_use - W_use;//[J/mol]
printf(" (a).The ireversibility value is %f J/mol\n\
   n",i);

//The irreversibility can also be calculated using
// i = T_0*S_gen
//S_gen = delta_S - (q/T_R)

//(b)
//V2 = 2*V_1 and therefore T_2 = 2*T_1, as P_2 = P_1
//Actual work done is same as before
//Let work done on stirrer be W_stir. Thus net work
   done by the system is W- W_stir.Fron energy
   balance we get,
W_stir = W + delta_U;
//Initially the exergy is due to that of the system
   at state 1 and stirrer work,'W_stir' and finally
   we have the exergy due to system at state 2,the
   stirrer work  is spent,thus availability is given
    by
delta_phi_b = (-delta_U) + P_0*(((R*T_1)/P_1)-((R*
   T_2)/P_1)) - T_0*(-delta_S) + W_stir;//[J/mol]
W_rev_use_b = delta_phi_b;// reversible useful work
   done
W_use_b = W_use;// useful work done
//Now the irreversibility is given by,
i_b = W_rev_use_b - W_use_b;//[J/mol]
printf(" (b).The ireversibility value is %f J/mol\n\
   n",i_b);

//The irreversibility can also be calculated using
// i_b = T_0*S_gen
//S_gen = delta_S - (q/T_R) and here, q = 0

//(c)
P_2_c = 10;//[bar] - Final pressure, (Given)
P_2_c = P_2_c*10^(5);//[Pa]
//(P_1^(1-Y))*(T_1^(Y)) = (P_2^(1-Y))*(T_2^(Y))
T_2_c = T_1*((P_1/P_2_c)^((1-Y)/Y));//[K]
//Work done is given by, W = -delta_U = -Cv_0*(T_2_c
    - T_1)
W_c = -Cv_0*(T_2_c - T_1);//[J/mol]
//The final molar volume is calculated using P_1*V_1
   ^(Y) = P_2*V_2^(Y)
//V_2 = V_1*((P_1/P_2_c)^(1/Y))
V_1 = (R*T_1)/P_1;//[cubic metre/mol] - Initial
   molar volume
V_2 = V_1*((P_1/P_2_c)^(1/Y));//[cubic metre/mol] -
   Final molar volume
//Now let us determine the work done to displace the
    atmosphere,
W_atm_c = P_0*(V_2 - V_1);//[J/mol] - work done to
   displace the atmosphere
//Thus useful work is given by,
W_use_c = W - W_atm_c;//[J/mol] - useful work done
```

```
//Here delta_S = 0,for reversible adiabatic process.
    Therefore,
W_rev_use_c = W_use_c;
//Now finally the irreversibility is given by,
i_c = W_rev_use_c - W_use_c;//[J/mol]
printf(" (c).The ireversibility value is %f J/mol\n\
    n",i_c);

//(d)
//Here temperature is constant,but V_2 = 2*V_1,
    therefore P_2 = P_1/2
V_2_d = 2*V_1;
P_2_d = P_1/2;
//Under isothermal conditions work done is
W_d = R*T_1*log(V_2_d/V_1);//[J/mol]
//Work done to displace the atmosphere is given by,
W_atm_d = P_0*(V_2_d - V_1);//[J/mol] - work done to
     displace the atmosphere
//Thus useful work is given by,
W_use_d = W_d - W_atm_d;//[J/mol] - useful work done
delta_U_d = 0;//isothermal conditions
q_d = W_d;// since, delta_U_d = 0
//delta_S_d = Cp_0*log(T_2/T_1) - R*log(P_2/P_1),
    and T_1 = T_2, Therefore
delta_S_d = -R*log(P_2_d/P_1);//[J/mol-K] - Entropy
    change
//The reversible useful work is given by,
W_rev_use_d = P_0*(V_1 - V_2_d) - T_0*(-delta_S_d) +
     q_d*(1-(T_0/T_R));//[J/mol] - Reversible useful
    work done.
//The irreversibility is given by,
i_d = W_rev_use_d - W_use_d;//[J/mol]
printf(" (d).The ireversibility value is %f J/mol\n\
    n",i_d);

//(e)
P_2_e = 10;//[bar] - Final pressure, (Given)
P_2_e = P_2_e*10^(5);//[Pa]
```

```
//During the expansion of an ideal gas in into
   vacuum the temperature of the gas remains the
   same,
T_2_e = T_1;// Final temperature
//Since boundary of the system is fixed so no net
   work is done, W = 0 and thus
W_use_e = 0;//[J/mol] - Useful work done
//Here, delta_U = 0,as temperature is same and
//(V_1-V_2) = 0,as for overall system there is no
   change in volume
delta_S_e = - R*log(P_2_e/P_1);//[J/mol-K] - Entropy
    change
//The reversible useful work is given by,
W_rev_use_e =  - T_0*(-delta_S_e);//[J/mol] -
   Reversible useful work done.
//The irreversibility is given by,
i_e = W_rev_use_e - W_use_e;//[J/mol]
printf(" (e).The ireversibility value is %f J/mol\n\
   n",i_e);
```

Scilab code Exa 5.4 Determination of useful work and irreversibility

```
clear;
clc;

//Example - 5.4
//Page number - 194
printf("Example - 5.4 and Page number - 194\n\n")


//Given
T_1 = 150+273.15;//[K] - Initial temperature.
m = 4.6;//[kg] - mass of water
P_1 = 1;//[MPa] - Initial pressure
Q = 11000;//[kJ] - Heat transferred to the system.
```

```
T_R = 600+273.15;//[K] - Temperature of the
    reservior.
T_0 = 298;//[K] - Temperature of the environment
P_0 = 100;//[kPa] - Pressure of the environment

//(1)
//The entropy change of an isothermal system
    undergoing an internally reversible process is
    given by,
delta_S_t = (Q/T_1);//[kJ] - Entropy change
delta_S = delta_S_t/m;//[kJ/kg-K] -

//At 150 C,it has been reported in the book that,
    P_sat - 0.4758 kPa, V_liq = 0.001091 m^(3)/kg,
    U_liq = 631.68 kJ/kg, S_liq = 1.8418 kJ/kg-K,
    S_vap = 6.8379 kJ/kg-K
V_1 = 0.001091;//[m^(3)/kg] - initial specific
    volume
U_1 = 631.68;//[kJ/kg] - initial specific internal
    energy
S_1 = 1.8418;//[kJ/kg-K] - initial entropy
//The initial state of the water is a compressed
    liquid state,and S_1 is therefore equal to the
    entropy of the saturated liquid of the saturated
    liquid at the  same temperature.
S_2 = S_1 + delta_S;//[kJ/kg-K] - Final entropy

//At the final state the temperature is 150 C and S
    = 7.499 kJ/kg-K which is more than S_vap
    therefore it is superheated steam.
S_final = 7.494;//[kJ/kg-K]
//At 150 C, and 0.1 MPa: V = 1.9364 m^(3)/kg, U =
    2582.8 kJ/kg, S = 7.6134 kJ/kg-K
//At 150 C, and 0.2 MPa: V = 0.9596 m^(3)/kg, U =
    2576.9 kJ/kg, S = 7.2795 kJ/kg-K
U_t_1 = 2582.8;//[kJ/kg] - Internal energy
U_t_2 = 2576.9;//[kJ/kg]
V_t_1 = 1.9364;//[m^(3)/kg] - Specific volume
```

```
V_t_2 = 0.9596;//[m^(3)/kg]
S_t_1 = 7.6134;//[kJ/kg-K] - Entropy
S_t_2 = 7.2795;//[kJ/kg-K]
//The pressure at exit is given by,
P_2 = ((S_final - S_t_1)/(S_t_2 - S_t_1))*(0.2 -
    0.1) + 0.1;//[Mpa] - Final pressure
//At final state
U_2 = U_t_1 + (U_t_2 - U_t_1)*((S_final - S_t_1)/(
    S_t_2 - S_t_1));//[kJ/kg] - Final specific
    internal energy
V_2 = V_t_1 + (V_t_2 - V_t_1)*((S_final - S_t_1)/(
    S_t_2 - S_t_1));//[m^(3)/kg] - Final specific
    volume

q = Q/m;//[kJ/kg] - Heat supplied per unit kg of
    mass.
W_rev_use = U_1 - U_2 + P_0*(V_1 - V_2) - T_0*(S_1 -
     S_2) + q*(1 - (T_0/T_R));//[kJ/kg] - Reversible
    useful work done.

//Now let us calculate the actual work done. We know
     q - W = delta_U , therefore
W = q - (U_2 - U_1);//[kJ/kg] - Work done
W_use = W - P_0*(V_2 - V_1);//[kJ/kg]
i = W_rev_use - W_use;//[kJ/kg] - Irreversibility
//Since the system contains 4.6 g therefore,
W_use_new = W_use*m;//[kJ]
W_rev_use_new = W_rev_use*m;//[kJ]
I = W_rev_use_new - W_use_new;//[kJ]

printf(" (1).The useful work obtained is %f kJ\n\n",
    W_use_new);
printf(" (2).The reversible usefuk work done is %f
    kJ\n\n",W_rev_use_new);
printf(" (3).The irreversibility is %f kJ\n\n",I);
```

Scilab code Exa 5.5 Determination of reversible work and irreversibility

```
clear;
clc;

//Example - 5.5
//Page number - 197
printf("Example - 5.5 and Page number - 197\n\n")

//Given
T_1 = 700+273.15;//[K] - Initial temperature.
P_1 = 12;//[MPa] - Initial pressure
P_2 = 0.6;//[MPa] - Final pressure
//At 12 MPa and 700 C,
H_1 = 3858.4;//[kJ/kg] - initial enthalpy
S_1 = 7.0757;//[kJ/kg-K] - initial entropy

//At 0.6 MPa and 200 C,
H_2 = 2850.1;//[kJ/kg]
S_2 = 6.9673;//[kJ/kg-K]

//At 0.6 MPa and 250 C,
H_3 = 2957.2;//[kJ/kg]
S_3 = 7.1824;//[kJ/kg-K]

//At 0.6 MPa and 300 C,
H_4 = 3061.6;//[kJ/kg]
S_4 = 7.3732;//[kJ/kg-K]

//(1)
//In the case of ideal turbine the entropy change
   does not take place,therefore the exit conditions
    are
P_exit = P_2;//[MPa] - exit pressure
```

```
T_exit = ((S_1 - S_2)/(S_3 - S_2))*(250 - 200) +
    200;//[C] - exit temperature
H_exit = ((S_1 - S_2)/(S_3 - S_2))*(H_3 - H_2) + H_2
    ;//[kJ/kg] - exit enthalpy

//Snce it is a flow pocess,therfore
//W_rev = H_1 - H_exit - T_0*(S_1 - S_2)
//As S_1 = S_2,the above equation becomes
W_rev_1 = H_1 - H_exit;//[kJ/kg] - reversible work
    done

//From the first law the actual work done can be
    calculated using, delta_H = q - W
//Since the turbine does not exchange heat,therefore
     W = - delta_H.
W_1 = - (H_exit - H_1);//[kJ/kg]

printf(" (1).The reversible work done is %f kJ/kg\n"
    ,W_1);
printf("      And since the maximum work is same as
    the actual work,therefore irreversibility is zero
    \n\n");

//(2)
//Given
T_0 = 298.15;//[K] - Environment temperature
P_0 = 1;//[atm] - Environment pressure
adi_eff = 0.88;//adiabatc efficiency

//(H_1 - H_exit_actual)/(H_1 - H_exit) = 0.88,
    therefore
H_exit_actual = H_1 - 0.88*(H_1 - H_exit);// -
    Actual exit enthalpy

//Now two properties i.e pressure = 0.6 MPa and
    enthalpy = H_exit_actual is fixed at the exit.
    The exit temperature is given by,
T_exit_actual = ((H_exit_actual - H_3)/(H_4 - H_3))
```

```
    *(300 - 250) + 250;//[C]
S_exit_actual = ((H_exit_actual - H_3)/(H_4 - H_3))
    *(S_4 - S_3) + S_3;//[kJ/kg]

//Now reversible work done is given by,
W_rev_2 = H_1 - H_exit_actual - T_0*(S_1 -
    S_exit_actual);//[kJ/kg]
printf(" (2).The reversible work done is %f kJ/kg\n"
    ,W_rev_2);

//The actual work is given by the first law,
W_2 = H_1 - H_exit_actual;//[kJ/kg] - Actual work
    done
i = W_rev_2 - W_2;//[kJ/kg] - irreversibility
printf("     The value of irreversibility is %f kJ/
    kg\n",i);

//The irreversibility can also be determined using
// i = T_0*S_gen, and S_gen is given by
// S_gen = (q/T_R) - delta_S

//The second law efficiency of the turbine is actual
     work done divided by reversible work,therefore
sec_eff = W_2/W_rev_2;
printf("     The second law efficiency of the
    turbine is %f\n",sec_eff);
```

Scilab code Exa 5.6 Determination of maximum obtainable work and efficiency

```
clear;
clc;

//Example - 5.6
//Page number - 198
```

```
printf("Example - 5.6 and Page number - 198\n\n")

//Given
P_1 = 8;//[bar] - Initial pressure
T_1 = 93 + 273.15;//[C] - Initial temperature
V_1 = 100;//[m/s] - Initial velocity
P_2 = 1.25;//[bar] - Exit pressure
T_2 = 27 + 273.15;//[C] - Exit temperature
V_2 = 60;//[m/s] - Exit velocity
Y = 1.4;//Ratio of specific heat capacities
T_0 = 298.15;//[K] - surrounding temperature
P_0 = 1;//[bar] - surrounding pressure
R = 8.314;//[J/mol*K] - Gas constant
Cp_0 = (R*Y)/(Y-1);//[J/mol-K] - Heat capacity at
    constant pressure

//Since the amount of heat transfer is negligible ,
    therefore from first law the actual work done is
    given by,
//W = delta_H + (delta_V_square)/2
delta_H = Cp_0*(T_2 - T_1);//[J/mol] - enthalpy
    change
delta_H = (delta_H/28.84);//[kJ/kg] - (1 mole =
    28.84 g).
delta_V_square = V_2^(2) - V_1^(2);

W = - delta_H - ((delta_V_square)/2)/1000;//[kJ/kg]
    - Actual work done
printf(" The actual work done is %f kJ/kg\n\n",W);

//Now let us calculate the maximum work that can be
    obtained
//W_rev = (H_1 + (V_1^(2))/2) - (H_2 + (V_2^(2))/2)
    - T_0*(S_1 - S_2)
delta_S = Cp_0*log(T_2/T_1) - R*log(P_2/P_1);//[J/
    mol-K] - Entropy change
delta_S = delta_S/28.84;//kJ/kg-K]
W_rev = -delta_H - ((delta_V_square/2)/1000) + T_0*
```

```
    delta_S;//[kJ/kg]
printf(" The maximum work obtainable per kg of air
    is %f kJ/kg\n\n",W_rev);

//The second law efficiency of the turbine is actual
     work done divided by reversible work,therefore
sec_eff = W/W_rev;
printf(" The second law efficiency of the turbine is
     %f\n\n",sec_eff);
```

Scilab code Exa 5.7 Determination of entropy generation rate and irreversibility

```
clear;
clc;

//Example - 5.7
//Page number - 200
printf("Example - 5.7 and Page number - 200\n\n")

//Given
m_cold_water = 60;//[kg/s] - mass flow rate of cold
    water
P_1 = 50;//[kPa]
T_2 = 250;//[C]
T_water_1 = 1000 + 273.15;//[K] - Entering
    temperature of water
T_water_2 = 450 +273.15;//[K] - Exit temperature of
    water
T_0 = 298.15;//[K] - surrounding temperature
P_0 = 1;//[atm] - surrounding pressure
Cp_0 = 1.005;//[kJ/kg-K]

//For water at 50 kPa under saturated conditions,
    T_sat = 81.33 C,
H_liq_1 = 340.49;//[kJ/kg] - Enthalpy
S_liq_1 = 1.0910;//[kJ/kg-K] - Entropy

//For steam at 50 kPa and 250 C,
H_2 = 2976.0;//[kJ/kg]
S_2 = 8.3556;//[kJ/kg-K]

//The cold stream is water which enters as saturated
    liquid at 50 kPa and exits as superheated vapour
    at 50 kPa and 250 C,since pressure drop is
   neglected.
//The mass flow rate of hot stream can be obtained
   from energy balance
m_hot_water = (m_cold_water*(H_2 - H_liq_1))/(Cp_0*(
   T_water_1 - T_water_2));//[kg/s] - mass flow rate
    of hot water

//Since there is no heat exchange with the
   surrounding therefore the total entropy
   generation is given by
//S_gen = delta_S_hot + delta_S_cold
delta_S_cold = S_2 - S_liq_1;//[kJ/kg-K] - change of
    entropy of cold water
//delta_S_hot = Cp_0*log(T_2/T_1)-R*log(P_2/P_1),
   But pressure drop is zero,therfore
delta_S_hot = Cp_0*log(T_water_2/T_water_1);//[kJ/kg
   -K] - change of entropy of hot water

S_gen = m_cold_water*delta_S_cold + m_hot_water*
   delta_S_hot;//[kW/K] - Entropy generated
printf(" The entropy generation rate is %f kW/K\n\n"
   ,S_gen);

//The irreversibility rete is given by
I = T_0*S_gen;//[kW]
printf(" The irreversibility rate of the heat
   exchanger is %f kW\n",I);

```

```
//The irreversibility can also be determined using
    the exergy approach
//We know that , I = W_rev - , but since actual work
    done zero in heat exchangers ,therefore I = W_rev
    = exergy change
//(si_1 - si_2)_cold = H_1 - H_2 - T_0*(S_1 - S_2)
//(si_1 - si_2)_hot = Cp_0*(T_1 - T_2)- T_0*(S_1 -
    S_2)
// I = (si_1 - si_2)_cold - (si_1 - si_2)_hot.
```

Scilab code Exa 5.8 Calculation of exit temperature entropy and irreversibility rate

```
clear;
clc;

//Example - 5.8
//Page number - 201
printf("Example - 5.8 and Page number - 201\n\n")

//Given
m_water = 10000;//[kg/h] - Mass flow rate of cold
    water
m_water = m_water/3600;//[kg/s]
T_1_water = 30 + 273.15;//[K] - Cold water entering
    temperature
m_HC = 5000;//[kg/h] - mass flow rate of hot
    hydrocarbon
m_HC = m_HC/3600;//[kg/s]
T_1_HC = 200 + 273.15;//[K] - Hot hydrocarbon
    entering temperature
T_2_HC = 100 + 273.15;//[K] - Hot hydrocarbon
    leaving temperature
Cp_0_water = 1.0;//[kcal/kg-K] - Mean heat capacity
    of cooling water
```

```
Cp_0_HC = 0.6;//[kcal/kg-K] - Mean heat capacity of
    hydrocarbon

//(1)
//Applying energy balance to the heat exchanger,we
    get
//m_water*Cp_0_water*(T - T_1_water) = m_HC*Cp_0_HC
    *(T_1_HC - T_2_HC)
T_2_water = ((m_HC*Cp_0_HC*(T_1_HC - T_2_HC))/(
    m_water*Cp_0_water)) + T_1_water;//[K]
T_2 = T_2_water - 273.15;//[C]
printf(" (1).The exit temperature of the cooling
    water is %f C\n\n",T_2);

//(2)
//delta_S_hot_HC = Cp_0*log(T_2/T_1)-R*log(P_2/P_1),
     But pressure drop is zero,therfore
delta_S_hot_HC = (Cp_0_HC*4.184)*log(T_2_HC/T_1_HC);
    //[kW/K] - change of entropy of hot hydrocarbon
delta_S_HC = m_HC*delta_S_hot_HC;//[kW/K] - Entropy
    change for hudrocarbon liquid
printf(" (2).Entropy change rate of hydrocarbon
    liquid is %f kW/K\n",delta_S_HC);

delta_S_cold_water = (Cp_0_water*4.184)*log(
    T_2_water/T_1_water);//[kW/K] - change of entropy
     of cooling water
delta_S_water = m_water*delta_S_cold_water;//[kW/K]
    - Entropy change for water
printf("     And entropy change rate of water is %f
    kW/K\n\n",delta_S_water);

//(3)
T_0 = 298.15;//[K] - Surrounding temperature
//S_gen = delta_S_cold_water + delta_S_hot_HC =
    m_water*delta_S_cold_water + m_HC*delta_S_hot_HC
    ;//[kW/K] - Entropy generated
S_gen = delta_S_water + delta_S_HC;//[kW/K]
```

```
I = T_0*S_gen;//[kW]
printf(" (3).The irreversibility rate of the heat
    exchanger is %f kW\n",I);
```

Scilab code Exa 5.9 Determinatio of exit temperature availability change and irreversibility

```
clear;
clc;

//Example - 5.9
//Page number - 202
printf("Example - 5.9 and Page number - 202\n\n");

//Given
T_1_hotgas = 800;//[K]
P_1_hotgas = 1;//[bar]
T_2_hotgas = 700;//[K]
P_2_hotgas = 1;//[bar]
T_1_air = 470;//[K]
P_1_air = 1;//[bar]
P_2_air = 1;//[bar]
Cp_0_hotgas = 1.08;//[kJ/kg-K] - Mean heat capacity
    of hot gas
Cp_0_air = 1.05;//[kcal/kg-K] - Mean heat capacity
    of air
T_0 = 298.15;//[K] - surrounding temperature
P_0 = 1;//[bar] - surrounding pressure
//m_air = 2*m_hotgas

//(1)
//Assuming heat exchange only takes places in-
    between the streams,from energy balance we get,
//m_gas*Cp_0_hotgas*(T_2_hotgas - T_1_hotgas) + 2*
    m_gas*Cp_0_air*(T - T_1_air),
T_2_air = T_1_air - ((Cp_0_hotgas*(T_2_hotgas -
   T_1_hotgas))/(2*Cp_0_air));//[K] - Temp of
   emerging air
printf(" (1).The temperature of emerging air is %f K
   \n\n",T_2_air);

//(2)
//Availability change of hot gas is given by,
//(si_1 - si_2)_hot = H_1 - H_2 - T_0*(S_1 - S_2)
delta_H_hotgas = (Cp_0_hotgas*(T_2_hotgas -
   T_1_hotgas));//[kJ/kg] - change in enthalpy of
   hotgas
//delta_S_hotgas = Cp_0_hotgas*log(T_2_hotgas/
   T_1_hotgas)- R*log(P_2/P_1), But pressure drop is
    zero (P_1 = P_2),therfore
delta_S_hotgas = Cp_0_hotgas*log(T_2_hotgas/
   T_1_hotgas);//[kJ/kg-K] - change of entropy of
   hot gas
delta_si_hotgas = (-delta_H_hotgas) - (-T_0*
   delta_S_hotgas);//[kJ/kg]
printf(" (2).The availability change of hot gas is
   %f kJ/kg\n\n",delta_si_hotgas);

//(3)
//Availability change of air is given by,
//(si_1 - si_2)_air = H_1 - H_2 - T_0*(S_1 - S_2)
delta_H_air = (Cp_0_air*(T_2_air - T_1_air));//[kJ/
   kg] - change in enthalpy of air
//delta_S_air = Cp_0_air*log(T_2_air/T_1_air)- R*log
   (P_2/P_1), But pressure drop is zero (P_1 = P_2),
   therfore
delta_S_air = Cp_0_air*log(T_2_air/T_1_air);//[kJ/kg
   -K] - change of entropy of air
delta_si_air = (-delta_H_air) - (-T_0*delta_S_air);
   //[kJ/kg]
printf(" (3).The availability change of air is %f kJ
   /kg\n\n",delta_si_air);

```

```
//(4)
//For the heat exchanger (Q = 0, W = 0)
//Basis : 1 kg of hot gas flowing through heat
   exchanger
S_gen = delta_S_hotgas + 2*delta_S_air;//[kJ/kg-K] -
    (as  m_air = 2*m_hotgas)
I = T_0*S_gen;//[kJ/kg]
printf(" (4).The irreversibility of thr exchanger
   per kg of hot gas flowing is %f kJ/kg\n",I);

//Irreversibility can also be obtained using
//I = 2*(si_1 - si_2)_air + (si_1 - si_2)_hotgas
```

Chapter 6

Chemical reactions

Scilab code Exa 6.1 Determination of enthalpy entropy and Gibbs free energy change of reaction

```
clear;
clc;

//Example - 6.1
//Page number - 217
printf("Example - 6.1 and Page number - 217\n\n");


//Given
T_1 = 298.15;//[K] - Standard temperature
T_2 = 880;//[K] - Reaction temperature

a_SO2 = 6.157;
a_SO3 = 3.918;
a_O2 = 6.732;
b_SO2 = 1.384*10^(-2);
b_SO3 = 3.483*10^(-2);
b_O2 = 0.1505*10^(-2);
c_SO2 = -0.9103*10^(-5);
c_SO3 = -2.675*10^(-5);
```

```
c_O2 = -0.01791*10^(-5);
d_SO2 = 2.057*10^(-9);
d_SO3 = 7.744*10^(-9);

delta_H_rkn_298 = -23.45*10^(3);//[cal] - Rkn
    enthalpy at 298.15 K
delta_H_SO2_for_298 = -70.94*10^(3);//[cal/mol] -
    Enthalpy of formation of S02 at 298.15 K
delta_H_SO3_for_298 = -94.39*10^(3);//[cal/mol] -
    Enthalpy of formation of SO3 at 298.15 K
delta_G_SO2_for_298 = -71.68*10^(3);//[cal/mol] -
    Gibbs free energy change for formation of SO2 at
    298.15 K
delta_G_SO3_for_298 = -88.59*10^(3);//[cal/mol] -
    Gibbs free energy change for formation of SO3 at
    298.15 K

//(1)
//Standard enthalpy change of reaction at
    temperature T is given by,
//delta_H_rkn_T = delta_rkn_298 + delta_Cp_0*delta_T
delta_a = a_SO3 - a_SO2 - (a_O2/2);
delta_b = b_SO3 - b_SO2 - (b_O2/2);
delta_c = c_SO3 - c_SO2 - (c_O2/2);
delta_d = d_SO3 - d_SO2;

//Cp_0 = delta_a + (delta_b*T) + (delta_c*T^(2)) + (
    delta_d*T^(3));
//Therefore we get,
delta_H_rkn_880 = delta_H_rkn_298 + integrate('
    delta_a+(delta_b*T)+(delta_c*T^(2))+(delta_d*T
    ^(3))','T',T_1,T_2);

//On manual simplification of the above expression,
    we will get the expression for 'delta_H_rkn_880'
    as a function of T,

printf(" (1).The expression for standard enthalpy
    change of reaction as a function of temperature
    is given by\n");
printf("      delta_H_rkn_880 = -22534.57 - 5.605*T
    + 1.012*10^(-2)*T^(2) - 0.585*10^(-5)*T^(3) +
    1.422*10^(-9)*T^(4)\n\n")

printf(" (2).Standard enthalpy change of reaction at
     880 K is %f cal\n\n",delta_H_rkn_880);

//(3)
//Let us determine the standard entropy change of
    reaction at 298.15 K
delta_S_SO2_298 = (delta_H_SO2_for_298 -
    delta_G_SO2_for_298)/298.15;//[cal/mol-K]
delta_S_SO3_298 = (delta_H_SO3_for_298 -
    delta_G_SO3_for_298)/298.15;//[cal/mol-K]
delta_S_O2_298 = 0;//[cal/mol-K]

delta_S_rkn_298 = delta_S_SO3_298 - delta_S_SO2_298
    - (delta_S_O2_298/2);//[cal/K]
delta_S_rkn_880 = delta_S_rkn_298 + integrate('(
    delta_a+delta_b*T+delta_c*T^(2)+delta_d*T^(3))/T'
    ,'T',T_1,T_2);//[cal/K]

printf(" (3).Standard entropy change of reaction at
    880 K is %f cal/K\n\n",delta_S_rkn_880);

//(4)
delta_G_rkn_880 = delta_H_rkn_880 - 880*
    delta_S_rkn_880;//[cal]

printf(" (4).Standard Gibbs free energy change of
    reaction at 880 K is %f cal\n\n",delta_G_rkn_880)
    ;
```

Scilab code Exa 6.2 Determination of standard enthalpy and Gibbs free energy change of reaction

```
clear;
clc;

//Example - 6.2
//Page number - 219
printf("Example - 6.2 and Page number - 219\n\n");

//Given
T_1 = 298.15;//[K] - Standard temperature
T_2 = 400;//[K] - Reaction temperature

a_CH3OH = 4.55;
a_CO = 6.726;
a_H2 = 6.952;
b_CH3OH = 2.186*10^(-2);
b_CO = 0.04001*10^(-2);
b_H2 = -0.04576*10^(-2);
c_CH3OH = -0.291*10^(-5);
c_CO = 0.1283*10^(-5);
c_H2 = 0.09563*10^(-5);
d_CH3OH = -1.92*10^(-9);
d_CO = -0.5307*10^(-9);
d_H2 = -0.2079*10^(-9);

delta_H_rkn_298 = -21.6643*10^(3);//[cal] - Reaction
     enthalpy at 298.15 K
delta_H_CO_for_298 = -26.4157*10^(3);//[cal/mol] -
    Enthalpy of formation of CO at 298.15 K
delta_H_CH3OH_for_298 = -48.08*10^(3);//[cal/mol] -
    Enthalpy of formation of CH3OH at 298.15 K
delta_G_CO_for_298 = -32.8079*10^(3);//[cal/mol] -
    Gibbs free energy change for formation of CO at
    298.15 K
delta_G_CH3OH_for_298 = -38.69*10^(3);//[cal/mol] -
    Gibbs free energy change for formation of CH3OH
    at 298.15 K

//Standard enthalpy change of reaction at
    temperature T is given by,
//delta_H_rkn_T = delta_rkn_298 + delta_Cp_0*delta_T
delta_a = a_CH3OH - a_CO - 2*(a_H2);
delta_b = b_CH3OH - b_CO - 2*(b_H2);
delta_c = c_CH3OH - c_CO - 2*(c_H2);
delta_d = d_CH3OH - d_CO - 2*(d_H2);

//Cp_0 = delta_a + (delta_b*T) + (delta_c*T^(2)) + (
    delta_d*T^(3));
//Therefore we get,
delta_H_rkn_400 = delta_H_rkn_298 + integrate('
    delta_a+(delta_b*T)+(delta_c*T^(2))+(delta_d*T
    ^(3))','T',T_1,T_2);

printf(" Standard enthalpy change of reaction at 400
     K is %f cal\n\n",delta_H_rkn_400);

//Let us determine the standard Gibbs free energy
    change of reaction at 298.15 K
delta_G_rkn_298 = delta_G_CH3OH_for_298 -
    delta_G_CO_for_298;//[cal]

//Now determining the standard entropy change of
    reaction at 298.15 K
delta_S_rkn_298 = (delta_H_rkn_298 - delta_G_rkn_298
    )/298.15;//[cal/mol-K]

delta_S_rkn_400 = delta_S_rkn_298 + integrate('(
    delta_a+delta_b*T+delta_c*T^(2)+delta_d*T^(3))/T'
    ,'T',T_1,T_2);//[cal/K]
//Therefore,the standard Gibbs free energy change of
     the reaction is given by,
delta_G_rkn_400 = delta_H_rkn_400 - 400*
    delta_S_rkn_400;//[cal]

```

```
printf(" Standard Gibbs free energy change of
   reaction at 400 K is %f cal\n",delta_G_rkn_400);
```

Scilab code Exa 6.3 Determination of standard enthalpy and Gibbs free energy change of reaction

```
clear;
clc;

//Example - 6.3
//Page number - 220
printf("Example - 6.3 and Page number - 220\n\n");

//Given
T_1 = 298.15;//[K] - Standard temperature
T_2 = 1200;//[K] - Reaction temperature


a_CO2 = 5.316;
a_H2 = 6.952;
a_CO = 6.726;
a_H2O = 7.700;
b_CO2 = 1.4285*10^(-2);
b_H2 = -0.04576*10^(-2);
b_CO = 0.04001*10^(-2);
b_H2O = 0.04594*10^(-2);
c_CO2 = -0.8362*10^(-5);
c_H2 = 0.09563*10^(-5);
c_CO = 0.1283*10^(-5);
c_H2O = 0.2521*10^(-5);
d_CO2 = 1.784*10^(-9);
d_H2 = -0.2079*10^(-9);
d_CO = -0.5307*10^(-9);
d_H2O = -0.8587*10^(-9);

```

```
delta_H_rkn_298 = -9.8382*10^(3);//[cal] - Reaction
   enthalpy at 298.15 K
delta_H_CO2_for_298 = -94.0518*10^(3);//[cal/mol-K]
   - Enthalpy of formation of CO2 at 298.15 K
delta_H_CO_for_298 = -26.4157*10^(3);//[cal/mol-K] -
   Enthalpy of formation of CO at 298.15 K
delta_H_H2O_for_298 = -57.7979*10^(3);//[cal/mol-K]
   - Enthalpy of formation of H2O at 298.15 K
delta_G_CO2_for_298 = -94.2598*10^(3);//[cal/mol] -
   Gibbs free energy change for formation of CO at
   298.15 K
delta_G_CO_for_298 = -32.8079*10^(3);//[cal/mol] -
   Gibbs free energy change for formation of CH3OH
   at 298.15 K
delta_G_H2O_for_298 = -54.6357*10^(3);//[cal/mol] -
   Gibbs free energy change for formation of H2O at
   298.15 K

//Standard enthalpy change of reaction at
   temperature T is given by,
//delta_H_rkn_T = delta_rkn_298 + delta_Cp_0*delta_T
delta_a = a_CO2 + a_H2 - a_CO - a_H2O;
delta_b = b_CO2 + b_H2 - b_CO - b_H2O;
delta_c = c_CO2 + c_H2 - c_CO - c_H2O;
delta_d = d_CO2 + d_H2 - d_CO - d_H2O;

//Cp_0 = delta_a + (delta_b*T) + (delta_c*T^(2)) + (
   delta_d*T^(3));
//Therefore we get,
delta_H_rkn_1200 = delta_H_rkn_298 + integrate('
   delta_a+(delta_b*T)+(delta_c*T^(2))+(delta_d*T
   ^(3))','T',T_1,T_2);

printf(" Standard enthalpy change of reaction at
   1200 K is %f cal\n\n",delta_H_rkn_1200);

//Let us determine the standard Gibbs free energy
   change of reaction at 298.15 K
```

```
delta_G_rkn_298 = delta_G_CO2_for_298 -
   delta_G_CO_for_298 - delta_G_H2O_for_298;//[cal]

//Now determining the standard entropy change of
   reaction at 298.15 K
delta_S_rkn_298 = (delta_H_rkn_298 - delta_G_rkn_298
   )/298.15;//[cal/mol-K]

delta_S_rkn_1200 = delta_S_rkn_298 + integrate('(
   delta_a+delta_b*T+delta_c*T^(2)+delta_d*T^(3))/T'
   ,'T',T_1,T_2);//[cal/K]
//Therefore,the standard Gibbs free energy change of
    the reaction is given by,
delta_G_rkn_1200 = delta_H_rkn_1200 - 1200*
   delta_S_rkn_1200;//[cal]

printf(" Standard Gibbs free energy change of
   reaction at 1200 K is %f cal",delta_G_rkn_1200);
```

Scilab code Exa 6.4 Determination of standard enthalpy and Gibbs free energy change of reaction

```
clear;
clc;

//Example - 6.4
//Page number - 221
printf("Example - 6.4 and Page number - 221\n\n");

//Given
T_1 = 298.15;//[K] - Standard temperature
T_2 = 500;//[K] - Reaction temperature

a_NH3 = 6.5846;
a_N2 = 6.903;
```

```
a_H2 = 6.952;
b_NH3 = 0.61251*10^(-2);
b_N2 = -0.03753*10^(-2);
b_H2 = -0.04576*10^(-2);
c_NH3 = 0.23663*10^(-5);
c_N2 = 0.1930*10^(-5);
c_H2 = 0.09563*10^(-5);
d_NH3 = -1.5981*10^(-9);
d_N2 = -0.6861*10^(-9);
d_H2 = -0.2079*10^(-9);

delta_H_rkn_298 = -22.08*10^(3);//[cal] - Reaction
   enthalpy at 298.15 K
delta_H_NH3_for_298 = -11.04*10^(3);//[cal/mol] -
   Enthalpy of formation of NH3 at 298.15 K
delta_G_NH3_for_298 = -3.976*10^(3);//[cal/mol] -
   Gibbs free energy change for formation of NH3 at
   298.15 K

//Standard enthalpy change of reaction at
   temperature T is given by,
//delta_H_rkn_T = delta_rkn_298 + delta_Cp_0*delta_T
delta_a = 2*a_NH3 - a_N2 - 3*a_H2;
delta_b = 2*b_NH3 - b_N2 - 3*b_H2;
delta_c = 2*c_NH3 - c_N2 - 3*c_H2;
delta_d = 2*d_NH3 - d_N2 - 3*d_H2;

//Cp_0 = delta_a + (delta_b*T) + (delta_c*T^(2)) + (
   delta_d*T^(3));
//Therefore we get,
delta_H_rkn_500 = delta_H_rkn_298 + integrate('
   delta_a+(delta_b*T)+(delta_c*T^(2))+(delta_d*T
   ^(3))','T',T_1,T_2);

printf(" Standard enthalpy change of reaction at 500
    K is %f cal\n\n",delta_H_rkn_500);

//Let us determine the standard Gibbs free energy
```

```
    change of reaction at 298.15 K
delta_G_rkn_298 = 2*delta_G_NH3_for_298;//[cal]

//Now determining the standard entropy change of
    reaction at 298.15 K
delta_S_rkn_298 = (delta_H_rkn_298 - delta_G_rkn_298
    )/298.15;//[cal/mol-K]

delta_S_rkn_500 = delta_S_rkn_298 + integrate('(
    delta_a+delta_b*T+delta_c*T^(2)+delta_d*T^(3))/T'
    ,'T',T_1,T_2);//[cal/K]
//Therefore,the standard Gibbs free energy change of
     the reaction is given by,
delta_G_rkn_500 = delta_H_rkn_500 - 500*
    delta_S_rkn_500;//[cal]

printf(" Standard Gibbs free energy change of
    reaction at 500 K is %f cal",delta_G_rkn_500);
```

Scilab code Exa 6.5 Determination of standard enthalpy and Gibbs free energy change of reaction

```
clear;
clc;

//Example - 6.5
//Page number - 222
printf("Example - 6.5 and Page number - 222\n\n");

//Given
//Cp_0 = 7.7 + 0.04594*10^(-2)*T + 0.2521*10^(-5)*T
    ^(2) - 0.8587*10^(-9)*T^(3)

delta_H_rkn_298 = -57.7979*10^(3);//[cal/mol] -
    Reaction enthalpy at 298.15 K
```

```
delta_G_rkn_298 = -54.6351*10^(3);//[cal/mol] -
   Gibbs free energy change for formation of H2O at
   298.15 K

//Standard enthalpy change of reaction at
   temperature T is given by,
//delta_H_rkn_T = delta_rkn_298 + delta_Cp_0*delta_T
T_1 = 298.15;//[K] - Standard temperature
T_2_1 = 873.15;//[K] - Reaction temperature
T_2_2 = 1000;//[K] - Reaction temperature

//Therefore we get,
delta_H_rkn_873 = delta_H_rkn_298 + integrate('
   7.7+0.04594*10^(-2)*T+0.2521*10^(-5)*T^(2)
   -0.8587*10^(-9)*T^(3)','T',T_1,T_2_1);;//[cal/mol
   ]
delta_H_rkn_1000 = delta_H_rkn_298 + integrate('
   7.7+0.04594*10^(-2)*T+0.2521*10^(-5)*T^(2)
   -0.8587*10^(-9)*T^(3)','T',T_1,T_2_2);//[cal/mol]

printf(" Standard enthalpy change of reaction at 873
    K is %f cal/mol\n\n",delta_H_rkn_873);
printf(" Standard enthalpy change of reaction at
   1000 K is %f cal/mol\n\n",delta_H_rkn_1000);

//Now determining the standard entropy change of
   reaction at 298.15 K
delta_S_rkn_298 = (delta_H_rkn_298 - delta_G_rkn_298
   )/298.15;//[cal/mol-K]

delta_S_rkn_873 = delta_S_rkn_298 + integrate('
   (7.7+0.04594*10^(-2)*T+0.2521*10^(-5)*T^(2)
   -0.8587*10^(-9)*T^(3))/T','T',T_1,T_2_1);//[cal/
   mol-K]
delta_S_rkn_1000 = delta_S_rkn_298 + integrate('
   (7.7+0.04594*10^(-2)*T+0.2521*10^(-5)*T^(2)
   -0.8587*10^(-9)*T^(3))/T','T',T_1,T_2_2);//[cal/
   mol-K]
```

```
//Therefore,the standard Gibbs free energy change of
    the reaction is given by,
delta_G_rkn_873 = (delta_H_rkn_873 - 873.15*
   delta_S_rkn_873)*10^(-3);//[kcal/mol]
delta_G_rkn_1000 = (delta_H_rkn_1000 - 1000*
   delta_S_rkn_1000)*10^(-3);//[kcal/mol]

printf(" Standard Gibbs free energy change of
   reaction at 873 K is %f kcal/mol\n\n",
   delta_G_rkn_873);
printf(" Standard Gibbs free energy change of
   reaction at 1000 K is %f kcal/mol\n",
   delta_G_rkn_1000);
```

Scilab code Exa 6.6 Calculation of heat exchange

```
clear;
clc;

//Example - 6.6
//Page number - 223
printf("Example - 6.6 and Page number - 223\n\n");

//Given
T_1 = 298.15;//[K] - Standard temperature
T_2 = 500;//[K] - Reaction temperature

a_C2H6 = 1.648;
a_O2 = 6.085;
a_CO2 = 5.316;
a_H2O = 7.700;
b_C2H6 = 4.124*10^(-2);
b_O2 = 0.3631*10^(-2);
b_CO2 = 1.4285*10^(-2);
b_H2O = 0.04594*10^(-2);
```

```
c_C2H6 = -1.530*10^(-5);
c_O2 = -0.1709*10^(-5);
c_CO2 = -0.8362*10^(-5);
c_H2O = 0.2521*10^(-5);
d_C2H6 = 1.740*10^(-9);
d_O2 = 0.3133*10^(-9);
d_CO2 = 1.784*10^(-9);
d_H2O = -0.8587*10^(-9);

//Since excess is entering and leaving at the same
   temperature,therefore it does not take or give
   any heat to the system.
//Therefore the heat exchange is only due to heat of
    raction at temperature T, or Q = delta_H_rkn_T

delta_H_C2H6_for_298 = -20.236*10^(3);//[cal/mol] -
   Enthalpy of formation of C2H6 at 298.15 K
delta_H_CO2_for_298 = -94.0518*10^(3);//[cal/mol] -
   Enthalpy of formation of CO2 at 298.15 K
delta_H_H2O_for_298 = -57.7979*10^(3);//[cal/mol] -
   Enthalpy of formation of H2O at 298.15 K

delta_H_rkn_298 = 2*delta_H_CO2_for_298 + 3*
   delta_H_H2O_for_298 - delta_H_C2H6_for_298;//[cal
   ] - Reaction enthalpy at 298.15 K

//Standard enthalpy change of reaction at
   temperature T is given by,
//delta_H_rkn_T = delta_rkn_298 + delta_Cp_0*delta_T
delta_a = 2*a_CO2 + 3*a_H2O - a_C2H6 - 7/2*(a_O2);
delta_b = 2*b_CO2 + 3*b_H2O - b_C2H6 - 7/2*(b_O2);
delta_c = 2*c_CO2 + 3*c_H2O - c_C2H6 - 7/2*(c_O2);
delta_d = 2*d_CO2 + 3*d_H2O - d_C2H6 - 7/2*(d_O2);

//Cp_0 = delta_a + (delta_b*T) + (delta_c*T^(2)) + (
   delta_d*T^(3));
//Therefore we get,
delta_H_rkn_500 = delta_H_rkn_298 + integrate('
    delta_a+(delta_b*T)+(delta_c*T^(2))+(delta_d*T
    ^(3))','T',T_1,T_2);//[cal]
delta_H_rkn_500 = delta_H_rkn_500*10^(-3);//[kcal]

printf(" The heat exchange of the reaction at 500 K
    is %f kcal",delta_H_rkn_500);
```

Scilab code Exa 6.7 Calculation of change in entropy

```
clear;
clc;

//Example - 6.7
//Page number - 224
printf("Example - 6.7 and Page number - 224\n\n");

//Given
T_1 = 298.15;// [K] - Standard temperature
T_2 = 600;// [K] - Reaction temperature

a_C2H6 = -8.65;
a_H2O = 7.700;
a_CH4 = 4.750;
a_O2 = 6.085;
b_C2H6 = 11.578*10^(-2);
b_H2O = 0.04594*10^(-2);
b_CH4 = 1.200*10^(-2);
b_O2 = 0.3631*10^(-2);
c_C2H6 = -7.540*10^(-5);
c_H2O = 0.2521*10^(-5);
c_CH4 = 0.3030*10^(-5);
c_O2 = -0.1709*10^(-5);
d_C2H6 = 18.54*10^(-9);
d_H2O = -0.8587*10^(-9);
d_CH4 = -2.630*10^(-9);
```

```
d_O2 = 0.3133*10^(-9);

delta_S_CH4_for_298 = 44.50;//[cal/mol-K] - Entropy
    of formation of CH4 at 298.15 K
delta_S_O2_for_298 = 49.00;//[cal/mol-K] - Entropy
    of formation of O2 at 298.15 K
delta_S_C2H6_for_298 = 64.34;//[cal/mol-K] - Entropy
     of formation of C2H6 at 298.15 K
delta_S_H2O_for_298 = 45.11;//[cal/mol-K] - Entropy
    of formation of C2H6 at 298.15 K

//Cp_0 = delta_a + (delta_b*T) + (delta_c*T^(2)) + (
    delta_d*T^(3));

//Standard entropy change of reaction at temperature
     T is given by,
//delta_S_rkn_T = delta_rkn_298 + delta_Cp_0*delta_T
delta_a = 1/6*(a_C2H6) + 3/2*(a_H2O) - a_CH4 - 3/4*(
    a_O2);
delta_b = 1/6*(b_C2H6) + 3/2*(b_H2O) - b_CH4 - 3/4*(
    b_O2);
delta_c = 1/6*(c_C2H6) + 3/2*(c_H2O) - c_CH4 - 3/4*(
    c_O2);
delta_d = 1/6*(d_C2H6) + 3/2*(d_H2O) - d_CH4 - 3/4*(
    d_O2);

delta_S_rkn_298 = 1/6*(delta_S_C2H6_for_298) + 3/2*(
    delta_S_H2O_for_298) - delta_S_CH4_for_298 -
    3/4*(delta_S_O2_for_298);//[cal/K]
delta_S_rkn_600 = delta_S_rkn_298 + integrate('(
    delta_a+delta_b*T+delta_c*T^(2)+delta_d*T^(3))/T'
    ,'T',T_1,T_2);//[cal/K]

printf(" Change in entropy of the reaction at 298.15
     K is %f cal/K\n\n",delta_S_rkn_298);
printf(" Standard entropy change of reaction at 600
    K is %f cal/K",delta_S_rkn_600);
```

Scilab code Exa 6.8 Calculation of standard enthalpy change and Gibbs free energy change

```
clear;
clc;

//Example - 6.8
//Page number - 225
printf("Example - 6.8 and Page number - 225\n\n");

//Given
T_1 = 298.15;//[K] - Standard temperature
T_2 = 973.15;//[K] - Reaction temperature

//At 298.15 K
delta_H_CH4_for_298 = -17.889*10^(3);//[cal/mol] -
    Enthalpy of formation of CH4 at 298.15 K
delta_H_C_for_298 = 0.00;//[cal/mol] - Enthalpy of
    formation of C (s, graphite) at 298.15 K
delta_H_H2_for_298 = 0.00;//[cal/mol] - Enthalpy of
    formation of H2 at 298.15 K
delta_G_CH4_for_298 = -12.140*10^(3);//[cal/mol] -
    Gibbs free energy change for formation of H2 at
    298.15 K
delta_G_C_for_298 = 0.00;//[cal/mol] - Gibbs free
    energy change for formation of C (s, graphite) at
     298.15 K
delta_G_H2_for_298 = 0.00;//[cal/mol] - Gibbs free
    energy change for formation of H2 at 298.15 K

///Standaerd heat capacity data in cal/mol-K are
    given below, T is in K
//Cp_0_CH4 = 4.75 + 1.2*10^(-2)*T + 0.303*10^(-5)*T
    ^(2) - 2.63*10^(-9)*T^(3)
```

```
//Cp_0_C = 3.519 + 1.532*10^(-3)*T - 1.723*10^(5)*T
    ^(-2)
//Cp_0_H2 = 6.952 - 0.04576*10^(-2)*T +
    0.09563*10^(-5)*T^(2) - 0.2079*10^(-9)*T^(3)

//Therefore standard heat capacity of reaction is
    given by,
//Cp_0_rkn = 2*Cp_0_H2 + Cp_0_C - Cp_0_CH4
//On simplification,we get the relation
//Cp_0_rkn = 12.673 - 0.0113832*T - 1.1174*10^(-6)*T
    ^(2) + 2.2142*10^(-9)*T^(3) - 1.723*10^(5)*T^(-2)

delta_H_rkn_298 = -delta_H_CH4_for_298;//[cal] -
    Enthalpy of reaction at 298.15 K
delta_G_rkn_298 = -delta_G_CH4_for_298;//[cal] -
    Gibbs free energy of the reaction at 298.15 K

delta_H_rkn_973 = delta_H_rkn_298 + integrate('
    12.673-0.0113832*T-1.1174*10^(-6)*T^(2)
    +2.2142*10^(-9)*T^(3)-1.723*10^(5)*T^(-2)','T',
    T_1,T_2);//[cal]

printf(" Standard enthalpy change of reaction at
    973.15 K is %f cal\n\n",delta_H_rkn_973);

//Now determining the standard entropy change of
    reaction at 298.15 K
delta_S_rkn_298 = (delta_H_rkn_298 - delta_G_rkn_298
    )/298.15;//[cal/K]
delta_S_rkn_973 = delta_S_rkn_298 + integrate('
    (12.673-0.0113832*T-1.1174*10^(-6)*T^(2)
    +2.2142*10^(-9)*T^(3)-1.723*10^(5)*T^(-2))/T','T'
    ,T_1,T_2);//[cal/K]

//Therefore,the standard Gibbs free energy change of
     the reaction is given by,
delta_G_rkn_973 = delta_H_rkn_973 - 973.15*
    delta_S_rkn_973;//[cal]
```

```

printf(" Standard Gibbs free energy change of
    reaction at 973 K is %f cal\n",delta_G_rkn_973);
```

Scilab code Exa 6.9 Calculation of standard enthalpy change and Gibbs free energy change

```
clear;
clc;

//Example - 6.9
//Page number - 226
printf("Example - 6.9 and Page number - 226\n\n");

//Given
T_1 = 298.15;//[K] - Standard temperature
T_2 = 1000;//[K] - Reaction temperature

//At 298.15 K
delta_H_C_for_298 = 0.00;//[cal/mol] - Enthalpy of
    formation of C(s,graphite) at 298.15 K
delta_H_H2O_for_298 = -57.7979*10^(3);//[cal/mol] -
    Enthalpy of formation of H2O at 298.15 K
delta_H_CO_for_298 = -26.4157*10^(3);//[cal/mol] -
    Enthalpy of formation of CO at 298.15 K
delta_H_H2_for_298 = 0.00;//[cal/mol] - Enthalpy of
    formation of H2 at 298.15 K
delta_G_C_for_298 = 0.00;//[cal/mol] - Gibbs free
    energy change for formation of C(s, graphite) at
    298.15 K
delta_G_H2O_for_298 = -54.6357*10^(3);//[cal/mol] -
    Gibbs free energy change for formation of H2O at
    298.15 K
delta_G_CO_for_298 = -32.8079*10^(3);//[cal/mol] -
    Gibbs free energy change for formation of CO at
```

```
   298.15 K
delta_G_H2_for_298 = 0.00;//[cal/mol] - Gibbs free
   energy change for formation of H2 at 298.15 K

///Standaerd heat capacity data in cal/mol-K are
   given below, T is in K
//Cp_0_C = 3.519 + 1.532*10^(-3)*T - 1.723*10^(5)*T
   ^(-2)
//Cp_0_H2O = 7.7 + 0.04594*10^(-2)*T +
   0.2521*10^(-5)*T^(2) - 0.8587*10^(-9)*T^(3)
//Cp_0_CO = 6.726 + 0.04001*10^(-2)*T +
   0.1283*10^(-5)*T^(2) - 0.5307*10^(-9)*T^(3)
//Cp_0_H2 = 6.952 - 0.04576*10^(-2)*T +
   0.09563*10^(-5)*T^(2) - 0.2079*10^(-9)*T^(3)

//Therefore standard heat capacity of reaction is
   given by,
//Cp_0_rkn = Cp_0_H2 + Cp_0_CO - Cp_0_C - Cp_0_H2O
//On simplification,we get the relation
//Cp_0_rkn = 2.459 - 2.0489*10^(-3)*T -
   2.817*10^(-7)*T^(2) + 1.201*10^(-10)*T^(3) +
   1.723*10^(5)*T^(-2)

delta_H_rkn_298 = delta_H_CO_for_298 +
   delta_H_H2_for_298 - delta_H_C_for_298 -
   delta_H_H2O_for_298;//[cal] - Enthalpy of
   reaction at 298.15 K
delta_G_rkn_298 = delta_G_CO_for_298 +
   delta_G_H2_for_298 - delta_G_C_for_298 -
   delta_G_H2O_for_298;//[cal] - Gibbs free energy
   of the reaction at 298.15 K

delta_H_rkn_1000 = delta_H_rkn_298 + integrate('
   2.459-2.0489*10^(-3)*T-2.817*10^(-7)*T^(2)
   +1.201*10^(-10)*T^(3)+1.723*10^(5)*T^(-2)','T',
   T_1,T_2);//[cal]

printf(" Standard enthalpy change of reaction at
   1000 K is %f cal\n\n",delta_H_rkn_1000);

//Now determining the standard entropy change of
   reaction at 298.15 K
delta_S_rkn_298 = (delta_H_rkn_298 - delta_G_rkn_298
   )/298.15;//[cal/K]
delta_S_rkn_1000 = delta_S_rkn_298 + integrate('
   (2.459-2.0489*10^(-3)*T-2.817*10^(-7)*T^(2)
   +1.201*10^(-10)*T^(3)+1.723*10^(5)*T^(-2))/T','T'
   ,T_1,T_2);//[cal/K]

//Therefore,the standard Gibbs free energy change of
    the reaction is given by,
delta_G_rkn_1000 = delta_H_rkn_1000 - 1000*
   delta_S_rkn_1000;//[cal]

printf(" Standard Gibbs free energy change of
   reaction at 1000 K is %f cal\n",delta_G_rkn_1000)
   ;
```

Scilab code Exa 6.10 Determination of standard enthalpy change and Gibbs free energy change

```
clear;
clc;

//Example - 6.10
//Page number - 228
printf("Example - 6.10 and Page number - 228\n\n");

//Given
T_1 = 298.15;//[K] - Standard temperature
T_2 = 1042;//[K] - Reaction temperature

//At 298.15 K
```

```
delta_H_CaCO3_for_298 = -289.5*10^(3);//[cal/mol] -
   Enthalpy of formation of CaCO3 at 298.15 K
delta_H_CaO_for_298 = -151.7*10^(3);//[cal/mol] -
   Enthalpy of formation of CaO at 298.15 K
delta_H_CO2_for_298 = -94.052*10^(3);//[cal/mol] -
   Enthalpy of formation of CO2 at 298.15 K
delta_G_CaCO3_for_298 = -270.8*10^(3);//[cal/mol] -
   Gibbs free energy change for formation of CaCO3
   at 298.15 K
delta_G_CaO_for_298 = -144.3*10^(3);//[cal/mol] -
   Gibbs free energy change for formation of CaO at
   298.15 K
delta_G_CO2_for_298 = -94.260*10^(3);//[cal/mol] -
   Gibbs free energy change for formation of CO2 at
   298.15 K

///Standaerd heat capacity data in cal/mol-K are
   given below, T is in K
//Cp_0_CO2 = 5.316 + 1.4285*10^(-2)*T -
   0.8362*10^(-5)*T^(2) + 1.784*10^(-9)*T^(3)
//Cp_0_CaO = 12.129 + 0.88*10^(-3)*T + 2.08*10^(5)*T
   ^(-2)
//Cp_0_CaCO3 = 24.98 + 5.240*10^(-3)*T +
   6.199*10^(5)*T^(-2)

//Therefore standard heat capacity of reaction is
   given by,
//Cp_0_rkn = Cp_0_CO2 + Cp_0_CaO - Cp_0_CaCO3
//On simplification,we get the relation
//Cp_0_rkn = -7.535 + 9.925*10^(-3)*T -
   0.8362*10^(-5)*T^(2) + 1.784*10^(-9)*T^(3) +
   4.119*10^(5)*T^(-2)

delta_H_rkn_298 = delta_H_CaO_for_298 +
   delta_H_CO2_for_298 - delta_H_CaCO3_for_298;//[
   cal] - Enthalpy of reaction at 298.15 K
delta_G_rkn_298 = delta_G_CaO_for_298 +
   delta_G_CO2_for_298 - delta_G_CaCO3_for_298;//[
   cal] - Gibbs free energy of the reaction at
   298.15 K

delta_H_rkn_1042 = delta_H_rkn_298 + integrate('
   -7.535+9.925*10^(-3)*T-0.8362*10^(-5)*T^(2)
   +1.784*10^(-9)*T^(3)+4.119*10^(5)*T^(-2)','T',T_1
   ,T_2);//[cal]

printf(" Standard enthalpy change of reaction at
   1042 K is %f cal\n\n",delta_H_rkn_1042);

//Now determining the standard entropy change of
   reaction at 298.15 K
delta_S_rkn_298 = (delta_H_rkn_298 - delta_G_rkn_298
   )/298.15;//[cal/K]
delta_S_rkn_1042 = delta_S_rkn_298 + integrate('
   (-7.535+9.925*10^(-3)*T-0.8362*10^(-5)*T^(2)
   +1.784*10^(-9)*T^(3)+4.119*10^(5)*T^(-2))/T','T',
   T_1,T_2);//[cal/K]

//Therefore,the standard Gibbs free energy change of
    the reaction is given by,
delta_G_rkn_1042 = delta_H_rkn_1042 - 1042*
   delta_S_rkn_1042;//[cal]

printf(" Standard Gibbs free energy change of
   reaction at 1042 K is %f cal",delta_G_rkn_1042);
```

Chapter 7

Thermodynamic property relations of pure substance

Scilab code Exa 7.1 Proving a mathematical relation

```
clear;
clc;

//Example - 7.1
//Page number - 235
printf("Example - 7.1 and Page number - 235\n\n");

//This problem involves proving a relation in which
  no numerical components are involved.
//For prove refer to this example 7.1 on page number
    235 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
  );
printf(" For prove refer to this example 7.1 on page
    number 235 of the book.");
```

Scilab code Exa 7.2 Proving a mathematical relation

```
clear;
clc;

//Example - 7.2
//Page number - 236
printf("Example - 7.2 and Page number - 236\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.2 on page number
    236 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.2 on page
    number 236 of the book.")
```

Scilab code Exa 7.3 Proving a mathematical relation

```
clear;
clc;

//Example - 7.3
//Page number - 236
printf("Example - 7.2 and Page number - 236\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.3 on page number
    236 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
```

```
printf(" For prove refer to this example 7.3 on page
    number 236 of the book.")
```

Scilab code Exa 7.4 Proving a mathematical relation

```
clear;
clc;

//Example - 7.4
//Page number - 240
printf("Example - 7.4 and Page number - 240\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.4 on page number
    240 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.4 on page
    number 240 of the book.")
```

Scilab code Exa 7.5 Proving a mathematical relation

```
clear;
clc;

//Example - 7.5
//Page number - 240
printf("Example - 7.5 and Page number - 240\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
```

```
//For prove refer to this example 7.5 on page number
    240 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.5 on page
    number 240 of the book.");
```

Scilab code Exa 7.6 Estimation of entropy change

```
clear;
clc;

//Example - 7.6
//Page number - 241
printf("Example - 7.6 and Page number - 241\n\n");

//Given
P_1 = 1;//[MPa] - Initial pressure
P_2 = 1.4;//[MPa] - Final pressure

//We know that
// dS = (Cp/T)*dT - (dV/dT)*dP
// Along an isothermal path,integration of above
   expression between states 1 and 2 yields
// S_2 - S_1 = - integral((dV/dT)*dP)_P
// An estimate can be made by assuming that (dV/dT)
   _P  remains constant over the range of pressure
   from P_1 to P_2 and evaluating the derivative at
   average pressure of 1.2 MPa
P_avg = P_2;
// S_2 - S_1 =-integral((dV/dT)*dP)_Pavg*(P_2 - P_1
   )
// (dV/dT)_P=1.2MPa = ((V_350 - V_250)/(350 - 250))
dV_dT = (0.2345 - 0.19234)/100;//[m^(3)/kg-K]
```

```
//Therefore
delta_S = -dV_dT*(P_2 - P_1)*1000;//[kJ/kg-K] -
    Entropy change

printf("The change in entropy is given by\n S_2-S_1
    = %f kJ/kg-K",delta_S);

//Let us compare this tabulated values. At 1MPA and
    300 C, S_1 = 7.1229 kJ/kg-K. At 1.4 MPa and 300 C
    , S_2 = 6.9534 kJ/kg-K.
//Therefore  S_2 - S_1 = -0.1695 kJ/kg-K
```

Scilab code Exa 7.7 Determination of work done

```
clear;
clc;

//Example - 7.7
//Page number - 241
printf("Example - 7.7 and Page number - 241\n\n");

//Given
T = 25 + 273.15;//[K] - Temperature of the
    surrounding
P_1 = 1;//[atm] - Initial pressure
P_2 = 1000;//[atm] - Final pressure

// V = 18.066 - 7.15*10^(-4)*P + 4.6*10^(-8)*P^(2)
       where, V is in 'cm^(3)/mol' and P is in 'atm
// (dV/dT)_P = 0.0045 + 1.4*10^(-6)*P    ;//cm^(3)/
    mol-K

// The work done by 1 mol is given by W = integral(P
    *dV)
// Differentiating both sides of the expression for
```

```
   V, we get
// dV = -7.15*10^(-4)*dP + 9.2*10^(-8)*(P*dP)
// P*dV = -7.15*10^(-4)*P*dP + 9.2*10^(-8)*(P^(2)*dP
   )

// The work done is given by
W = integrate('-7.15*10^(-4)*P + 9.2*10^(-8)*(P^(2))
   ','P',P_1,P_2);//[atm-cm^(3)/mol]
W = W*101325*10^(-6);//[J/mol]

printf("The necessary work to compress water from 1
   atm to 1000 atm is  %f J/mol\n\n",W);

//Let us calculate the amount of heat transfer
// q = integral(T*dS)
// dS = ((ds/dT)_P)*dT + ((dS/dP)_T)*dP
// Since the temperature is constant the first term
   is zero and
// dS = ((dS/dP)_T)*dP = -((dV/dT)_P)*dP
// Thus, q = integral(T*dS) = T*(integral(dS))      (
    as temperature is constant )
// or, q = -T*(integral((dV/dT)_P)*dP)

// Thus the heat transfer is given by
q = -T*integrate('0.0045+1.4*10^(-6)*P','P',P_1,P_2)
   ;//[atm-cm^(3)/mol]
q = q*101325*10^(-6);//[J/mol]

// q - W = delta_U
// Thus delta_U is given by
delta_U = q - W;//[J/mol]

printf("The change in internal energy is  %f J/mol",
   delta_U);
```

Scilab code Exa 7.8 Proving a mathematical relation

```
clear;
clc;

//Example - 7.8
//Page number - 243
printf("Example - 7.8 and Page number - 243\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.8 on page number
    243 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.8 on page
    number 243 of the book.")
```

Scilab code Exa 7.9 Proving a mathematical relation

```
clear;
clc;

//Example - 7.9
//Page number - 244
printf("Example - 7.9 and Page number - 244\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.9 on page number
    244 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
```

```
printf(" For prove refer to this example 7.9 on page
    number 244 of the book.");
```

Scilab code Exa 7.10 Proving a mathematical relation

```
clear;
clc;

//Example - 7.10
//Page number - 245
printf("Example - 7.10 and Page number - 245\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.10 on page
   number 245 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.10 on
   page number 245 of the book.");
```

Scilab code Exa 7.11 Proving a mathematical relation

```
clear;
clc;

//Example - 7.11
//Page number - 246
printf("Example - 7.11 and Page number - 246\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
```

```
//For prove refer to this example 7.11 on page
   number 246 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.11 on
   page number 246 of the book.");
```

Scilab code Exa 7.12 Evaluation of beta and K for nitrogen gas

```
clear;
clc;

//Example - 7.12
//Page number - 247
printf("Example - 7.9 and Page number - 244\n\n");

//given
T = 25+273.15;//[K] - Temperature
P = 1;//[atm] - Pressure
P = P*101325;//[Pa]
Tc = 126.2;//[K] - Critical temperature
Pc = 34;//[bar] - Critical pressure
Pc = Pc*10^(5);//[Pa]
R=8.314;//[J/mol*K] - Universal gas constant

a = (27*R^(2)*Tc^(2)/(64*Pc));//[Pa-m^(6)/mol^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]


// the cubic form of van der Walls equation of state
    is
// V^(3)-(b+(R*T)/P)*V^(2)+(a/P)*V-(a*b)/P=0
//Solving the cubic equation
deff('[y]=f(V)','y=V^(3)-(b+(R*T)/P)*V^(2)+(a/P)*V-(
    a*b)/P');
V = fsolve(1,f);
//The above equation has 1 real and 2 imaginary
    roots. We consider only real root.

Beta = R/((P*V)-(a/V)+((2*a*b)/V^(2)));//[K^(-1)]

K_t = (V-b)/((P*V)-(a/V)+((2*a*b)/V^(2)));//[Pa^(-1)
    ]
K_t = K_t*101325;//[atm^(-1)]

printf(" Beta\t = \t %f K^(-1)\n",Beta);
printf(" K_t\t = \t %f atm^(-1)",K_t);

//For ideal gas, Beta = 1/T = 0.0033354 K^(-1)
    and K_t = 1/P = 1 atm^(-1)
// So results obtained are convergent with those
    obtained assuming ideal gas.
```

Scilab code Exa 7.13 Calculation of temperature change and entropy change of water

```
clear;
clc;

//Example - 7.13
//Page number - 248
printf("Example - 7.13 and Page number - 248\n\n");

//Given
T = 45+273.15;//[K]
P_1 = 10;//[kPa] - Initial pressure
P_2 = 8600;//[kPa] - Final pressure
V = 1010;//[cm^(3)/kg] - Specific volume for
    saturated liquid water at 45 C
```

```
V = V*10^(-6);//[m^(3)/kg]
// Beta = (1/V)*(dV/dT)_P
Beta = 4.25*10^(-4);//[k^(-1)]
Cp = 4.178;//[kJ/kg-K] - Specific heat at constant
   pressure
eff = 0.75;// Efficiency of the pump

//(1)
//when efficiency of the pump is 100% , W = -
   delta_Hs
// Now delta_H = T*dS + V*dP, therefore under
   isentropic conditions , dH = V*dP
// Since the fluid is liquid , therefore the specific
    volume can be taken to be constant and
   integrating the above equaton we get
// delta_Hs = V*dP
delta_Hs = V*(P_2 - P_1);//[kJ/kg]

//Actual pumps are not isentropic and therefore not
   100% efficient. Therefore actual work done by the
    pump is given by
W = -delta_Hs/eff;//[kJ/kg]

printf(" (1).The work done by the pump is %f kJ/kg\n
   ",W);

//(2)
// We know that dH = Cp*dT + (1 - Beta*T)*V*dP
// Beta and V are weak functions of pressure in the
   case of liquids.
// Integrating the above equation we get
// delta_H = Cp*delta_T + (1 - Beta*T)*V*(delta_P)
// Now from energy balance delta_H = q - W . But q =
    0. Therefore ,
delta_H = -W;//[kJ/kg]
// Solving for delta_T
delta_T = (delta_H - (1 - Beta*T)*V*(P_2-P_1))/Cp;

```

```
printf(" (2).The temperature of water change by,
   delta_T = %f K\n",delta_T);

//(3)
T_1 = T;//[K]
T_2 = T + delta_T;//[K]
// dS = (Cp/T)*dT - Beta*V*dP
// Beta and V are weak functions of pressure in the
   case of liquids. Integrating the above equation
   we get
delta_S = Cp*log(T_2/T_1) - Beta*V*(P_2-P_1);//[kJ/
   kg-K]

printf(" (3).The entropy change of water is given by
    delta_S = %f kJ/kg-K",delta_S);
```

Scilab code Exa 7.14 Estimation of change in entropy and enthalpy

```
clear;
clc;

//Example - 7.14
//Page number - 249
printf("Example - 7.14 and Page number - 249\n\n");

//Given
T = 270;//[K]
P_1 = 381;//[kPa] - Initial pressure
P_2 = 1200;//[kPa] - Final pressure
V_liq = 1.55*10^(-3);//[m^(3)/kg] - Specific volume
   for saturated water in liquid phase at 270 C
Beta = 2.095*10^(-3);//[K^(-1)]

//dH = Cp*dT + [V - T*(dV/dT)_P]*dP
// dS = (Cp/T)*dT - ((dV/dT)_P)*dP
// Since isothermal conditions are maintained we get
// dH = [V - T*(dV/dT)_P]*dP = V*(1 - Beta*T)*dP
// For the liquid assuming V and Beta to remain
   constant during pressure change, and since
   temperature is constant we get
delta_H = V_liq*(1 - Beta*T)*(P_2 - P_1);//[kJ/kg]

printf("The enthalpy change is given by delta_H = %f
    kJ/kg\n",delta_H);

// Under isothermal conditions
// dS = -((dV/dT)_P)*dP = -Beta*V_liq*dP
// If Beta*V is assumed to remain constant during
   pressure change we get
delta_S = -Beta*V_liq*(P_2-P_1);//[kJ/kg-K]

printf("The entropy change is given by delta_S = %e
   kJ/kg-K",delta_S);
```

Scilab code Exa 7.15 Calculation of percentage change in volume

```
clear;
clc;

//Example - 7.15
//Page number - 249
printf("Example - 7.15 and Page number - 249\n\n");

//Given
T_1 = 0;//[C] - Initial tempetaure
T_2 = 100;//[C] - Final temperature
// Beta = 1.0414*10^(-3) + 1.5672*10^(-6)*T +
   5.148*10^(-8)*T^(2),   where T is in C
// At constant pressure  (1/V)*(dV/dT) = Beta
// or, d(log(V)) = Beta*dT
// Integrating  we get log(V_2/V_1) = integral(Beta*
   dT) from limit T_1 to T_2
integral = integrate('1.0414*10^(-3)+1.5672*10^(-6)*
   T+5.148*10^(-8)*T^(2)','T',T_1,T_2);

// log(V_2/V_1) = integral
// (V_2/V_1) = exp(integral)
// (V_2 - V_1)/V_1 = change = exp(integral) - 1;
change = exp(integral) - 1;
per_change = 100*change;

printf("The percentage change in volume = %f %%",
   per_change);
```

Scilab code Exa 7.16 Determination of enthalpy and entropy change

```
clear;
clc;

//Example - 7.16
//Page number - 250
printf("Example - 7.16 and Page number - 250\n\n");

//Given
T_1 = 25 + 273.15;//[C] - Initial tempetaure
T_2 = 50 + 273.15;//[C] - Final temperature
P_1 = 1;//[bar] - Initial pressure
P_2 = 1000;//[bar] - Final pressure

Cp_T1_P1 = 75.305;//[J/mol-K]
Cp_T2_P1 = 75.314;//[J/mol-K]
V_T1_P1 = 18.071;//[cm^(3)/mol]
V_T1_P2 = 18.012;//[cm^(3)/mol]
V_T2_P1 = 18.234;//[cm^(3)/mol]
V_T2_P2 = 18.174;//[cm^(3)/mol]
```

```
Beta_T1_P1 = 256*10^(-6);//[K^(-1)]
Beta_T1_P2 = 366*10^(-6);//[K^(-1)]
Beta_T2_P1 = 458*10^(-6);//[K^(-1)]
Beta_T2_P2 = 568*10^(-6);//[K^(-1)]

// The entropy change is given by
// dS = (Cp/T)*dT - ((dV/dT)_P)*dP
// The mean Cp between 25 and 50 C is
Cp_mean = (Cp_T1_P1 + Cp_T1_P1)/2;//[J/mol-K]


// (dV/dT)_P=1bar = (V_T2_P1 - V_T1_P1)/(50 - 25)
dV_dT_P1 = ((V_T2_P1 - V_T1_P1)/(50 - 25))*10^(-6);
    //[m^(-3)/mol-K]
dV_dT_P2 = ((V_T2_P2 - V_T1_P2)/(50 - 25))*10^(-6);
    //[m^(-3)/mol-K]
// The mean value of (dV/dT)_P between 1 and 1000
    bar is
dV_dT_mean = (dV_dT_P1 + dV_dT_P2)/2;//[m^(-3)/mol-K
    ]
delta_S = Cp_mean*log(T_2/T_1) - dV_dT_mean*(P_2 -
    P_1)*10^(5);//[J/mol-K]

printf(" The value of entropy change is given by,
    delta_S = %f J/mol-K\n",delta_S);

// Now let us determine the enthalpy change. We know
     that
// dH = Cp*dT + [V - T*(dV/dT)_P]*dP
// [V - T*(dV/dT)_P] = (V - T*V*Beta) = val (say)
// At state 1
val_1 = ((V_T1_P1)*10^(-6))*(1 - (T_1)*(Beta_T1_P1))
    ;//[m^(3)/mol]
// At state 2
val_2 = ((V_T2_P2)*10^(-6))*(1 - (T_2)*(Beta_T2_P2))
    ;//[m^(3)/mol]
val_mean = (val_1 + val_2)/2;//[m^(3)/mol]

```

```
delta_H = Cp_mean*(T_2 - T_1) + val_mean*(P_2-P_1)
   *10^(5);//[J/mol]

printf(" The value of enthalpy change is given by,
   delta_H = %f J/mol",delta_H);
```

Scilab code Exa 7.17 Proving a mathematical relation

```
clear;
clc;

//Example - 7.17
//Page number - 253
printf("Example - 7.17 and Page number - 253\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.17 on page
   number 253 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.17 on
   page number 253 of the book.");
```

Scilab code Exa 7.18 Proving a mathematical relation

```
clear;
clc;

//Example - 7.18
//Page number - 253
printf("Example - 7.18 and Page number - 253\n\n");
```

```

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.18 on page
   number 253 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.18 on
   page number 253 of the book.");
```

Scilab code Exa 7.19 Proving a mathematical relation

```
clear;
clc;

//Example - 7.19
//Page number - 254
printf("Example - 7.19 and Page number - 254\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.19 on page
   number 254 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.19 on
   page number 254 of the book.");
```

Scilab code Exa 7.20 Proving a mathematical relation

```
clear;
```

```
clc;

//Example - 7.20
//Page number - 254
printf("Example - 7.20 and Page number - 254\n\n");

//This problem involves proving a relation in which
    no numerical components are involved.
//For prove refer to this example 7.20 on page
    number 254 of the book.
printf(" This problem involves proving a relation in
     which no numerical components are involved.\n\n"
    );
printf(" For prove refer to this example 7.20 on
    page number 254 of the book.");
```

Scilab code Exa 7.21 Proving a mathematical relation

```
clear;
clc;

//Example - 7.21
//Page number - 255
printf("Example - 7.21 and Page number - 255\n\n");

//This problem involves proving a relation in which
    no numerical components are involved.
//For prove refer to this example 7.21 on page
    number 255 of the book.
printf(" This problem involves proving a relation in
     which no numerical components are involved.\n\n"
    );
printf(" For prove refer to this example 7.21 on
    page number 255 of the book.");
```

Scilab code Exa 7.22 Calculation of volume expansivity and isothermal compressibility

```
clear;
clc;

//Example - 7.22
//Page number - 256
printf("Example - 7.22 and Page number - 256\n\n");

// Given
T = 100 + 273.15;//[K]
P = 10;//[MPa]

// The volume expansivity is defined as
// Beta = (1/V)*(del V/del T)_P = (1/V)*(dV/dT)_P
// From compressed liquid water tables at 100 C and
   10 MPa,
V = 0.0010385;//[m(3)/kg]
Beta = (1/V)*((0.0010549 - 0.0010245)/(120 - 80));//
   [K^(-1)] // The values are obtained from the
   steam table as reported in the book.

printf("The value of volume expansivity is  Beta =
   %e K^(-1)\n",Beta);

//Isothermal compressibility is defined as
// K_t = -(1/V)*(del V/del T)_T = -(1/V)*(dV/dT)_T
K_t = -(1/V)*((0.0010361 - 0.0010410)/(15 - 5));//[
   MPa^(-1)]  // The values are obtained from the
   steam table as reported in the book.

K_t = K_t*10^(-3);//[kPa]

```

```
printf("The value of isothermal compressibility is
    K_t = %e kPa^(-1)\n",K_t);

// Cp - Cv = (T*V*(Beta^(2)))/K_t
R = (T*V*(Beta^(2)))/K_t;//[kJ/kg-K]

printf("The value of the difference between Cp and
    Cv is  Cp-Cv = %f kJ/kg-K",R);
```

Scilab code Exa 7.23 Estimation of specific heat capacity

```
clear;
clc;

//Example - 7.23
//Page number - 257
printf("Example - 7.23 and Page number - 257\n\n");

// Given
T = 300 + 273.15;//[K]
P = 4;//[MPa]

Cp_0 = 7.7 + 0.04594*10^(-2)*T + 0.2521*10^(-5)*T
    ^(2) - 0.8587*10^(-9)*T^(3);//[cal/mol-K]
Cp_0 = (Cp_0*4.186)/18.015;//[kJ/kg-K]

// Cp(T,P) = Cp_0(T,P=0) - T*integral((del^2 V/del T
    ^(2))_P)*dP from limit 0 to P
// Cp = Cp_0 - T*((del^2 V/del T^2)_Pavg)*(P_2 - P_1
    )

P_avg = (0+4)/2;//[MPa]

//Using finite difference we get  (del^2 V/del T^(2)
    ) = ((V_(T+delta T) - 2*V_T + V_(T-delta T))/(
   delta_T^(2))
//(del^2 V/del T^(2))_Pavg = (V_(350 C) + V_(250 C)
   - 2*V_(300 C))/(delta_T^(2)) = del_2 (say)
del_2 = (0.13857 + 0.11144 - 2*0.12547)/(50^(2));//[
   m^(3)/kg-K^2] // The values are obtained from the
    steam table as reported in the book.


Cp = Cp_0 - T*del_2*4000;//[kJ/kg-K]

printf(" The value of constant pressure specific
   heat capacity is , Cp = %f kJ/kg-K",Cp);

// At P = 4 MPa
// Cp = (del H/del T)_P = (H_350 C - H_250 C)/(350 -
    250.4)
// Cp = (3092.5 - 2801.4)/(350 - 250.4) = 2.923 [kJ/
   kg-K]
```

Scilab code Exa 7.24 Estimation of specific heat capacity

```
clear;
clc;

//Example - 7.24
//Page number - 257
printf("Example - 7.24 and Page number - 257\n\n");

// Given
T = 300 + 273.15;//[K]
P = 2.0;//[MPa]

// At 2 MPa and 250 C
H_1 = 2902.5;//[kJ/kg]
// At 2 MPa and 350 C
```

```
H_2 = 3137.0;//[kJ/kg]

Cp = (H_2 - H_1)/(350 - 250);//[kJ/kg-K]

printf(" The value of constant pressure specific
   heat capacity is , Cp = %f kJ/kg-K",Cp);
```

Scilab code Exa 7.25 Calculation of volume expansivity and isothermal compressibility

```
clear;
clc;

//Example - 7.25
//Page number - 258
printf("Example - 7.25 and Page number - 258\n\n");

// Given
T = 80 + 273.15;//[K]
P = 10;//[MPa]

// Beta = (1/V)*(del V/del T)_P

// Pressure is kept fixed at 10 MPa and (del V/del T
   )_P is evaluated. Looking in the compressed
   liquid water tables ,at
// At 80 C and 10 MPa
V_1 = 0.0010245;//[m^(3)/kg]
// At 60 C and 10 MPa
V_2 = 0.0010127;//[m^(3)/kg]
// At 100 C and 10 MPa
V_3 = 0.0010385;//[m^(3)/kg]

Beta = (1/V_1)*((V_3 - V_2)/(100 - 60));//[K^(-1)]

```

```
printf("The value of volume expansivity is  Beta =
   %e K^(-1)\n",Beta);

//Isothermal compressibility is given by
// K_t = -(1/V)*(del V/del P)_T

// Temperature is kept fixed at 80 C and different
   pressures are taken to calculate (del V/del P)_T
// At 80 C and 5 MPa
V_4 = 0.0010268;//[m^(3)/kg]
// At 80 C and 10 MPa
V_5 = 0.0010245;//[m^(3)/kg]
// At 80 C and 15 MPa
V_6 = 0.0010222;//[m^(3)/kg]

// K_t = -(1/V)*(del V/del T)_P
K_t = -(1/V_1)*((V_4 - V_6)/(5 - 15));//[MPa^(-1)]
K_t = K_t*10^(-6);//[Pa^(-1)]

printf("The value of isothermal compressibility is
   K_t = %e Pa^(-1)\n",K_t);

// Cp - Cv = (T*V*(Beta^(2)))/K_t
R = (T*V_1*(Beta^(2)))/K_t;//[J/kg-K]
R = R*10^(-3);//[kJ/kg-K]

printf("The value of the difference between Cp and
   Cv is  Cp-Cv = %f kJ/kg-K",R);
```

Scilab code Exa 7.26 Calculation of mean Joule Thomson coefficient

```
clear;
clc;

//Example - 7.26
```

```
//Page number - 260
printf("Example - 7.26 and Page number - 260\n\n");

// Given
P_1 = 150;//[bar]
P_2 = 1;//[bar]

T_1 = 300;//[K]
T_2 = 260;//[K]
T_3 = 280;//[K]
T_4 = 200;//[K]
T_5 = 120;//[K]
T_6 = 140;//[K]

H_P1_T1 = 271.8;//[kJ/kg]
H_P2_T2 = 260.0;//[kJ/kg]
H_P2_T3 = 280.2;//[kJ/kg]
H_P1_T4 = 129.2;//[kJ/kg]
H_P2_T5 = 118.8;//[kJ/kg]
H_P2_T6 = 139.1;//[kJ/kg]

//(a)
// During the Joule-Thomson expansion the enthalpy
   should remain constant
// Therefore at 1 bar the exit temperature is such
   that enthalpy is 271.8 kJ/kg
// The temperature at which enthalpy is 271.8 kJ/kg
   is given by,
T_new = ((H_P1_T1 - H_P2_T2)/(H_P2_T3 - H_P2_T2))*(
   T_3 - T_2) + T_2;//[K]

// Therefore Joule-Thomson coefficient is given by,
meu = (T_1 - T_new)/(P_1 - P_2);//[K/bar]

printf(" (a).The value of Joule-Thomson coefficient
   (for initial T = 300 K) is %f J/bar\n",meu);

//(b)
```

```
// During the Joule-Thomson expansion the enthalpy
   should remain constant
// Therefore at 1 bar the exit temperature is such
   that enthalpy is 129.2 kJ/kg
// The temperature at which enthalpy is 129.2 kJ/kg
   is given by,
T_new_prime = ((H_P1_T4 - H_P2_T5)/(H_P2_T6 -
   H_P2_T5))*(T_6 - T_5) + T_5;//[K]

// Therefore Joule-Thomson coefficient is given by,
meu_prime = (T_4 - T_new_prime)/(P_1 - P_2);//[K/bar
   ]

printf(" (b).The value of Joule-Thomson coefficient
   (for initial T = 200 K) is %f J/bar",meu_prime);

// Therefore the Joule-Thomson coefficient is higher
    for low initial temperatures and therefore the
   drop in temperature is more.
```

Scilab code Exa 7.27 Estimation of Joule Thomson coefficient

```
clear;
clc;

//Example - 7.27
//Page number - 261
printf("Example - 7.27 and Page number - 261\n\n");

//(a)
//This part involves proving a relation in which no
   numerical components are involved.
//For prove refer to this example 7.27 on page
   number 261 of the book.

```

```
//(b)
//This part involves proving a relation in which no
    numerical components are involved.
//For prove refer to this example 7.27 on page
    number 261 of the book.

//(c)
T = 300;//[K] - Temperature
P = 5;//[atm] - Pressure
P = P*101325;//[Pa]
Cp_0 = 35.78;//[J/mol-K] - Standard specific heat
    capacity at constant pressure
B = -50;//[cm^(3)/mol]
B = B*10^(-6);//[m^(3)/mol]

// (dB/dT) = 1.0 = dB_dT (say)
dB_dT = 1.0;//[cm^(3)/mol-K]
dB_dT = dB_dT*10^(-6);//[m^(3)/mol-K]

// (d^2 B/d T^2) = -0.01 = dB_dT_2 (say)
dB_dT_2 = -0.01;//[cm^(3)/mol-K^(2)]
dB_dT_2 = dB_dT_2*10^(-6);//[m^(3)/mol-K^(2)]

Cp = Cp_0 - P*T*(dB_dT_2);//[[J/mol-K]] - Specific
    heat capacity at constant pressure

//Therefore Joule-Thomson coefficient is given by,
meu = (1/Cp)*(-B + T*dB_dT);//[K/Pa]
meu = meu*10^(5);//[K/bar]

printf(" (c).The value of Joule-Thomson coefficient
    is %f J/bar",meu);
```

Scilab code Exa 7.28 Proving a mathematical relation

```
clear;
clc;

//Example - 7.28
//Page number - 262
printf("Example - 7.28 and Page number - 262\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.28 on page
   number 262 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.28 on
   page number 262 of the book.");
```

Scilab code Exa 7.29 Proving a mathematical relation

```
clear;
clc;

//Example - 7.29
//Page number - 263
printf("Example - 7.29 and Page number - 263\n\n");

//This problem involves proving a relation in which
   no numerical components are involved.
//For prove refer to this example 7.29 on page
   number 263 of the book.
printf(" This problem involves proving a relation in
    which no numerical components are involved.\n\n"
   );
printf(" For prove refer to this example 7.29 on
   page number 263 of the book.");
```

Scilab code Exa 7.30 Calculation of pressure

```
clear;
clc;

//Example - 7.30
//Page number - 267
printf("Example - 7.30 and Page number - 267\n\n");

//Given
den_liq = 13690;//[kg/m^(3)] - Density of liquid
   mercury
den_solid = 14190;//[kg/m^(3)] - Density of solid
   mercury
mp = -38.87;//[C] - Melting point of mercury at
   pressure of 1 bar
mp = mp + 273.15;//[K]
T_req = 0;//[C] - Required temperature to which the
   melting point is to be raised
T_req = T_req + 273.15;//[K]
H_fus = 11.62;//[kJ/kg] - Latent heat of fusion of
   mercury

V_liq = (1/den_liq);//[m^(3)/kg] - Specific volume
   of liquid mercury
V_solid = (1/den_solid);//[m^(3)/kg] - Specific
   volume of solid mercury

// (delta P/delta T) = ((P - 1)*100)/(T_req - mp)
// delta H/(T*delta V) = (H_liq - H_solid)/(T*(V_liq
    - V_solid)) = del (say)
del = (H_fus)/(mp*(V_liq - V_solid));//[kPa/K] -
   delta H/(T*delta V)

```

```
//Equating the two sides and then solving we get
P = (del*(T_req - mp))/100 + 1;//[bar]

printf(" The required pressure should be %f bar",P);
```

Scilab code Exa 7.31 Calculation of enthalpy change and entropy change

```
clear;
clc;

//Example - 7.31
//Page number - 268
printf("Example - 7.31 and Page number - 268\n\n");

// The clapeyron equation is
// (dP/dT)_sat = delta_H_fus/(T*delta_V_fus)

//(1)
// At 1 bar
// Considering the data given at pressure 1 and 1000
    bar, we have
delta_H_fus_1 = ((1000-1)*10^(5)*(273.15-22.6)
   *3.97*10^(-6))/(14.8+22.6);//[J/mol]
delta_S_fus_1 = delta_H_fus_1/(273.15-22.6);//[J/mol
   -K]

printf(" (1).The delta_H_fus at 1 bar is %f J/mol\n"
   ,delta_H_fus_1);
printf("      The delta_S_fus at 1 bar is %f J/mol-K\
   n\n",delta_S_fus_1);

//(2)
// At 6000 bar
T_mean = (128.8+173.6)/2;//[C] - Mean temperature
T_mean = T_mean + 273.15;//[K]
```

```
delta_V_fus_mean = (1.12+1.55)/2;//[cm^(3)/mol]

// Consider the data at pressure of 5000 and 7000
   bar we get ,
delta_H_fus_2 = ((7000-5000)*10^(5)*(T_mean*
   delta_V_fus_mean*10^(-6)))/(173.6-128.8);//[J/mol
   ]
delta_S_fus_2 = delta_H_fus_2/T_mean;//[J/mol-K]

printf(" (2).The delta_H_fus at 6000 bar is %f J/mol
   \n",delta_H_fus_2);
printf("     The delta_S_fus at 6000 bar is %f J/mol
   -K\n\n",delta_S_fus_2);
```

Scilab code Exa 7.32 Estimation of ratio of temperature change and pressure change

```
clear;
clc;

//Example - 7.32
//Page number - 268
printf("Example - 7.32 and Page number - 268\n\n");

//Given
H_fus = 80;//[cal/g] - Heat of fusion at 0 C and 1
   atm pressure
T = 0+273.15;//[K] - Temperature
vol_ratio = 1.091;// Ratio of the specific volume of
    ice and water.
sp_vol = 0.001;//[m^(3)/kg] - Specific volume of
   saturated liquid water.

// The clapeyron equation can be written as
// (dP/dT)_sat = T*delta V_LS/(delta H_LS) = (T*(
   V_ice - V_water))/(H_ice - H_water)
dP_dT = (T*(vol_ratio - 1)*10^(-3))/(-H_fus*4.186);
   //[K/kPa]

printf("The value of (dT/dP)_sat is %e K/kPa",dP_dT)
   ;
```

Scilab code Exa 7.33 Determination of boiling point of water

```
clear;
clc;

//Example - 7.33
//Page number - 268
printf("Example - 7.33 and Page number - 268\n\n");

//Given
P = 2;//[atm] - Surrounding pressure
bp_water = 100 + 273.15;//[K] - Boiling point of
    water at 1 atm pressure
delta_H_vap = 2257;//[kJ/kg] - Enthalpy of
    vaporization
delta_H_vap = delta_H_vap*18.015;//[J/mol]
R = 8.314;//[J/mol*K] - Universal gas constant

// The clapeyron equation is given by
// log(P_2_sat/P_1_sat) = (-delta H_vap/R)*(1/T_2 -
    1/T_1)
P_1_sat = 1;//[atm]
P_2_sat = P;
T_1 = bp_water;

// Solving the above equation
T_2 = 1/((log(P_2_sat/P_1_sat))/(-delta_H_vap/R) +
    (1/T_1));//[K]
```

```
T_2 = T_2 - 273.15;//[C]

printf(" The boiling point of water at a pressure of
    2 atm is %f C",T_2);
```

Scilab code Exa 7.34 Calculation of enthalpy and entropy of vaporization of water

```
clear;
clc;

//Example - 7.34
//Page number - 269
printf("Example - 7.34 and Page number - 269\n\n");

//Given
T_1 = 0.01 +273.15;//[K]
T_2 = 1 + 273.15;//[K]
P_sat_1 = 0.611;//[kPa] - P_sat at temperature T_1
P_sat_2 = 0.657;//[kPa] - P_sat at temperature T_2
Mol_wt = 18.015;//[g/mol] - Molecular weight of
   water
R = 8.314;//[J/mol*K] - Universal gas constant

// The clapeyron equation is given by
// log(P_sat_2/P_sat_1) = (-delta H_LV/R)*(1/T_2 -
   1/T_1)

// Solving the above equation
delta_H = -(log(P_sat_2/P_sat_1)/(1/T_2 - 1/T_1))*R;
   //[J/mol]
delta_H = delta_H/Mol_wt;//[kJ/kg]

printf(" The enthalpy of vaporization is %f kJ/kg\n"
   ,delta_H);
```

```

// Entropy of vaporization is given by
S_vap = delta_H/T_2;//[kJ/kg-K]
printf(" The entropy of vaporization is %f kJ/kg-K",
   S_vap);
```

Scilab code Exa 7.35 Estimation of heat of vaporization of water

```
clear;
clc;

//Example - 7.35
//Page number - 269
printf("Example - 7.35 and Page number - 269\n\n");

//Given
T = 100 + 273.15;//[K]
// (dT/dP)_sat = (1/27.12) K/mm
dT_dP = (1/27.12);//[K/mm]
dT_dP = dT_dP*(760/101325);//[K/Pa]

// The clapeyron equation is given by
// (dP/dT)_sat = (-delta H_LV)/(T*delta V_LV)
// delta H_LV = T*delta V_LV*(dP/dT)_sat

// (dP/dT)_sat = 1/(dT/dP)_sat
dP_dT = 1/dT_dP;//[Pa/K]

// From saturated steam table at 100 C
V_vap = 1.6729;//[m^(3)/kg]
V_liq = 0.001044;//[m^(3)/kg]
delta_V = V_vap - V_liq;//[m^(3)/kg]

// Therefore delta_H_LV is given by
delta_H_LV = T*delta_V*(dP_dT);//[J/kg]
delta_H_LV = delta_H_LV*10^(-3);//[kJ/kg]

printf(" The heat of vaporization of water is %f kJ/
   kg\n",delta_H_LV);
```

Scilab code Exa 7.36 Calculation of latent heat of vaporization

```
clear;
clc;

//Example - 7.36
//Page number - 270
printf("Example - 7.36 and Page number - 270\n\n");

//Given
T_1 = 100 + 273.15;//[K]
P_1 = 1.01325;//[bar]
T_2 = 98 + 273.15;//[K]
P_2 = 0.943;//[bar]
V_vap = 1.789;//[m^(3)] - Volume in vapour phase
vessel_vol = 1.673;//[m^(3)] - Volume of the vessel
R = 8.314;//[J/mol*K] - Universal gas constant

// The total volume remains constant as the walls
   are rigid. At 98 C we get
// vessel_vol = V_liq*(1 - x) + V_vap*x
// Since V_liq is negligible as compared to V_vap,
   therefore
x = vessel_vol/V_vap;

// The quantity is given by x = m_vap/(m_liq + m_vap
   )
// Since (m_liq + m_vap) = 1, therefore at 98 C
   saturated vapour is x and saturated liquid is (1
   - x)
```

```
m_vap = x;//[kg] - Mass of saturated vapour
m_liq = (1 - x);//[kg] - Mass of saturated liquid

printf(" The amount of vapour condensed is %f kg\n",
   m_liq);

// The clapeyron equation is given by
// log(P_2_sat/P_1_sat) = (-delta H_LV/R)*(1/T_2 -
   1/T_1)

// Solving the above equation
delta_H = -(log(P_2/P_1)/(1/T_2 - 1/T_1))*R;
delta_H = delta_H/18.015;//[kJ/kg]

printf(" The latent heat of vaporization is %f kJ/kg
   \n",delta_H);
```

Scilab code Exa 7.37 Determination of temperature dependence

```
clear;
clc;

//Example - 7.37
//Page number - 270
printf("Example - 7.37 and Page number - 270\n\n");

//Given
T_1 = 298.15;//[K] - Standard reaction temperature
delta_H_gas = -52.23;//[kcal/mol] - Enthalpy of
   formation of C2H5OH(gas)
delta_H_liq = -66.35;//[kcal/mol] - Enthalpy of
   formation of C2H5OH(liq)

// For ethanol(g) [T is in K and Cp_0 in cal/mol-K]
// Cp_0 = 4.75 + 5.006*10^(-2)*T - 2.479*10^(-5)*T
```

```
    ^(2) + 4.79*10^(-9)*T^(3)

// For ethanol(l) [T is in K and Cp_0 in cal/mol-K]
// Cp_0 = 67.29 - 0.343*T - 6.94*10^(-4)*T^(2)

// The vaporization of a liquid can be written as
   C2H5OH(liq) - C2H5OH(gas)
// Since the pressure is 1 atm therefore the
   standard data can be used
delta_H_298 = delta_H_gas - delta_H_liq;//[kcal/mol]
delta_H_298 = delta_H_298*1000;//[cal/mol]
delta_a = 4.75 - 67.29;
delta_b = 5.006*10^(-2) - (-0.343);
delta_c = -2.479*10^(-5) - 6.94*10^(-4);
delta_d = 4.79*10^(-9);

// The standard enthalpy of vaporization at a
   temperature T is given by
// delta_H_T =  delta_H_298 + integrate('delta_a +
   delta_b*T + delta_c*T^(2) + delta_d*T^(3)','T',
   T_1,T);//[cal/mol]

// Therefore the standard enthalpy of vaporization
   at a temperature T = 283 K is given by
T_2 = 283;//[K]
delta_H_283 =  delta_H_298 + integrate('delta_a+
   delta_b*T+delta_c*T^(2)+delta_d*T^(3)','T',T_1,
   T_2);//[cal/mol]

// Therefore the standard enthalpy of vaporization
   at a temperature T = 348 K is given by
T_3 = 348;//[K]
delta_H_348 =  delta_H_298 + integrate('delta_a+
   delta_b*T+delta_c*T^(2)+delta_d*T^(3)','T',T_1,
   T_3);//[cal/mol]

// From the values of standard enthalpy of
   vaporization obtained above at 283, 298, and 348
   K, it is clear that enthalpy decreases with
   increase in temperature
printf(" The value of enthalpy of vaporization at
   283 K is %f cal/mol\n",delta_H_283);
printf(" The value of enthalpy of vaporization at
   298.15 K is %f cal/mol\n",delta_H_298);
printf(" The value of enthalpy of vaporization at
   348 K is %f cal/mol\n",delta_H_348);
printf(" Therefore standard enthalpy of vaporization
    decrease with the increase in temperature\n\n");

// Solving the above equatio manually we get,
// delta_H_vap = 1.1975*10^(-9)*T^(4) -
   2.396*10^(-4)*T^(3) + 0.1965*T^(2) - 62.54*T +
   21639.54
// Solving for 'T' at which 'delta_H_vap' = 0
deff('[y]=f(T)','y=1.1975*10^(-9)*T^(4)
   -2.396*10^(-4)*T^(3)+0.1965*T^(2)-62.54*T +
   21639.54');
T_0 = fsolve(500,f);//[J/mol]

// We know that at critical point (critical
   temperature and critical pressure) the enthalpy
   of vaporization is zero.
// Here we have made the standard enthalpy of
   vaporization equal to zero which is at standard
   pressure of 1 atm.
// Therefore following conclusions can be drawn
printf(" The temperature obtained by equating
   standard enthalpy of vaporization equal to zero
   is %f K\n",T_0);
printf(" But the critical temperature of ethanol is
   513.9 K, which is far from the temperature
   obtained above\n")
printf(" Therefore the temperature obtained by
   equating standard enthalpy of vaporization equal
   to zero is not the critical temperature")
```

Scilab code Exa 7.38 Calculation of fugacity of water

```
clear;
clc;

//Example - 7.38
//Page number - 276
printf("Example - 7.38 and Page number - 276\n\n");

//Given
T = 300 + 273.15;//[K] - Temperature
P = 9000;//[kPa] - Pressure
P_sat = 8592.7;//[kPa] - Vapour pressure of
    saturated water at 300 C
f_sat = 6738.9;//[kPa] - Fugacity of saturated water
     at 300 C
V_liq = 25.28;//[cm^(3)/mol] - Molar volume of water
     in liquid phase
V_liq = V_liq*10^(-6);// [m^(3)/mol]
V_vap = 391.1;//[cm^(3)/mol] - Molar volume of water
     in vapour phase
V_vap = V_vap*10^(-6);// [m^(3)/mol]
R = 8.314;//[J/mol*K] - Universal gas constant

// At 300 C and 9000 kPa water is a compressed
    liquid and its fugacity is given by
// f = f_sat*exp[V_liq*(P - P_sat)/R*T]
fugacity = f_sat*exp((V_liq*(P - P_sat)*1000)/(R*T))
    ;

printf(" The fugacity of water at 9000 kPa is %f kPa
    ",fugacity);
```

Scilab code Exa 7.39 Estimation of fugacity of saturated steam

```
clear;
clc;

//Example - 7.39
//Page number - 276
printf("Example - 7.39 and Page number - 276\n\n");

//Given
T = 200 + 273.15;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant

// From steam table at 200 C as reported in the book
P_sat = 1.5538;//[MPa] - Vapour pressure of
    saturated steam
H_vap = 2793.2;//[kJ/kg] - Enthalpy of saturated
    steam in vapour phase
S_vap = 6.4323;//[kJ/kg-K] - Entropy of saturated
    steam in vapour phase
G_sat = H_vap - T*S_vap;//[kJ/kg] - Gibbs free
    energy
G_sat = G_sat*18.015;//[J/mol]

// Now let us calculate the Gibbs free energy at the
     lowest pressure available in superheated steam
    tables at 200 C
// At 200 C and 0.01 MPa as reported in the book
H = 2879.5;//[kJ/kg] - Enthalpy
S = 8.9038;//[kJ/kg-K] - Entropy
G_ig = H - T*S;//[kJ/kg] - Gibbs free energy
G_ig = G_ig*18.015;//[J/mol]

// Integrating from ideal gas state at 200 C and
```

```
    0.01 MPa to saturated vapour at 200 C we get
// G_sat - G_ig = R*T*log(f_sat/f_ig)

// Under the ideal gas condition the pressure is
    small therefore f_ig = P = 0.01 MPa
f_ig = 0.01;//[MPa]

// Solving the above equation
f_sat = f_ig*(exp((G_sat - G_ig)/(R*T)));//[MPa]

printf(" The fugacity of saturated steam at 200 C is
    %f MPa",f_sat);
```

Scilab code Exa 7.40 Estimation of fugacity of steam

```
clear;
clc;

//Example - 7.40
//Page number - 277
printf("Example - 7.40 and Page number - 277\n\n");

// Given
T = 320 + 273.15;//[K]
P_1 = 70;//[bar]
P_2 = 170;//[bar]
R = 8.314;//[J/mol*K] - Universal gas constant

//(a)
// dG = R*T*dlog(f)
// G - G_ig = R*T*log(f/f_ig)

// From steam table the low pressure that is
    available is 1 kPa.
f_ig = 1;//[kPa] - Assuming ideal gas behaviour at
```

```
    such low pressure

// At 1 kPa (under saturated conditions)
P_sat = 112.891;//[bar]
// Therefore at both 1 kPa and 70 bar the stem is
    superheated and byond a pressure of 112.891 bar
    it is compressed liquid.

// For superheated steam table at 1 kPa and 320 C,
    as repoted in the book
H_1 = 3117.08;//[kJ/kg] - Enthalpy
S_1 = 10.41232;//[kJ/kg-K] - Entropy

// For superheated steam table at 70 bar and 320 C,
    as repoted in the book
H_2 = 2916.92;//[kJ/kg] - Enthalpy
S_2 = 6.0651;//[kJ/kg-K] - Entropy

// At 70 bar and 320 C,
G = H_2 - T*S_2;//[kJ/kg] - Gibbs free energy
// At 1 kPa and 320 C
G_ig = H_1 - T*S_1;//[kJ/kg] - Gibbs free energy

// log(f/f_ig) = (G - G_ig)/(R*T)
f = f_ig*(exp((G - G_ig)*18/(R*T)));//[kPa]
f = f*10^(-2);//[bar]

// At 70 bar
phi = f/P_1;

printf(" (a).The fugacity of steam at 320 C and 70
    bar is %f bar\n",f);
printf("      The fugacity coefficient at 320 C and
    70 bar is , phi = %f\n\n",phi);

//(b)
// Now consider saturated steam at 320 C. We have
P_sat = 112.891;//[bar]
```

```
V_liquid = 1.5;//[cm^(3)/mol] - Molar vlume of
    saturated liquid
V_liquid = V_liquid*10^(-6);//[m^(3)/mol]
V_vapour = 15.48;//[cm^(3)/mol] - Molar vlume of
    saturated vapour
U_liqid = 1445.7;//[Kj/kg] - Internal energy of
    satuarted liquid
U_vapour = 2528.9;//[kJ/kg] - Internal energy of
    satuarted vapour
H_liquid = 1462.6;//[kJ/kg] - Enthalpy of saturated
    liquid
H_vapour = 2703.7;//[kJ/kg] - Enthalpy of saturated
    vapour
S_liquid = 3.45;//[kJ/kg-K]  - Entropy of saturated
    liquid
S_vapour = 5.5423;//[kJ/kg-K] - Entropy of saturated
     vapour

// Now let us calculate Gibbs free energy of
    saturated liquid and saturated vapour
G_liquid = H_liquid - T*S_liquid;//[kJ/kg]
G_vapour = H_vapour - T*S_vapour;//[kJ/kg]
// Note that under saturated conditions
// G_sat = G_liquid = G_vapour
G_sat = G_liquid;//[kJ/kg]

// log(f_sat/f_ig) = (G_sat - G_ig)/(R*T)
f_sat = f_ig*(exp((G_sat - G_ig)*18/(R*T)));//[kPa]
f_sat = f_sat*10^(-2);//[bar]

phi_sat = f_sat/P_sat;

// And now the fugacity is to be determined at 320 C
     and P = 170 bar. We know the following relation
    for compressed liquid.
// f_CL = f_sat*exp(V_liquid*(P-P_sat)/(R*T))
f_CL = f_sat*exp(V_liquid*18*(P_2-P_sat)*10^(5)/(R*T
    ));//[bar]
```

```

// Therefore the fugacity coefficient at 170 bar and
    320 C is given by
phi_2 = f_CL/P_2;

printf(" (b).The fugacity of steam at 320 C and 170
   bar is %f bar\n",f_CL);
printf("      The fugacity coefficient at 320 C and
   170 bar is , phi = %f\n\n",phi_2);
```

Scilab code Exa 7.41 Determination of fugacities at two states

```
clear;
clc;

//Example - 7.41
//Page number - 278
printf("Example - 7.41 and Page number - 278\n\n");

//Given
T = 300 + 273.15;//[K]
P_1 = 12500*10^(3);//[Pa]
P_2 = 8581*10^(3);//[Pa]
P_3 = 300*10^(3);//[Pa]
V_liq = 1.404;//[cm^(3)/g] - Specific volume of
   liquid
V_liq = (V_liq/10^(6))*18.015;//[m^(3)/mol]
R = 8.314;//[J/mol*K] - Universal gas constant

// state 1: 300 C, 12500 kPa
// state 2: 300 C, 8581 kPa
// state 3: 300 C, 300 kPa

// From state 1 to state 2 the system is liquid
   water and if the molar volume of liquid is
```

```
    assumed costant we can write
// G_2 - G_1 = V_liq*(P_2 - P_1)
// G_2 - G_1 = R*Tlog(f_2/f_1)
// Comparing the above two equations we get
// (f_2/f_1) = exp((V_liq*(P_2 - P_1)/(R*T))
f2_f1 = exp((V_liq*(P_2 - P_1)/(R*T))); // (f_2/f_1)
    = f2_f1 (say)

// In state 2 the fugacity of liquid is same as that
    of saturated vapour and for the vapour phase
    change from state 2 to 3 the fugacity ratio is
    calculated using
// G_3 - G_2 = R*Tlog(f_3/f_2)

// At 300 C, 8581 kPa
H_liq_2 = 2749.0;//[kJ/kg]
S_vap_2 = 5.7045;//[kJ/kg-K]
G_vap_2 = -520.53;//[kJ/kg]
G_vap_2 = G_vap_2*18.015;//[J/mol]

// At 300 C, 300 kPa
H_3 = 3069.3;//[kJ/kg]
S_3 = 7.7022;//[kJ/kg-K]
G_3 = -1345.22;//[kJ/kg]
G_3 = G_3*18.015;//[J/mol]

// Substituting and solving the equation  G_3 - G_2
    = R*Tlog(f_3/f_2)
f3_f2 = exp((G_3 - G_vap_2)/(R*T));//  (f_3/f_2) =
    f3_f2 (say)

// (f_3/f_1) = (f_3/f_2)*(f_2/f_1)
f3_f1 = f3_f2*f2_f1;

printf(" The ratio of fugacity in the final state to
    that in the initial state is given by f3/f2 = %f
    ",f3_f2);
```

Chapter 8

Thermodynamic Cycles

Scilab code Exa 8.1 Calculation of work done

```
clear;
clc;

//Example - 8.1
//Page number - 287
printf("Example - 8.1 and Page number - 287\n\n");

//Given
P_1 = 30;//[bar]
P_2 = 0.04;//[bar]

//(1).Carnot cycle
//It has been reported in the book that at 30 bar
   pressure (saturated) :
H_liq_1 = 1008.42;//[kJ/kg]
H_vap_1 = 2804.2;//[kJ/kg]
S_liq_1 = 2.6457;//[kJ/kg-K]
S_vap_1 = 6.1869;//[kJ/kh-K]
//Therefore, H_1 = H_liq_1, H_2 = H_vap_1, S_1 =
   S_liq_1 and S_2 = S_vap_1
H_1 = H_liq_1;
```

```
H_2 = H_vap_1;
S_1 = S_liq_1;
S_2 = S_vap_1;

//At 0.04 bar pressure (saturated) :
H_liq_2 = 121.46;//[kJ/kg]
H_vap_2 = 2554.4;//[kJ/kg]
S_liq_2 = 0.4226;//[kJ/kg-K]
S_vap_2 = 8.4746;//[kJ/kh-K]

//Dryness fraction at state 3 can be found the fact
   that S_3 = S_2
x_3 = (S_2 - S_liq_2)/(S_vap_2 - S_liq_2);
H_3 = H_liq_2*(1 - x_3) + x_3*H_vap_2;//[kJ/kg]

//Dryness fraction at state 4 can be found the fact
   that S_4 = S_1
x_4 = (S_1 - S_liq_2)/(S_vap_2 - S_liq_2);
H_4 = H_liq_2*(1 - x_4) + x_4*H_vap_2;//[kJ/kg]

//Work done by turbine W_tur = -delta_H = -(H_3 -
   H_2)
W_tur = H_2 - H_3;//[kJ/kg]

//Work supplied by boiler,
q_H = H_2 - H_1;//[kJ/kg]

//Work transfer in compressor is given by
W_com = -(H_1 - H_4);//[kJ/kg]

//Efficiency can now be calculated as
//n = (Net work done/Work supplied by boiler)
n_carnot = (W_tur + W_com)/q_H;

//Efficiency of the Carnot cycle can also be
   determined from the formula
// n = 1 - (T_L/T_H), Where T_L is saturated
   temperature at 0.04 bar and T_H is saturated
```

```
    temperature at 30 bar

printf("(1).Carnot cycle\n\n");
printf("The work done by the turbine is %f kJ/kg\n\n
    ",W_tur);
printf("The heat transfer in the boiler is %f kJ/kg\
    n\n",q_H);
printf("The cycle efficiency is %f\n\n\n",n_carnot);

//(2).Rankine cycle
//The enthalpies at state 2 and 3 remain as in the
    Carnot cycle
//Saturated liquid enthalpy at 0.04 bar is
H_4_prime = H_liq_2;

//Saturated liquid volume at 0.04 bar as reported in
     the book is
V_liq = 0.001004;//[m^(3)/kg]
//Work transfer in pump can be calculated as
W_pump = -V_liq*(P_1 - P_2)*100;//[kJ/kg]

//Work transfer around pump gives, W_pump = -delta_H
     = -(H_1_prime - H_4_prime);
H_1_prime = H_4_prime - W_pump;//[kJ/kg]

//Heat supplied to boiler is
q_H_prime = H_2 - H_1_prime;//[kJ/kg]

//Work done by turbine is
W_tur_prime = H_2 - H_3;//[kJ/kg]

//Efficiency can now be calculated as
//n = (Net work done/Heat input)
n_rankine = (W_tur_prime + W_pump)/q_H_prime;//

printf("(2).Rankine cycle\n\n");
printf("The work done by the turbine is %f kJ/kg\n\n
    ",W_tur_prime);
```

```
printf("The heat transfer in the boiler is %f kJ/kg\
   n\n",q_H_prime);
printf("The cycle efficiency is %f",n_rankine);
```

Scilab code Exa 8.2 Calculation of efficiency of Rankine cycle

```
clear;
clc;

//Example - 8.2
//Page number - 288
printf("Example - 8.2 and Page number - 288\n\n");

//Given
T_max = 700+273.15;//[K] - Maximum temperature.
P_boiler = 10*10^(6);//[Pa] - Constant pressure in
   the boiler
P_condenser = 10*10^(3);//[Pa] - Constant pressure
   in the condenser

//At state 2 i.e, at 700 C and 10 MPa,it has been
   reported in the book that from steam table
S_2 = 7.1687;//[kJ/kg-K] - Entropy
H_2 = 3870.5;//[kJ/kg] - Enthalpy

//At state 3 i.e, at 700 C and 10 KPa,
S_3 = S_2;//[kJ/kg-K]- Entropy

//For sturated steam at 10 kPa, it has been reported
    in the book that from steam table
S_liq = 0.6493;//[kJ/kg-K]- Entropy of saturated
   liquid
S_vap = 8.1502;//[kJ/kg-K] - Enthalpy of saturated
   liquid
//Therefore steam is saturated and its dryness
```

```
    factor can be calculated as
x = (S_2 - S_liq)/(S_vap - S_liq);

//The enthalpy at state 3 is now calculated. For
    steam at 10 kPa,it has been reported in the book
    that from steam table
H_liq = 191.83;//[kJ/kg]
H_vap = 2584.7;//[kJ/kg]
//Therefore enthalpy at state 3 is
H_3 = H_liq*(1-x) + H_vap*x;//[kJ/kg]

//Work done by the turbine
W_tur = -(H_3 - H_2);//[kJ/kg]

//Now we have to calculate work input to the pump
//State 4:Saturated liquid at 10 kPa
//State 4:Compressed liquid at 10 MPa
//Since volume of liquid does not get affected by
    pressure we take volume of saturated liquid at 10
     kPa,
V_liq = 0.001010;//[m^(3)/kg]

//Work transfer in the pump is
W_pump = -V_liq*(P_boiler - P_condenser)*10^(-3);//[
    kJ/kg]

//Energy balance around pump gives, W_pump = -
    delta_H = -(H_1 - H_4)
H_4 = H_liq;// Enthalpy at state 4 (saturated liquid
     at 10 kPa)
H_1 = H_4 - W_pump;//[kJ/kg]

//Heat supplied to boiler is
q_H = H_2 - H_1;//[kJ/kg]

//Efficiency can now be calculated as
//n = (Net work done/Heat input)
n_rankine = (W_tur + W_pump)/q_H;
```

```
printf("The efficiency of the Rankine cycle is found
    to be %f",n_rankine);

//Now let us determine the efficiency of Carnot
   cycle. The maximun temperature is 700 C and
   minimum temperature is that of saturated steam at
    10 kPa,
T_min = 45.81 + 273.15;//[K] - From steam table as
   reported in the book
n_carnot = 1-(T_min/T_max);
//Note that the efficiency of Rankine cycle is less
   than that of carnot cycle.
```

Scilab code Exa 8.3 Calculatrion of COP of carnot refrigerator and heat rejected

```
clear;
clc;

//Example - 8.3
//Page number - 291
printf("Example - 8.3 and Page number - 291\n\n");

//Given
W = 1.1;//[kW] - Work done per ton of refrigeration
//1 ton refrigeration = 3.517 kW, therefore
H = 3.517;//[kW] - Heat absorbed
T_low = -30 + 273.15;//[K] - Low temperature
   maintained

//COP can be calculated as
//COP = (Heat absorbed/Work done)
COP = H/W;

```

```
//For reversed carnot cycle , COP = T_low/(T_high -
    T_low). Solving this we get
T_high = (T_low/COP) + T_low;//[K] - Higher
    temperature

//Heat rejected is
H_rej = W + H;//[kW];

printf("The COP is %f\n\n",COP);
printf("The higher temperature of the cycle is %f K\
   n\n",T_high);
printf("The heat rejected per ton of refrigeration
    is %f kW\n\n",H_rej);
```

Scilab code Exa 8.4 Calculation of minimum power required

```
clear;
clc;

//Example - 8.4
//Page number - 292
printf("Example - 8.4 and Page number - 292\n\n");

//Given
T_high = 20 + 273.15;//[K] - High temperature
T_low = 0 + 273.15;//[K] - Low temperature
Q_H = 10;//[kW] - Heat supplied

//If 'Q_H' is the rate at which heat is taken from
    surrounding and 'W' is the rate at which work is
    done,then
// Q_H = W + Q_L
//(Q_H/Q_L) = (T_high/T_low)
//Also for a reversible cycle, (Q_H/Q_L) = 1 + (W/
    Q_L). Solving we get,
Q_L = (T_low/T_high)*Q_H;//[kW]
W = (Q_H - Q_L) ;//[kW]

printf("The minimum power required is %f kW",W);
```

Scilab code Exa 8.5 Determination of COP and power required

```
clear;
clc;

//Example - 8.5
//Page number - 292
printf("Example - 8.5 and Page number - 292\n\n");

//Given
T_high = 40 + 273.15;//[K] - High temperature
T_low = -20 + 273.15;//[K] - Low temperature
C = 10;//[tons of refrigeration] - Capacity
//1 ton refrigeration = 3.517 kW, therefore
H = C*3.517;//[kW] - Heat absorbed

//For reversed carnot cycle, COP = T_low/(T_high -
   T_low)
COP = T_low/(T_high - T_low);

// COP = (Refrigerating effect)/(Work input),
   therefore power required is given by
P = (H/COP);//[kW]

printf("The COP is %f\n\n",COP);
printf("The power required is %f kW",P);
```

Scilab code Exa 8.6 Determination of maximum refrigeration effect

```
clear;
clc;

//Example - 8.6
//Page number - 292
printf("Example - 8.6 and Page number - 292\n\n");

//Given
COP = 4;//Coefficient of performance
P = 10;//[kW] - Work done on the cycle

//For reversed carnot cycle, COP = T_low/(T_high -
    T_low)
//ratio = (T_high/T_low),therefore
ratio = -1/(COP + 1);

// Refrigerating effect = (COP)*Work input,
    therefore refrigeration is given by
H = COP*P;// [kW]

//Maximum refrigearation in tons is given by
H_max = (H/3.517);

printf("The maximum refrigeration value is %f ton",
    H_max);
```

Scilab code Exa 8.7 Determination of refrigeration effect power consumed and COP of refrigerator

```
clear;
clc;

//Example - 8.7
//Page number - 292
printf("Example - 8.7 and Page number - 292\n\n");
```

```

//Given
m = 0.6;//[kg/s] - mass flow rate
T_low = -20+273.15;//[K] - Temperature at which
    vapour enters the compressor
T_high = 30+273.15;//[K] - Temperature at which
    vapour leaves the condenser

//From saturated refrigeration-12 tables we get,at
    -20 C
H_1 = 178.74;//[kJ/kg] - (H_1 = H_vap)
P_1 = 0.15093;//[MPa] - (P_1 = P_sat)
P_4 = P_1;
S_1 = 0.7087;//[kJ/kg-K] - (S_1 = S_vap)
S_2 = S_1;

//At 30 C
P_2 = 0.7449;//[MPa] - (P_2 = P_sat)
P_3 = P_2;
H_3 = 64.59;//[kJ/kg] - (H_3 = H_liq)
H_4 = H_3;
S_3 = 0.24;//[kJ/kg-K] - (S_3 = S_liq)

//It is assumed that presssure drop in the
    evaporator and condenser are negligible. The heat
     transfer rate in the evaporator is
Q_L = m*(H_1 - H_4);

printf("The heat transfer rate in the evaporator is
    %f kW\n\n",Q_L);

//At state 2 (P = 0.7449 MPa and S = 0.7087 kJ/kg-K)
     and looking in the superheated tables we have to
     calculate the enthalpy at state 2

//At P = 0.7 MPa and S = 0.6917 kJ/kg-K,
H_11 = 200.46;//[kJ/kg]

```

```
//At P = 0.7 MPa and S = 0.7153 kJ/kg-K,
H_12 = 207.73;//[kJ/kg]

//Thus at P = 0.7 MPa and S = 0.7087 kJ/kg-K,
   enthalpy is given by
H_13 = ((S_2 -0.6917)/(0.7153 - 0.6917))*(H_12 -
   H_11) + H_11;//[kJ/kg]

//At P = 0.8 MPa and S = 0.7021 kJ/kg-K,
H_21 = 206.07;//[kJ/kg]

//At P = 0.8 MPa and S = 0.7253 kJ/kg-K,
H_22 = 213.45;//[kJ/kg]

//Thus at P = 0.8 MPa and S = 0.7087 kJ/kg-K,
   enthalpy is given by
H_23 = ((S_2 -0.7021)/(0.7253 - 0.7021))*(H_22 -
   H_21) + H_21;//[kJ/kg]

//At P = 0.7449 MPa, S = 0.7087 kJ/kg-K, the
   enthalpy is
H_2 = ((0.7449 - 0.7)/(0.8 - 0.7))*(H_23 - H_13) +
   H_13;//[kJ/kg]

//Power consumed by the compressor is
W_comp = m*(H_2 - H_1);//[kW]

printf("The power consumed by the compressor is %f
   kW\n\n",W_comp);

//Heat removed in evaporator/work done on compressor
COP_R = Q_L/W_comp;

printf("The COP the refrigerator is %f kW\n\n",COP_R
   );


//At -20 C,saturated conditions
```

```
H_liq = 17.82;//[kJ/kg]
H_vap = 178.74;//[kJ/kg]
x_4 = (H_4 - H_liq)/(H_vap - H_liq);

printf("The dryness factor of refrigerant after the
   expansion valve is %f\n\n",x_4);

//The heat transfer rate in the condenser is
Q_H = m*(H_3 - H_2);//[kW]

printf("The heat transfer rate in the condenser is
   %f kW\n\n",Q_H);

//If the cycle would have worked as a pump then,
//COP_HP = (Heat supplied from condenser/Work done
   on compressor)
COP_HP = (-Q_H)/W_comp;

printf("The COP if cycle would work as a heat pump
   is %f kW\n\n",COP_HP);

//If the cycle would have been a reversed Carnot
   cycle then
COP_C = T_low/(T_high - T_low);

printf("The COP if cycle would run as reversed
   Carnot cycle is %f kW\n\n",COP_C);
```

Scilab code Exa 8.8 Calculation of amount of air

```
clear;
clc;

//Example - 8.8
//Page number - 300
```

```
printf("Example - 8.8 and Page number - 300\n\n");

//Given
//From compressor to the expansion valve the
   pressure is 200 bar and from expansion valve to
   the inlet of compressor the pressure is 1 bar.
//Point 5 is saturated liquid at 1 bar and point 6
   is saturated vapour at 1 bar

//It has been reported in the book that at state 1
   (310 K, 1 bar)
H_1 = 310.38;//[kJ/kg]
//At state 2 (310 K, 200 bar)
H_2 = 277.7;//[kJ/kg]
//At state 5 (1 bar,saturated liquid)
H_5 = -122.6;//[kJ/kg]
//At state 6 (1 bar,saturated vapur)
H_6 = 77.8;//[kJ/kg]

//The enthalpy at point 3 is same at point 4 as the
   expansion is isenthalpic

//The mass condensed is 1 kg and therefore m_1 = m+6
    + 1

//Enthalpy balance around heat exchanger
//m_2*H_2 + m_2*H_6 = m_3*H_3 + m_7*H_7

//Enthalpy balance around separator
//m_4*H_4 = m_5*H_5 + m_6*H_6
//It can be seen that m_1 = m_2 = m_3 = m_4
//and m_6 = m_7 = m_1 - 1

//Substituting the values for enthalpy balance
   around heat exchanger we get,
//m_1*H_2 + (m_1 - 1)*(H_6) = m_1*H_3 + (m_1 - 1)*
   H_1
//and substituting the values for enthalpy balance
```

```
   around seperator we get
//m_1*H_3 = (1)*(-122.6) + (m_1 - 1)*77.8
//H_3 = ((1)*(-122.6) + (m_1 - 1)*77.8)/m_1
//Substituting the expression for 'H_3' in the above
    equation and then solving for m_1, we get
deff('[y]=f(m_1)','y=m_1*H_2+(m_1-1)*(H_6)-m_1*(((1)
   *(-122.6) + (m_1 - 1)*77.8)/m_1)-(m_1-1)*H_1');
m_1 = fsolve(4,f);//[kg]
//Thus to liquify 1 kg of air compression of m_1 kg
   of air is carried out.

//Now substituting this value of m_1 to get the
   value of H_3,
H_3 = ((1)*(-122.6) + (m_1 - 1)*77.8)/m_1;//[kJ/kg]

//From given compressed air table we see at 200 bar
   and 160 K,
H_3_1 = 40.2;//[kJ/kg]

//At 200 bar and 180 K,
H_3_2 = 79.8;//[kJ/kg]
//By interpolation we get,
T_3 = ((H_3 - H_3_1)*(180 - 160))/(H_3_2 - H_3_1) +
   160;//[K]

printf("Temperature before throttling is %f",T_3);
```

Scilab code Exa 8.9 Calculation of amount of air and temperature

```
clear;
clc;
funcprot(0);

//Example - 8.9
//Page number - 304
```

```
printf("Example - 8.9 and Page number - 304\n\n");

//Given
//At 1 bar, 310 K
H_1 = 310.38;//[kJ/kg]
//At 200 bar, 310 K
H_2 = 277.7;//[kJ/kg]
//At 1 bar, Saturated liquid
H_7 = -122.6;//[kJ/kg]
//At 1 bar, Saturated vapour
H_8 = 77.8;//[kJ/kg]
//At 200 bar, 200 K
H_3 = 117.6;//[kJ/kg]
//At 1 bar, 100 K
H_11 = 98.3;//[kJ/kg]

//(1)
//For 1 kg of liquid air obtained,the overall
    enthalpy balance is
//m_2*H_2 = W - 122.6 + (m_2 - 1)*H_1
//W = - 0.8*m_2*(H_11 - H_3)
//Overall enthalpy balance equation becomes
//H_2*m_2 = 15.44*m_2 - H_7 + (m_2 - 1)*H_1, solving
m_2_prime = (H_7 - H_1)/(H_2 - 15.44 - H_1);

printf("The number of kilograms of air compressed
    per kg of liquid air produced is %f kg\n\n",
    m_2_prime);

//(2)
//Enthalpy balance around separator is
//0.2*m_2*H_5 = -H_7 + (0.2*m_2 - 1)*H_8, solving
m_2 = m_2_prime;
H_5_prime = ((0.2*m_2-1)*H_8 - H_7)/(0.2*m_2);

//At point 5, P = 200 bar and enthalpy is
H_5_1 = -33.53;//[kJ/kg]
//From compressed air tables at 200 bar and 140 K,
H_5_2 = 0.2;//[kJ/kg]
//At 200 bar and 120 K,
H_5_3 = -38.0;//[kJ/kg]
//Solving by interpolation we get
T_5 = ((H_5_1 - H_5_3)*(140 - 120))/(H_5_2 - H_5_3)
    + 120;//[K]

printf("The temperature of air before throttling is
    %f K\n\n",T_5);

//(3)
//During mixing of streams 8 and 11 to produce
    stream 9, the enthalpy balance is
// (0.2*m_2 - 1)*H_8 + 0.8*m_2*H_11 = (m_2 - 1)*H_9,
    Solving for H_9

H_9_prime = ((0.2*m_2-1)*H_8+0.8*m_2*H_11)/(m_2 - 1)
    ;

//From given compressed air tables at 1 bar and 100
    K,
H_9_1 = H_11;
//At 1 bar and 90 K
H_9_2 = 87.9;//[kJ/kg]
//Solving by interpolation we get
T_9 = ((H_9_prime - H_9_2)*(100 - 90))/(H_9_1 -
    H_9_2) + 90;//[K]

printf("The temperature of stream entering second
    heat exchanger is %f K\n\n",T_9);

//(4)
//Enthalpy balance around first heat exchanger is
//H_2*m_2 + (m_2 - 1)*H_10 = H_3*m-2 + (m-2 - 1)*H_1
    , solving for H_10

H_10_prime = ((m_2 - 1)*H_1 + H_3*m_2 - H_2*m_2)/(
    m_2 - 1);
```

```
//From given compressed air tables at 1 bar and 140
   K,
H_10_1 = 139.1;//[kJ/kg]
//At 1 bar and 120 K
H_10_2 = 118.8;//[kJ/kg]
//Solving by interpolation we get
T_10 = ((H_10_prime - H_10_2)*(140 - 120))/(H_10_1 -
    H_10_2) + 120;//[K]

printf("The temperature of stream exiting second
   heat exchanger is %f K\n\n",T_10);
```

Scilab code Exa 8.10 Determination of temperature of air

```
clear;
clc;
funcprot(0);

//Example - 8.10
//Page number - 307
printf("Example - 8.10 and Page number - 307\n\n");

//Given
P_high = 40;//[bar]
P_low = 5;//[bar]
m_1 = 0.5;//[kg/s] - Rate of mass moving through the
    expander
m_2 = 0.1;//[kg/s] - Rate of mass of vapour mixing
   with air
e = 0.7;//Efficiency

//At state 3,(40 bar and 200 K),enthalpy and entropy
    is given by
H_3 = 179.7;//[kJ/kg]
```

```
S_3 = 5.330;//[kJ/kg-K]

//If isentropic conditions exits in the turbine then
    state 11 is at 5 bar
S_11 = 5.330;//[kJ/kg-K]
//From given compressed air tables at 5 bar and 120
   K,
H_11_1 = 113.6;//[kJ/kg]
S_11_1 = 5.455;//[kJ/kg-K]
//At 5 bar and 100 K
H_11_2 = 90.6;//[kJ/kg]
S_11_2 = 5.246;//[kJ/kg-K]
//The enthalpy has to be determined when S = S_3
//Solving by interpolation we get
H_11_s = ((H_11_1 - H_11_2)*(S_3 - S_11_2))/(S_11_1
   - S_11_2) + H_11_2;//[kJ/kg]

//The adiabatic efficiency of tyrbine is given by
//(H_3 - H_11_a)/(H_3 - H_11_s) = e
H_11_a = H_3 - e*(H_3 - H_11_s);//[kJ/kg] - Actual
   enthalpy

 //At 5 bar,the saturated enthalpy is given to be
H_8 = 88.7;//[kJ/kg]
//From enthalpy balance during mixing we get,
//0.1*H_8 + 0.5*H_11_a = 0.6*H_9
H_9 = (m_2*H_8 + m_1*H_11_a)/(m_1 + m_2);//[kJ/kg]

//From given compressed air tables at 5 bar and 140
   K,
H_9_1 = 135.3;//[kJ/kg]
//At 5 bar and 120 K
H_9_2 = 113.6;//[kJ/kg]
//By interpolation we get
T_9 = ((H_9 - H_11_1)*(140 - 120))/(H_9_1 - H_11_1)
   + 120;//[K]

printf(" The temperature of air entering the second
    heat exchanger is %f K\n\n",T_9);
```

Chapter 10

Residual Properties by Equations of State

Scilab code Exa 10.1 Determination of expression for residual enthalpy internal energy and Gibbs free energy

```
clear;
clc;

//Example - 10.1
//Page number - 323
printf("Example - 10.1 and Page number - 323\n\n");

//This problem involves proving a relation in which
   no mathematical components are involved.
//For prove refer to this example 10.1 on page
   number 323 of the book.
printf("This problem involves proving a relation in
   which no mathematical components are involved.\n\
   n");
printf("For prove refer to this example 10.1 on page
    number 323 of the book.")
```

Scilab code Exa 10.2 Preparation of fugacity and fugacity coefficient

```
clear;
clc;

//Example - 10.2
//Page number - 334
printf("Example - 10.2 and Page number - 334\n\n");

// Given
T = 40 + 273.15;//[C] - Temperature
P_1 = 0;//[bar]
P_2 = 10;//[bar]
V_liq = 90.45;//[cm^(3)/mol]
V_liq = V_liq*10^(-6);//[m^(3)/mol]
P_sat = 4.287;//[bar]

// For butadiene
T_c = 425.0;//[K] - Critical temperature
P_c = 43.3;//[bar] - Critical pressure
P_c = P_c*10^(5);//[N/m^(2)]
w = 0.195;// Acentric factor
R = 8.314;//[J/mol*K] - Universal gas constant

// Let us calculate second virial coefficient at 40
   C
Tr = T/T_c;// Reduced temperature
B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B = ((B_0 + (w*B_1))*(R*T_c))/P_c;//[m^(3)/mol] -
    Second virial coefficient

// log(f/P) = (B*P)/(R*T)
// f = P*exp((B*P)/(R*T))

printf(" The table is as follows\n\n")
printf(" P(bar) \t\t f(bar) \t\t phi\n");

P = [1,2,3,4,4.287,5,6,8,10];
f = zeros(9);
phi = zeros(9);
for i=1:5;
    f(i)=P(i)*(exp((B*P(i)*10^(5))/(R*T)));//[bar]
       // Pressure inside the exponential term has
       to be in N/m^(2)
    phi(i)= (f(i)/P(i));
   printf(" %f \t %f \t\t\t %f\n",P(i),f(i),phi(i));
end
f_sat = f(5);

// From pressure of 4.287 bar onwards the valid
   equation to compute fugacity of compressed liquid
    is given by
// f = f_sat*exp[V_liq*(P-P_sat)/(R*T)]

for j=6:9
    f(j) = f_sat*exp((V_liq*(P(j)-P_sat)*10^(5))/(R*
       T));//[bar]  // Pressure inside the
       exponential term has to be in N/m^(2)
    phi(j) = f(j)/P(j);
    printf(" %f \t %f \t\t\t %f\n",P(j),f(j),phi(j))
       ;
end
```

Scilab code Exa 10.3 Calculation of enthalpy entropy and internal energy change

```
clear;
```

```
clc;

//Example - 10.3
//Page number - 334
printf("Example - 10.3 and Page number - 334\n\n");

//Given
n = 100;//[mol] - No of moles
T_1 = 600;//[K] - Initial temperature
T_2 = 300;//[K] - Final temperature
P_1 = 10;//[atm] - Initial pressure
P_1 = P_1*101325;//[Pa]
P_2 = 5;//[atm] - Final presssure
P_2 = P_2*101325;//[Pa]
Tc = 369.8;//[K] - Critical temperature
Pc = 42.48;//[bar] - Critical pressure
Pc = Pc*10^(5);//[Pa]
w = 0.152;
R = 8.314;//[J/mol*K] - Universal gas constant

// At 600 K
Tr = T_1/Tc;// Reduced temperature
B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
B = ((B_0 + (w*B_1))*(R*Tc))/Pc;//[m^(3)/mol] -
    Second virial coefficient
dB0_dT = 0.422*1.6*Tc^(1.6)*T_1^(-2.6);// (dB_0/dT)
dB1_dT = 0.172*4.2*Tc^(4.2)*T_1^(-5.2);// (dB_1/dT)
dB_dT = ((R*Tc)/(Pc))*(dB0_dT + w*dB1_dT);// dB/dT

// Now let us calculate B and dB/dT at 300 K
Tr_prime = T_2/Tc;// Reduced temperature
B_0_prime = 0.083-(0.422/(Tr_prime)^(1.6));
B_1_prime = 0.139-(0.172/(Tr_prime)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
B_prime = ((B_0_prime + (w*B_1_prime))*(R*Tc))/Pc;//
    [m^(3)/mol] - Second virial coefficient
```

```
dB0_dT_prime = 0.422*1.6*Tc^(1.6)*T_2^(-2.6);// (
   dB_0/dT)
dB1_dT_prime = 0.172*4.2*Tc^(4.2)*T_2^(-5.2);// (
   dB_1/dT)
dB_dT_prime = ((R*Tc)/(Pc))*(dB0_dT_prime + w*
   dB1_dT_prime);// dB/dT

// The change in enthalpy for ideal gas is given by
delta_H_ig = integrate('-0.966+7.279*10^(-2)*T
   -3.755*10^(-5)*T^(2)+7.58*10^(-9)*T^(3)','T',T_1,
   T_2);//[cal/mol]
delta_H_ig = delta_H_ig*4.184;//[J/mol]

// We know that delta_H_ig = delta_U_ig + R*delta_T.
    Therefore change in internal energy is given by
delta_U_ig = delta_H_ig - R*(T_2 - T_1);//[J/mol]

// The change in entropy of ideal gas is given by
//delta_S_ig = integrate('Cp_0/T','T',T_1,T_2) - R*
   log(P_2/P_1);
delta_S_ig = integrate('(-0.966+7.279*10^(-2)*T
   -3.755*10^(-5)*T^(2)+7.58*10^(-9)*T^(3))/T','T',
   T_1,T_2)*4.184 - R*log(P_2/P_1);// [J/mol-K]

// Now let us calculate the change in enthalpy of
   gas. We know that
// delta_H = delta_H_ig + delta_H_R
// delta_H_R = H_2_R - H_1_R
H_2_R = B_prime*P_2 - P_2*T_2*dB_dT_prime;// [J/mol]
H_1_R = B*P_1 - P_1*T_1*dB_dT;// [J/mol]
delta_H_R = H_2_R - H_1_R;// [J/mol]
delta_H = delta_H_ig + delta_H_R;// [J/mol]

// Let us calculate the residual entropy of gas
S_2_R = -P_2*dB_dT_prime;//[J/mol-K]
S_1_R = -P_1*dB_dT;//[J/mol-K]
delta_S = delta_S_ig + (S_2_R - S_1_R);//[J/mol-K]

```

```
// Let us calculate the residual internal energy of
   gas
U_2_R = -P_2*T_2*dB_dT_prime;//[J/mol-K]
U_1_R = -P_1*T_1*dB_dT;//[J/mol-K]
delta_U = delta_U_ig + (U_2_R - U_1_R);//[J/mol-K]

// For 100 mol sample,
delta_H_ig = delta_H_ig*n*10^(-3);//[kJ/mol]
delta_H = delta_H*n*10^(-3);//[kJ/mol]

delta_U_ig = delta_U_ig*n*10^(-3);//[kJ/mol]
delta_U = delta_U*n*10^(-3);//[kJ/mol]

delta_S_ig = delta_S_ig*n*10^(-3);//[kJ/mol]
delta_S = delta_S*n*10^(-3);//[kJ/mol]

printf(" The value of delta_H = %f kJ/mol\n",delta_H
   );
printf(" The value of delta_H_ig (ideal gas)= %f kJ/
   mol\n\n",delta_H_ig);
printf(" The value of delta_U = %f kJ/mol\n",delta_U
   );
printf(" The value of delta_U_ig (ideal gas) = %f kJ
   /mol\n\n",delta_U_ig);
printf(" The value of delta_S = %f kJ/mol\n",delta_S
   );
printf(" The value of delta_S_ig (ideal gas) = %f kJ
   /mol\n\n",delta_S_ig);
```

Scilab code Exa 10.4 Calculation of molar heat capacity

```
clear;
clc;

//Example - 10.4
```

```
//Page number - 337
printf("Example - 10.4 and Page number - 337\n\n");

// Given
T = 35 + 273.15;//[K] - Temperature
P = 10;//[atm] - Pressure
P = P*101325;//[Pa]
// Methane obeys the equation of state
// Z = 1 + (P*B)/(R*T)

// At 35 C,
B = -50;//[cm^(3)/mol]
dB_dT = 1.0;//[cm^(3)/mol-K] - dB/dT
dB_dT = dB_dT*10^(-6);//[m^(3)/mol-K]
d2B_dT2 = -0.01;//[cm^(3)/mol-K^(2)] - d^2(B)/d(T^2)
d2B_dT2 = d2B_dT2*10^(-6);//[m^(3)/mol-K^(2)]

// Ideal gas molar heat capacity of methane is given
    by
// Cp_0 = 4.75 + 1.2*10^(-2)*T + 0.303*10^(-5)*T^(2)
    - 2.63*10^(-9)*T^(3)

// The molar heat capacity is given by
// Cp = Cp_0 + Cp_R
// For virial gas equation of state
Cp_R = -P*T*d2B_dT2;//[J/mol-K]

// thus heat capacity is given by
// Cp = a + b*T + c*T^(2) + d*T^(3) - P*T*d2B_dT2
// Putting the values , we get
Cp = (4.75 + 1.2*10^(-2)*T + 0.303*10^(-5)*T^(2) -
    2.63*10^(-9)*T^(3))*4.184 - P*T*d2B_dT2;//[J/mol-
    K]

printf(" The molar heat capacity of methane is %f J/
    mol-K\n",Cp);
```

Scilab code Exa 10.5 Calculation of final temperature after expansion

```
clear;
clc;

//Example - 10.5
//Page number - 338
printf("Example - 10.5 and Page number - 338\n\n");

//Given
T_1 = 360;//[K] - Initial temperature
P_1 = 10;//[bar] - Initial pressure
P_1 = P_1*10^(5);//[Pa]
Tc = 408.1;//[K] - Critical temperature
Pc = 36.48;//[bar] - Critical pressure
Pc = Pc*10^(5);//[Pa]
w = 0.181;
R = 8.314;//[J/mol*K] - Universal gas constant
Cv_0 = 106.0;//[J/mol-K]

// At 360 K
Tr = T_1/Tc;// Reduced temperature
B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
B = ((B_0 + (w*B_1))*(R*Tc))/Pc;//[m^(3)/mol] -
    Second virial coefficient
dB0_dT = 0.422*1.6*Tc^(1.6)*T_1^(-2.6);// (dB_0/dT)
dB1_dT = 0.172*4.2*Tc^(4.2)*T_1^(-5.2);// (dB_1/dT)
dB_dT = ((R*Tc)/(Pc))*(dB0_dT + w*dB1_dT);// dB/dT

// Since system is adiabatic therefore no heat
    exchange will take place , q = 0
// and expansion takes place into vacuum,threfore W
```

```
    = 0
// From first law delta_U = 0. If the gas would have
    followed ideal gas equation of state the final
   temperature would have been the same as initial
   as delta_U = 0
// But for real gases
// delta_U = delta_U_ig + delta_U_R
// delta_U = delta_U_ig + U_2_R - U_1_R
// For equation of state Z = 1 + (B*P)/(R*T)
// V = B + (R*T)/P
// U_R = -P*T*(dB/dT)

// delta_U_ig = Cv_0*(T_2 - T_1)
// delta_U = Cv_0*(T_2 - T_1) - P_2*T_2*(dB/dT)_2 +
   P_1*T_1*(dB/dT)_1

// At state 1
V_1 = B + (R*T_1)/P_1;//[m^(3)/mol] - Molar volume
// At state 1
V_2 = 10*V_1;//[m^(3)/mol] - Molar volume

// From the equation delta_U = 0
// Cv_0*(T_2 - T_1) - ((R*T_2)/(V_2 - B_2))*T_2*(dB/
   dT)_2 + P_1*T_1*(dB/dT)_1 = 0

// Now we need to solve the above equation to get
   the value of T_2
// In above equation the magnitude of second term is
    much smaller as compared to the third term
   because the molar volume has become 10 times
// So neglecting second term, we have
// Cv_0*(T_2 - T_1) + P_1*T_1*(dB/dT)_1 = 0
T_2 = -(P_1*T_1*(dB_dT))/Cv_0 + T_1;//[K]

// For exact calculation of final temperature, let
   us start with a temperature, say
T = 350;

```

```
fault = 10;
while(fault>0.007)
    Tr_prime = T/Tc;// Reduced temperature
    B_0_prime = 0.083-(0.422/(Tr_prime)^(1.6));
    B_1_prime = 0.139-(0.172/(Tr_prime)^(4.2));
    //We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
    B_prime = ((B_0_prime + (w*B_1_prime))*(R*Tc))/
       Pc;//[m^(3)/mol] - Second virial coefficient
    dB0_dT_prime = 0.422*1.6*Tc^(1.6)*T_2^(-2.6);//
       (dB_0/dT)
    dB1_dT_prime = 0.172*4.2*Tc^(4.2)*T_2^(-5.2);//
       (dB_1/dT)
    dB_dT_prime = ((R*Tc)/(Pc))*(dB0_dT_prime + w*
       dB1_dT_prime);// dB/dT
    deff('[y]=f(T)','y= 106*(T-T_1)+972.72-((R*T^(2)
       )/(V_2-B_prime))*dB_dT_prime');
    T_prime = fsolve(0.15,f);
    fault=abs(T-T_prime);
    T = T + 0.001;
end

printf(" The final temperature is %f K\n",T);
```

Scilab code Exa 10.6 Calculation of fugacity of liquid benzene

```
clear;
clc;

//Example - 10.6
//Page number - 339
printf("Example - 10.6 and Page number - 339\n\n");

//Given
T = 220 + 273.15;//[K] - Temperature
Tc = 562.2;//[K] - Critical temperature
```

```
Pc = 48.98;//[bar] - Critical pressure
Pc = Pc*10^(5);//[Pa]
w = 0.210;
R = 8.314;//[J/mol*K] - Universal gas constant
P_sat = 1912.86;//[kPa] - Saturation pressure at 220
    C
P_sat = P_sat*10^(3);//[Pa]
Mol_wt = 78.114;//[g/mol] - Molecular weight of
   benzene

//(1)
// Since liquid and vapour are in equilibrium the
   fugacity is saturated fugacity (f_sat) and can be
    calculated using virial gas equation of state
// At 220 C
Tr = T/Tc;// Reduced temperature
B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
B = ((B_0 + (w*B_1))*(R*Tc))/Pc;//[m^(3)/mol] -
   Second virial coefficient

// We know that log(f/P) = (B*P)/(R*T)
// Thus at saturated conditions
// log(f_sat/P_sat) = B*P_sat/(R*T)
f_sat = P_sat*(exp((B*P_sat)/(R*T)));//[Pa]
f_sat = f_sat*10^(-3);//[kPa]

printf(" (1).The fugacity of liquid benzene is %f
   kPa\n\n",f_sat);

//(2)
P = 2014.7;// [psia] - Total gauge pressure
P = 138.94;// [bar]
P = P*10^(5);// [Pa]
den = 0.63;// [g/cm^(3)] - density of benzene
den = den*10^(3);// [kg/m^(3)]

```

```
// Therefore specific volume is
V = 1/den;//[m/^(3)/kg]
// Molar volume is given by
V = V*Mol_wt*10^(-3);//[m^(3)/mol]

// Thus fugacity at 220 C and pressure P is given by
f = f_sat*(exp((V*(P-P_sat))/(R*T)));

printf(" (2).The fugacity of liquid benzene is %f
   kPa\n\n",f);
```

Scilab code Exa 10.7 Calculation of molar enthalpy

```
clear;
clc;

//Example - 10.7
//Page number - 341
printf("Example - 10.7 and Page number - 341\n\n");

//Given
// C = -0.067 + 30.7/T
// D = 0.0012 - 0.416/T

T = 80 + 273.15;//[K]
P = 30;//[bar]
//P = P;//[N/m^(2)]
R = 8.314;//[J/mol*K] - Universal gas constant

// We have the relation derived in the book
// d(G/(R*T)) = (V/(R*T))dP - (H/(R*T^(2)))dT
// Writing the same equation for ideal gas and
   subtracting it from the above equation we get
// d(G_R/(R*T)) = (V_R/(R*T))dP - (H_R/(R*T^(2)))dT
// Therefore , H_R/(R*T^(2)) = -[del((G_R)/(R*T))/del
```

```
   (T)]_P

// Substituting the relation G_R/(R*T) = log(f/P),
   we get
// H_R/(R*T^(2)) = -[del(log(f/P))/del(T)]_P = -[del
   (-C*P - D*P^(2))/del(T)]_P
// or, H_R/(R*T^(2)) = P*(dC/dT) + P^(2)*dD/dT
// Note that in the above equation the partial
   derivative is replaced by full derivative as C
   and D are functions of temperature. Therfore we
   get
// H_R/(R*T^(2)) = (30.7*P)/T^(2) + (0.416*P^(2))/T
   ^(2)
// H_R/R = - 30.7*P + 0.416*P^(2)

// Substituting the given conditions we get
H_R = R*(-30.7*P + 0.416*P^(2));//[J/mol]

printf(" The molar enthalpy of the gas relative to
   that of the ideal gas at 80 C and 30 bar pressure
    is, H_R = %f J/mol\n",H_R);
```

Scilab code Exa 10.8 Determination of second and third virial coefficients and fugacity

```
clear;
clc;
funcprot(0);

//Example - 10.8
//Page number - 341
printf("Example - 10.8 and Page number - 341\n\n");

//Given
// (1)
```

```
T = 311;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant
// Pressure in 'bar' is given below
P =
   [0.690,1.380,2.760,5.520,8.280,11.034,13.800,16.550];

// Molar volume in 'm^(3)/mol' is given below
V =
   [0.0373,0.0186,0.00923,0.00455,0.00298,0.00220,0.00175,0.00144];


// Z = 1 + (B/V) + (C/V^(2))
// (Z-1)*V = B + (C/V)


Z=zeros(8);
k=zeros(8);
t=zeros(8);
for i=1:8;
    Z(i)=(P(i)*10^(5)*V(i))/(R*T);
    k(i)=(Z(i)-1)*V(i);
    t(i)=1/V(i);
end
[C,B,sig]=reglin(t',k');

//From the regression , we get intercept = B and
   slope = C,and thus ,

printf(" (1).The second virial coefficient of CO2 is
    given by B = %e m^(3)/mol\n",B);
printf("      The thied virial coefficient of CO2 is
   given by C = %e m^(6)/mol^(2)\n\n",C);

// (2)
P_final = 13.8;//[bar]
// We know that R*T*log(f/P) = integrate('V-(R*T)/P
   ','P',0,P)
// Therefore we have to plot V - (R*T)/P versus P
```

```
    and calculate the area beneath the curve from 0
    to 13.8 bar
// For this we need the value of the term V − (R∗T)/
    P at P = 0. At low pressure the virial equation
    becomes
// Z = 1 + (B/V)
//and V − (R∗T)/P = (Z∗R∗T)/P − (R∗T)/P = (1 + (B/V)
    )∗((R∗T)/P) − (R∗T)/P = (B∗R∗T)/(P∗V) = (B/Z)
// Thus lim P tending to zero (V − (R∗T)/P) = B     (
     as P tend to zero , Z tend to 1 )

P_prime =
    [0.000,0.690,1.380,2.760,5.520,8.280,11.034,13.800];

V_prime =
    [0.000,0.0373,0.0186,0.00923,0.00455,0.00298,0.00220,0.00175];

summation = 0;
x=zeros(8);
y=zeros(8);
z=zeros(8);
for j=2:8;
    x(j)=V_prime(j)-(R*T)/(P_prime(j)*10^(5));//[m
        ^(3)/mol]
    y(j)=(x(j) + x(j-1))/2;
    z(j)=y(j)*((P_prime(j)-P_prime(j-1)))*10^(5);
    summation = summation + z(j) ;//[J/mol]
end

summation = summation + 2*z(2) - z(2);// Because in
    the above calculation ,in order to calculate the
    average a summation of z(2) is not included ,only
    half of it gets added

// Now we have , area = integrate('V − (R∗T)/P','P
    ',0,13.8∗10^(5)) = summatiom
// R∗T∗log(f/P) = summation
f = P_final*(exp(summation/(R*T)));//[bar]
```

```

printf(" (2).The fugacity of steam at 311 K and 13.8
    bar pressure is %f bar",f);
```

Scilab code Exa 10.9 Determination of second and third virial coefficients

```
clear;
clc;

//Example - 10.9
//Page number - 344
printf("Example - 10.9 and Page number - 344\n\n");

//Given
T = 0 + 273.15;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant
// Pressure in 'atm' is given below
P = [100,200,300,400,500,600,700,800,900,1000];
// The compressibility factor values are
Z =
   [1.069,1.138,1.209,1.283,1.356,1.431,1.504,1.577,1.649,1.720];

// Z = 1 + (B/V) + (C/V^(2))
// (Z-1)*V = B + (C/V)


V = zeros(1,10);
k = zeros(1,10);
t = zeros(1,10);
for i=1:10;
    V(1,i)=Z(i)*R*T/(P(i)*101325);//[m^(3)/mol]
    k(1,i)=(Z(i)-1)*V(i);
    t(1,i)=1/V(i);
end
```

```
[C,B,sig]=reglin(t,k);

//From the regression , we get intercept = B and
   slope = C, and thus ,

printf(" (1).The second virial coefficient of H2 is
   given by B = %e m^(3)/mol\n",B);
printf("     The thied virial coefficient of H2 is
   given by C = %e m^(6)/mol^(2)\n\n",C);

// (2)
// We know that , limit P tending to zero (V-(R*T)/P)
    = B, therfore P = 0, V-(R*T)/P = B
// Now let us tabulate V-(R*T)/P and determine the
   integral integrate('(V-(R*T)/P)','P',0,1000)

P_prime =
   [0,100,200,300,400,500,600,700,800,900,1000];
Z_prime =
   [0,1.069,1.138,1.209,1.283,1.356,1.431,1.504,1.577,1.649,1.720];


summation = 0;
V_prime = zeros(1,11);
x = zeros(1,11);
y = zeros(1,11);
z = zeros(1,11);
for j=2:11;
    V_prime(1,j)=Z_prime(j)*R*T/(P_prime(j)*101325);
       //[m^(3)/mol]
    x(1,j)=V_prime(j)-(R*T)/(P_prime(j)*101325);
    y(1,j)=(x(j) + x(j-1))/2;
    z(1,j)=y(j)*((P_prime(j)-P_prime(j-1)))*101325;
    summation = summation + z(j) ;//[J/mol]
end

summation = summation + 2*z(2) - z(2);// Because in
   the above calculation ,in order to calculate the
```

```
    average a summation of z(2) is not included ,only
    half of it gets added

// Now we have
// R*T*log(f/P) = summation
P_dash = 1000;//[atm] - Pressure at which fugacity
    is to be calculated
T_dash = 273.15;//[K] - Temperature at which
    fugacity is to be calculated
f = P_dash*exp(summation/(R*T_dash));//[atm]

printf(" (2).The fugacity of H2 at 0 C and 1000 atm
    pressure is , f = %f atm\n",f);
```

Scilab code Exa 10.10 Determination of work done and the exit temperature

```
clear;
clc;

//Example - 10.10
//Page number - 345
printf("Example - 10.10 and Page number - 345\n\n");

//Given
P_1 = 1*10^(6);//[Pa] - Initial pressure
T_1 = 200 + 273.15;//[K] - Initial temperature
P_2 = 8*10^(6);//[Pa] - Final pressure
Tc = 647.1;//[K] - Critical temperature of water
Pc = 220.55;//[bar] - Critical pressure of water
Pc = Pc*10^(5);//[Pa]
w = 0.345;
R = 8.314;//[J/mol*K] - Universal gas constant

// For the virial gas the following are the
```

```
    relations for residual enthalpy and entropy
// H_R = B*P - P*T*(dB/dT)
// S_R = -P*(dB/dT)
// Where, (dB/dT) = ((R*Tc)/Pc)*((dB_0/dT) + w*(dB_1
    /dT))
// dB0_dT = 0.422*1.6*Tc^(1.6)*T^(-2.6);// (dB_0/dT)
// dB1_dT = 0.172*4.2*Tc^(4.2)*T^(-5.2);// (dB_1/dT)

// (1)
Cp_0 = 29.114;//[J/mol-K] - Specific heat capacity
    at constant pressure
// For the isentropic process entropy change is zero
    , thus
// delta_S = Cp_0*log(T_2/T_1) - P_2*(dB/dT)_2 + P_1
    *(dB/dT)_1 = 0

// At state 1,
Tr_1 = T_1/Tc;
B0_1 = 0.083 - 0.422/(Tr_1^(1.6));
B1_1 = 0.139 - 0.172/(Tr_1^(4.2));
// (B*Pc)/(R*Tc) = B0 + w*B1
B_1 = ((B0_1 + w*B1_1)*(R*Tc))/Pc;// [m^(3)/mol] -
    Second virial coefficient at state 1
dB0_dT_1 = 0.422*1.6*Tc^(1.6)*T_1^(-2.6);// (dB_0/dT
    )
dB1_dT_1 = 0.172*4.2*Tc^(4.2)*T_1^(-5.2);// (dB_1/dT
    )
dB_dT_1 = ((R*Tc)/Pc)*((dB0_dT_1) + w*(dB1_dT_1));//
     (dB/dT)_1

// Now let us assume the exit temperature to be 870
    K, at this temperature
// T_2 = 870;//[K] -
// At this temperature
// delta_S = Cp_0*log(T_2/T_1) - P_2*(dB/dT)_2 + P_1
    *(dB/dT)_1 =


T_2 = 860;//[K] - Exit temperature
// Therefore at state 2, we have
Tr_2 = T_2/Tc;
B0_2 = 0.083 - 0.422/(Tr_2^(1.6));
B1_2 = 0.139 - 0.172/(Tr_2^(4.2));
// (B*Pc)/(R*Tc) = B0 + w*B1
B_2 = ((B0_2 + w*B1_2)*(R*Tc))/Pc;// [m^(3)/mol] -
    Second virial coefficient at state 2
dB0_dT_2 = 0.422*1.6*Tc^(1.6)*T_2^(-2.6);// (dB_0/dT
    )
dB1_dT_2 = 0.172*4.2*Tc^(4.2)*T_2^(-5.2);// (dB_1/dT
    )
dB_dT_2 = ((R*Tc)/Pc)*((dB0_dT_2) + w*(dB1_dT_2));//
     (dB/dT)_2

delta_H_s = Cp_0*(T_2 - T_1) + B_2*P_2 -P_2*T_2*(
    dB_dT_2) - B_1*P_1 + P_1*T_1*(dB_dT_1);//[J/mol]
    - Enthalpy change

// As no heat exchange is assumed to take place with
     the surroundings,work transfer is given by
W_1 = - delta_H_s;// [J/mol]

printf(" (1).The exit temperature is %f K\n",T_2);
printf("     The required amount of work is %f J/mol
    \n\n",W_1);


// (2)
eff = 0.8;// Adiabatic efficiency
delta_H_a = delta_H_s/0.8;// Actual enthalpy change

// Now for calculating the value of T_exit
// delta_H_a = Cp_0*(T_exit - T_1) + B*P_2 -P_2*
    T_exit*(dB_dT) - B_1*P_1 + P_1*T_1*(dB_dT_1)
// On simplification we get
// 29.114*(T_2 - T_1)*B_2*8*10^(6)-8*10^(6)*T_2*(dB/
    dT)_2 = 12643.77
```

```

// Let us assume a temperature of say
T = 900;//[K]
fault=10;

while(fault>0.3)
    Tr = T/Tc;
    B0 = 0.083 - 0.422/(Tr^(1.6));
    B1 = 0.139 - 0.172/(Tr^(4.2));
    // (B*Pc)/(R*Tc) = B0 + w*B1
    B = ((B0 + w*B1)*(R*Tc))/Pc;// [m^(3)/mol] -
       Second virial coefficient at state 2
    dB0_dT = 0.422*1.6*Tc^(1.6)*T^(-2.6);// (dB_0/dT
       )
    dB1_dT = 0.172*4.2*Tc^(4.2)*T^(-5.2);// (dB_1/dT
       )
    dB_dT = ((R*Tc)/Pc)*((dB0_dT) + w*(dB1_dT));// (
       dB/dT)_1
    deff('[y]=f(T_exit)','y = delta_H_a - Cp_0*(
       T_exit - T_1) + B*P_2 -P_2*T_exit*(dB_dT) -
       B_1*P_1 + P_1*T_1*(dB_dT_1)');
    T_exit = fsolve(900,f);
    fault=abs(T-T_exit);
    T = T + 0.2;
end
Texit = T;

// As no heat exchange is assumed to take place with
    the surroundings,work transfer is given by
W_2 = - delta_H_a;// [J/mol]

printf(" (2).The exit temperature is %f K\n",Texit);
printf("      The required amount of work is %f J/mol
   \n\n",W_2);

//(3)
// Cp_0 = 7.7 + 0.04594*10^(-2)*T + 0.2521*10^(-5)*T
   ^(2) - 0.8587*10^(-9)*T^(3)
```

```
// The entropy change for a gas following the virial
    equation of state is given by
// delta_S = integrate('Cp_0/T','T',T_1,T_2) - R*log
   (P_2/P_1) - P_2*(dB/dT)_2 + P_1*(dB/dT)_1
// For an isentropic process the entropy change is
   zero and substituting the various values in the
   above equation we get
// 32.2168*log(T_2) + 0.1922*10^(-2)*T_2 +
   0.5274*10^(-5)*T_2^(2) - 1.1976*10^(-9)*T_2^(3)
   -8*10^(6)*(dB/dT)_2 -216.64 = 0

// Let us assume a temperature of say
T_prime = 700;//[K]
fault1=10;

while(fault1>0.5)
    Tr_prime = T_prime/Tc;
    B0_prime = 0.083 - 0.422/(Tr_prime^(1.6));
    B1_prime = 0.139 - 0.172/(Tr_prime^(4.2));
    // (B*Pc)/(R*Tc) = B0 + w*B1
    B_prime = ((B0_prime + w*B1_prime)*(R*Tc))/Pc;//
        [m^(3)/mol] - Second virial coefficient at
       state 2
    dB0_dT_prime = 0.422*1.6*Tc^(1.6)*T_prime^(-2.6)
       ;// (dB_0/dT)
    dB1_dT_prime = 0.172*4.2*Tc^(4.2)*T_prime^(-5.2)
       ;// (dB_1/dT)
    dB_dT_prime = ((R*Tc)/Pc)*((dB0_dT_prime) + w*(
       dB1_dT_prime));// (dB/dT)_1
    deff('[y]=f1(T_out)','y = 32.2168*log(T_out) +
       0.1922*10^(-2)*T_out + 0.5274*10^(-5)*T_2^(2)
        - 1.1976*10^(-9)*T_out^(3) -8*10^(6)*
       dB_dT_prime -216.64');
    T_out = fsolve(10,f1);
    fault1=abs(T_prime-T_out);
    T_prime = T_prime + 0.5;
end
T_out = T_prime;
```

```

// Now we have to calculate enthalpy change as W = -
   delta_H
delta_H_3 = integrate('(7.7 + 0.04594*10^(-2)*T +
   0.2521*10^(-5)*T^(2) - 0.8587*10^(-9)*T^(3))
   *4.184','T',T_1,T_out) + B_prime*P_2 - P_2*T_out*
   dB_dT_prime - B_1*P_1 + P_1*T_1*dB_dT_1;//[J/mol]

W_3 = - delta_H_3;// [J/mol]

printf(" (3).The exit temperature is %f K\n",T_out);
printf("     The required amount of work is %f J/mol
   \n\n",W_3);

//(4)
n = 0.8;// Adiabatic efficiency
delta_H_a_4 = delta_H_3/n;//[J/mol]
W_4 = -delta_H_a_4;//[J/mol]

// Now we have to determine the exit temperature
   when the enthalpy is delta_H_a_4
// 7.7*4.184*(T_2-T_1) + ((0.04594*4.184*10^(-2))/2)
   *(T_2^(2)-T_1^(2)) + ((0.2521*4.184*10^(-5))/3)*(
   T_2^(3)-T_1^(3)) - ((0.8587*4.184*10^(-9))/4)*(
   T_2^(4)-T_1^   (4)) + B_2*8*10^(6) - 8*10^(6)*T_2
   *(dB/dT)_2 + 191.7 + 496.81 = delta_H_a_4

// Let us assume a temperature of say
T_prime1 = 700;//[K]
fault2=10;

while(fault2>0.5)
    Tr_prime1 = T_prime1/Tc;
    B0_prime1 = 0.083 - 0.422/(Tr_prime1^(1.6));
    B1_prime1 = 0.139 - 0.172/(Tr_prime1^(4.2));
    // (B*Pc)/(R*Tc) = B0 + w*B1
    B_prime1 = ((B0_prime1 + w*B1_prime1)*(R*Tc))/Pc
       ;// [m^(3)/mol] - Second virial coefficient
```

```
        at state 2
    dB0_dT_prime1 = 0.422*1.6*Tc^(1.6)*T_prime1
        ^(-2.6);// (dB_0/dT)
    dB1_dT_prime1 = 0.172*4.2*Tc^(4.2)*T_prime1
        ^(-5.2);// (dB_1/dT)
    dB_dT_prime1 = ((R*Tc)/Pc)*((dB0_dT_prime1) + w
        *(dB1_dT_prime1));// (dB/dT)_1
    deff('[y]=f2(T_2)','y = 7.7*4.184*(T_2-T_1) +
        ((0.04594*4.184*10^(-2))/2)*(T_2^(2)-T_1^(2))
         + ((0.2521*4.184*10^(-5))/3)*(T_2^(3)-T_1
        ^(3)) - ((0.8587*4.184*10^(-9))/4)*(T_2^(4)-
        T_1^(4)) + B_prime1*8*10^(6) - 8*10^(6)*T_2*
        dB_dT_prime1 + 191.7 + 496.81 - delta_H_a_4')
        ;
    T_out1 = fsolve(100,f2);
    fault2=abs(T_prime1-T_out1);
    T_prime1 = T_prime1 + 0.5;
end
T_out1 = T_prime1;

printf(" (4).The exit temperature is %f K\n",T_out1)
    ;
printf("      The required amount of work is %f J/mol
    \n\n",W_4);
```

Scilab code Exa 10.11 Calculation of temperature and pressure

```
clear;
clc;
funcprot(0);

//Example - 10.11
//Page number - 348
printf("Example - 10.11 and Page number - 348\n\n");

```

```
//Given
Vol = 0.15;//[m^(3)] - Volume of the cylinder
P_1 = 100;//[bar] - Initial pressure
P_1 = P_1*10^(5);//[Pa]
T_1 = 170;//[K] - Initial temperature
n_withdrawn = 500;//[mol] - Withdrawn number of
   moles
R = 8.314;//[J/mol*K] - Universal gas constant


//(1)
Y = 1.4;// Coefficient of adiabatic expansion
n_total = (P_1*Vol)/(R*T_1);//[mol] - Total number
   of moles
n_2 = n_total - n_withdrawn;//[mol] - Left number of
    moles
V_1 = Vol/n_total;//[m^(3)/mol] - Molar volume at
   initial state.
// At final state
V_2 = Vol/n_2;//[m^(3)/mol] - Molar volume at final
   state

// During duscharging  P_1*V_1^(Y) = P_2*V_2^(Y),
   therefore
P_2_1 = P_1*((V_1/V_2)^(Y));//[Pa] - Final pressure
P_2_1 = P_2_1*10^(-5);//[bar]
T_2_1 = ((P_2_1*10^(5))*V_2)/R;//[K] - Final
   temperature

printf(" (1).The final temperature %f K\n",T_2_1);
printf("     The final pressure %f bar\n\n",P_2_1);

//(2)
// Cp_0 = 27.2 + 4.2*10^(-3)*T
// For a discharge process entropy per mol of the
   gas that remains in the cylinder is constant,
   delta_S = 0
// Therefore foe one mol of ideal gas   integrate('
    Cp_0/T','T',T_1,T_2) - R*log(P_2/P_1) = 0
// Since the gas is assumed to be ideal,therefore
    P_2*Vol = n_2*R*T_2
// P_2 = (n_2*R*T_2)/V_2. Substituting in above
    equation after simplification we get
// 27.2*log(T_2/T_1) + 4.2*10^(-3)*(T_2 - T_1) - R*
    log(P_2/P_1)
// f(T) = 18.886*log(T) + 4.2*10^(-3)*T - 92.4
// f(T)_dash = 18.886/T + 4.2*10^(-3)   // Derivative
     of the above function

// Starting with a temperature of 150 K
T_prime = 150;//[K]
error = 10;
while(error>1)
    f_T = 18.886*log(T_prime) + 4.2*10^(-3)*T_prime
        - 92.4;
    f_T_dash = 18.886/T_prime + 4.2*10^(-3);
    T_new = T_prime - (f_T/f_T_dash);
    error=abs(T_prime - T_new);
    T_prime = T_new;
end

T_2_2 = T_prime;//[K] - Final temperature
P_2_2 = ((n_2*R*T_2_2)/Vol)*10^(-5);//[bar] - Final
    pressure

printf(" (2).The final temperature %f K\n",T_2_2);
printf("     The final pressure %f bar\n\n",P_2_2);

//(3)
Tc = 126.2;//[K] - Critical temperature of nitrogen
Pc = 34.0;//[bar] - Critical pressure of nitrogen
Pc = Pc*10^(5);//[Pa]
w = 0.038;// Acentric factor

// Virial equation of state , Z = 1 + (B*P)/(R*T)
// S_R = -P*(dB/dT)
```

```

dB0_dT = 0.422*1.6*Tc^(1.6)*T_1^(-2.6);// (dB_0/dT)
    at state 1
dB1_dT = 0.172*4.2*Tc^(4.2)*T_1^(-5.2);// (dB_1/dT)
    at state 1
dB_dT = ((R*Tc)/Pc)*((dB0_dT) + w*(dB1_dT));// (dB/
   dT) at state 1
// The residual entropy at the initial state is
    given by
S_R_1 = -P_1*(dB_dT);//[J/mol-K]

// Now let us calculate molar volume at initial
    state
Tr = T_1/Tc;// Reduced temperature
B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));

//We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
B = ((B_0+(w*B_1))*(R*Tc))/Pc;//[m^(3)/mol]

V_1_3 = B + (R*T_1)/P_1;//[m^(3)/mol]
// Therefore number of moles in the initial state is
n_1_3 = Vol/V_1_3;//[mol]
// Therefore final number of moles is
n_2_3 = n_1_3 - n_withdrawn;

// Therefore molar volume at final state is
V_2_3 = Vol/n_2_3;//[m^(3)/mol]

// Now let us determine the relation between
    pressure and temperature in the final state
// P_2_3 = (R*T_2_3)/(V_2_3 - B_2)
//delta_S = 0, thus delta_S_ig + delta_S_R = 0
delta_S_R = - S_R_1;
// integrate('Cp_0/T','T',T_1,T_2) - R*log(P_2/P_1)
   - P_2*(dB/dT)_2 + S_R_1
// On simplification,
// delta_S = 27.2*(log(T_2_prime/T_1)) + 4.2*10^(-3)
```

```
    *(T_2_prime - T_1) - R*(log(P_2_3/P_1)) - P_2_3*(
    dB_dT_3) + delta_S_R
// Starting with a temperature of 135 K

T_2_prime = 135;//[K]
delta = 0.1;
error = 10;
while(error>0.01)
    T_r = T_2_prime/Tc;// Reduced temperature
    B_0_3 = 0.083-(0.422/(T_r)^(1.6));
    B_1_3 = 0.139-(0.172/(T_r)^(4.2));
    B_3 = ((B_0_3+(w*B_1_3))*(R*Tc))/Pc;//[m^(3)/mol
       ]
    dB0_dT_3 = 0.422*1.6*Tc^(1.6)*T_2_prime^(-2.6);
       // (dB_0/dT)
    dB1_dT_3 = 0.172*4.2*Tc^(4.2)*T_2_prime^(-5.2);
       // (dB_1/dT)
    dB_dT_3 = ((R*Tc)/Pc)*((dB0_dT_3) + w*(dB1_dT_3)
       );// (dB/dT)
    P_2_3 = (R*T_2_prime)/(V_2_3 - B_3);
    delta_S = 27.2*(log(T_2_prime/T_1)) +
       4.2*10^(-3)*(T_2_prime - T_1) - R*(log(P_2_3/
       P_1)) - P_2_3*(dB_dT_3) + delta_S_R;
    T_new = T_2_prime + delta;
    error=abs(delta_S);
    T_2_prime = T_new;
end

T_2_3 = T_2_prime;//[K] - Final temperature
// Therefore at T_2_3
P_2_3 = P_2_3*10^(-5);//[bar] - Final pressure

printf(" (3).The final temperature %f K\n",T_2_3);
printf("     The final pressure %f bar\n\n",P_2_3);
```

Scilab code Exa 10.12 Calculation of change of internal energy enthalpy entropy and exergy

```
clear;
clc;
funcprot(0);

//Example - 10.12
//Page number - 351
printf("Example - 10.12 and Page number - 351\n\n");

//Given
P_1 = 80;//[bar] - Initial pressure
P_1 = P_1*10^(5);//[Pa]
T_1 = 300 + 273.15;//[T] - Initial temperature
P_2 = 40;//[bar] - Final pressure
P_2 = P_2*10^(5);//[Pa]
T_2 = 300 + 273.15;//[K] - Final temperature
T_0 = 25 + 273.15;//[K] - Surrounding temperature
P_0 = 1;//[atm] - Surrounding pressure
P_0 = P_0*101325;//[Pa]
Tc = 647.1;//[K]
Pc = 220.55;//[bar]
Pc = Pc*10^(5);//[Pa]
R = 8.314;//[J/mol*K] - Universal gas constant

// For van der Walls equation of state
a = (27*R^(2)*Tc^(2))/(64*Pc);//[Pa-m^(6)/mol^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]

// The cubic form of van der Walls equation of state
    is given by,
// V^(3) - (b + (R*T)/P)*V^(2) + (a/P)*V - (a*b)/P =
    0

// Solving the cubic equation
// At 80 bar and 300 K
deff('[y]=f(V)','y=V^(3)-(b+(R*T_1)/P_1)*V^(2)+(a/
    P_1)*V-(a*b)/P_1');
V_1_1=fsolve(0.1,f);
V_1_2=fsolve(10,f);
V_1_2=fsolve(100,f);
// The largest root is considered because of vapour
V_1 = V_1_1;

U_R_1 = -a/V_1;//[J/mol] - Internal energy
H_R_1 = P_1*V_1 - R*T_1 - a/V_1;//[J/mol] - Enthalpy
S_R_1 = R*log((P_1*(V_1-b))/(R*T_1));

// Now let us calculate the residual properties at
    state 2
// At 40 bar and 300 K
deff('[y]=f1(V)','y=V^(3)-(b+(R*T_2)/P_2)*V^(2)+(a/
    P_2)*V-(a*b)/P_2');
V_2_1 = fsolve(0.1,f1);
V_2_2 = fsolve(10,f1);
V_2_3 = fsolve(100,f1);
// The above equation has 1 real and 2 imaginary
    roots. We consider only real root.
V_2 = V_2_1;

U_R_2 = -a/V_2;//[J/mol] - Internal energy
H_R_2 = P_2*V_2 - R*T_2 - a/V_2;//[J/mol] - Enthalpy
S_R_2 = R*log((P_2*(V_2-b))/(R*T_2));

delta_U_R = U_R_2 - U_R_1;//
delta_H_R = H_R_2 - H_R_1;//
delta_S_R = S_R_2 - S_R_1;//

delta_U_ig = 0;//[J/mol] - As temperature is
    constant
delta_H_ig = 0;//[J/mol] - As temperature is
    constant
// delta_S_ig = Cp_0*log(T_2/T_1) - R*log(P_2/P_1)
    ;// [J/mol-K]
// Since T_1 = T_2
// Therefore
delta_S_ig = - R*log(P_2/P_1);// [J/mol-K]
delta_U = delta_U_R + delta_U_ig;//[J/mol]
delta_H = delta_H_R + delta_H_ig;//[J/mol]
delta_S = delta_S_R + delta_S_ig;//[J/mol-K]

// Change in exergy is given by
// delta_phi = phi_1 - phi_2 = U_1 - U_2 + P_0*(V_1
   - _V_2) - T_0*(S_1 - S_2)
delta_phi = - delta_U + P_0*(V_1 - V_2) - T_0*(-
   delta_S);//[J/mol]

printf(" The change in internal energy is %f J/mol\n
   ",delta_U);
printf(" The change in enthalpy is %f J/mol\n",
   delta_H);
printf(" The change in entropy is %f J/mol-K\n",
   delta_S);
printf(" The change in exergy is %f J/mol\n",
   delta_phi);
```

Scilab code Exa 10.13 Calculation of change in enthalpy

```
clear;
clc;

//Example - 10.13
//Page number - 353
printf("Example - 10.13 and Page number - 353\n\n");

//Given
T_1 = 500;//[K] - Initial temperature
P_1 = 30;//[atm] - Initial pressure
P_1 = P_1*101325;//[Pa]
P_2 = 1;//[atm] - Final pressure
```

```
P_2 = P_2*101325;//[Pa]
R = 8.314;//[J/mol*K] - Universal gas constant
// For chlorine
Tc = 417.2;//[K] - Critical temperature
Pc = 77.10;//[bar] - Critical pressure
Pc = Pc*10^(5);//[Pa]

//Redlich Kwong equation of state,
a = (0.42748*(R^(2))*(Tc^(2.5)))/Pc;// [Pa*m^(6)*K
    ^(1/2)/mol]
b = (0.08664*R*Tc)/Pc;// [m^(3)/mol]

// The cubic form of Redlich Kwong equation of state
     is given by,
// V^(3)-((R*T)/P)*V^(2)-((b_1^(2))+((b_1*R*T)/P)-(a
    /(T^(1/2)*P))*V-(a*b)/(T^(1/2)*P)=0
//Solving the cubic equation
// At state 1 (500 K, 30 atm)
deff('[y]=f1(V)','y=V^(3)-((R*T_1)/P_1)*V^(2)-((b
    ^(2))+((b*R*T_1)/P_1)-(a/(T_1^(1/2)*P_1)))*V-(a*b
    )/(T_1^(1/2)*P_1)');
V_1=fsolve(1,f1);
V_2=fsolve(10,f1);
V_3=fsolve(100,f1);
// The above equation has 1 real and 2 imaginary
    roots. We consider only real root,
V = V_1;//[m^(3)/mol]

// Thus compressibility factor is
Z = (P_1*V_1)/(R*T_1);//compressibility factor

// The residual enthalpy at state 1 is given by
H_R_1 = (Z-1)*R*T_1 + ((3*a)/(2*b*T_1^(1/2)))*(log(V
    /(V+b)));//[J/mol]

// Since chlorine is assumed to behave ideally under
     the final condition,therefore
H_R_2 = 0;// Residual enthalpy at state 2
```

```
delta_H_R = H_R_2 - H_R_1;//[J/mol] - Residual
   enthalpy change
// and since isothermal conditions are maintained,
   therfore
delta_H_ig = 0;// Enthalpy change under ideal
   condition
delta_H = delta_H_R + delta_H_ig;//[J/mol]

printf(" The change in enthalpy is given by, delta_H
    = %f J/mol\n",delta_H);
```

Scilab code Exa 10.14 Calculation of final temperature

```
clear;
clc;

//Example - 10.14
//Page number - 353
printf("Example - 10.14 and Page number - 353\n\n");

//(1)
//This part involves proving a relation in which no
   mathematical components are involved.
//For prove refer to this example 10.14 on page
   number 354 of the book.
printf(" (1).This part involves proving a relation
   in which no mathematical components are involved
   .\n");
printf("      For prove refer to this example 10.14
   on page number 354 of the book.\n\n")

//(2)
//Given
Vol_1 = 0.1;//[m^(3)] - Initial volume of each
   compartment
```

```
n_1 = 400;//[mol] - Initial number of moles in
    compartment 1
V_1 = Vol_1/n_1;//[m^(3)/mol] - Molar volume at
    state 1
T_1 = 294;//[K]
Vol_2 = 0.2;//[m^(3)] - Final volume of the
    compartment after removing the partition.
n_2 = n_1;//[mol] - Number of moles remains the same
V_2 = Vol_2/n_2;//[m^(3)/mol] - Molar volume at
    state 2

// For argon
a = 0.1362;//[Pa-m^(6)/mol^(2)]
b = 3.215*10^(-5);//[m^(3)/mol]
Cv_0 = 12.56;//[J/mol-K] - Heat capacity in ideal
    gas state

// For overall system q = 0, and no work is done,
    therefore delta_U = 0
// Therfore from the relation proved in part (1), we
     have
T_2 = T_1 + (a/Cv_0)*(1/V_2 - 1/V_1);//[K]

printf(" (2).The final temperatutre is %f K\n",T_2)
```

Scilab code Exa 10.15 Proving a mathematical relation

```
clear;
clc;

//Example - 10.15
//Page number - 354
printf("Example - 10.15 and Page number - 354\n\n");

//This problem involves proving a relation in which
```

```
    no mathematical components are involved.
//For prove refer to this example 10.15 on page
    number 354 of the book.
printf(" This problem involves proving a relation in
     which no mathematical components are involved.\n
    \n");
printf(" For prove refer to this example 10.15 on
    page number 354 of the book.")
```

Scilab code Exa 10.16 Proving a mathematical relation

```
clear;
clc;

//Example - 10.16
//Page number - 355
printf("Example - 10.16 and Page number - 355\n\n");

//This problem involves proving a relation in which
    no mathematical components are involved.
//For prove refer to this example 10.16 on page
    number 355 of the book.
printf(" This problem involves proving a relation in
     which no mathematical components are involved.\n
    \n");
printf(" For prove refer to this example 10.16 on
    page number 355 of the book.")
```

Scilab code Exa 10.17 Determination of work done and the exit temperature

```
clear;
clc;
```

```
funcprot(0);

//Example - 10.17
//Page number - 356
printf("Example - 10.17 and Page number - 356\n\n");

//Given
P_1 = 1*10^(6);//[Pa] - Initial pressure
T_1 = 200 + 273.15;//[K] - Initial temperature
P_2 = 8*10^(6);//[Pa]
R = 8.314;//[J/mol*K] - Universal gas constant
Y = 1.4;// Index of expansion
Cp_0 = 29.114;//[J/mol-K]
// For H20, the van der Walls constants are
a = 0.55366;//[Pa-m^(6)/mol^(2)]
b = 3.049*10^(-5);//[m^(3)/mol]

// At state 1 (200 C, 1 MPa)
// The molar volume of steam following van der Walls
    equation of state (as reported in the book) is
V_1 = 3.816*10^(-3);//[m^(3)/mol]
// And the compressibility factor is
Z_1 = (P_1*V_1)/(R*T_1);

// Assuming ideal gas behaviour the exit temperature
    is given by
T_2 = T_1*(P_2/P_1)^((Y-1)/Y);//[K]

// At 8 MPa and T_2,
// The molar volume of steam following van der Walls
    equation of state (as reported in the book) is
V_2 = 8.41*10^(-4);//[m^(3)/mol]
// And the compressibility factor is
Z_2 = (P_2*V_2)/(R*T_2);

// For van der Walls equation of state we know that
// delta_S_R/R = log(Z_2/Z_1) + log((V_2 - b)/V_2) -
    log((V_1 - b)/V_1)
```

```
delta_S_R = R*(log(Z_2/Z_1) + log((V_2 - b)/V_2) -
   log((V_1 - b)/V_1));//[J/mol]

// delta_S_ig = Cp_0*log(T_2/T_1) - R*log(P_2/P_1)
// The entropy change is therefore
// delta_S = delta_S_ig + delta_S_R
// But during an isentropic process the total
   entropy change is zero
// Therefore we have to modify the exit temperature
   so that the entropy change is zero

// Let us assume a temperature, say T = 870 K
// At 870 K the molar volume of steam following van
   der Walls equation of state (as reported in the
   book) is
//  V_3 = 8.57*10^(-4);// [m^(3)/mol]
// Therefore
// Z_3 = (P_2*V_3)/(R*T_2);
// At this temperature,
// delta_S = Cp_0*log(T/T_1) - R*log(P_2/P_1) + R*(
   log(Z/Z_1) + R*log((V - b)/V) - R*log((V_1 - b)/
   V_1))

T = 800;//[K]
fault=10;

while(fault>0.3)
    // At T and 8 MPa
    deff('[y]=f1(V)','y=V^(3)-(b+(R*T)/P_2)*V^(2)+(a
       /P_2)*V-(a*b)/P_2');
    V = fsolve(1,f1);
    Z = (P_2*V)/(R*T);

    deff('[y]=f1(T)','y = Cp_0*log(T/T_1) - R*log(
       P_2/P_1) + R*(log(Z/Z_1) + R*(log((V - b)/V))
        - R*(log((V_1 - b)/V_1)))');
    T_exit = fsolve(0.1,f1);
    fault=abs(T-T_exit);
    T = T + 0.5;
end
Texit = T;

// Now applying the first law to an adiabatic
   process we get
// W = - delta_H

// For van der Walls gas the enthalpy change is
   given by
delta_H_s = Cp_0*(T_exit - T_1) + (Z - 1)*R*T_exit -
    a/V - (Z_1-1)*R*T_1 + a/V_1;//[J/mol]
W = - delta_H_s;//[J/mol]

printf(" (1).The exit temperature is %f K\n",Texit);
printf("      The work required is given by, W = %f J
   /mol\n\n",W);

//(2)
eff = 0.8;// Adiabatic efficiency
delta_H_a = eff*delta_H_s;//[J/mol] - Actual
   enthalpy change
W_2 = - delta_H_a;

// Let us assume a temperature, say
T_prime= 900;//[K]
fault1=10;

while(fault1>0.3)
    // At T_prime and 8 MPa
    deff('[y]=f2(V)','y=V^(3)-(b+(R*T_prime)/P_2)*V
       ^(2)+(a/P_2)*V-(a*b)/P_2');
    V_prime=fsolve(1,f2);
    Z_prime = (P_2*V_prime)/(R*T_prime);

    deff('[y]=f3(T_prime)','y = Cp_0*(T_prime - T_1)
        + (Z_prime - 1)*R*T_prime - a/V_prime -
       13230.49');
    T_exit1 = fsolve(100,f3);
    fault1=abs(T_prime-T_exit1);
    T_prime = T_prime + 0.2;
end
Texit1 = T_prime;

printf(" (2).The exit temperature is %f K\n",Texit1)
   ;
printf("      The work required is given by, W = %f J
   /mol\n\n",W_2);
```

Scilab code Exa 10.18 Proving a mathematical relation

```
clear;
clc;

//Example - 10.18
//Page number - 358
printf("Example - 10.18 and Page number - 358\n\n");

//This problem involves proving a relation in which
   no mathematical components are involved.
//For prove refer to this example 10.18 on page
   number 358 of the book.
printf(" This problem involves proving a relation in
    which no mathematical components are involved.\n
   \n");
printf(" For prove refer to this example 10.18 on
   page number 358 of the book.")
```

Scilab code Exa 10.19 Calculation of molar volume and fugacity

```
clear;
```

```
clc;
funcprot(0);

//Example - 10.19
//Page number - 359
printf("Example - 10.19 and Page number - 359\n\n");

//Given
T = 100 + 273.15;//[K] - Temperature
Tc = 647.1;//[K] - Critical temperature of water
Pc = 220.55;//[bar] - Critical pressure of water
Pc = Pc*10^(5);//[Pa]
R = 8.314;//[J/mol*K] - Universal gas constant

// For van der Walls equation of state
a = (27*R^(2)*Tc^(2))/(64*Pc);//[Pa-m^(6)/mol^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]

// The cubic form of van der Walls equation of state
    is given by,
// V^(3) - (b + (R*T)/P)*V^(2) + (a/P)*V - (a*b)/P =
    0

// For water vapour at 100 C under saturated
   conditions pressure is 1 atm, therefore
P = 1;//[atm]
P = P*101325;//[Pa]

// At 100 C and 1 atm
deff('[y]=f(V)','y=V^(3)-(b+(R*T)/P)*V^(2)+(a/P)*V-(
   a*b)/P');
V_1 = fsolve(0.1,f);
V_1 = fsolve(10,f);
V_1 = fsolve(100,f);
// The largest root is considered because of molar
   volume of vapour phase is to determined
V = V_1;//[m^(3)/mol]

```

```
// Now the figacity is given by
// log(f/P) = log((R*T)/(P*(V-b))) + b/(V-b) - (2*a)
   /(R*T*V);
f = P*(exp(log((R*T)/(P*(V-b))) + b/(V-b) - (2*a)/(R
   *T*V)));//[Pa]
f = f/101325;//[atm]

printf(" The molar volume is %f m^(3)/mol\n\n",V);
printf(" The fugacity is %f atm\n\n",f);
```

Scilab code Exa 10.20 Calculation of enthalpy and entropy change

```
clear;
clc;
funcprot(0);

//Example - 10.20
//Page number - 359
printf("Example - 10.20 and Page number - 359\n\n");

//Given
P_1 = 6;//[bar] - Initial pressure
P_1 = P_1*10^(5);//[Pa]
T_1 = 100 + 273.15;//[T] - Initial temperature
P_2 = 12;//[bar] - Final pressure
P_2 = P_2*10^(5);//[Pa]
T_2 = 500 + 273.15;//[K] - Final temperature
R = 8.314;//[J/mol*K] - Universal gas constant
Y = 1.126;// Index of expansion
Cp_0 = (R*Y)/(Y-1);//[J/mol-K]

// For propane
Tc = 369.8;//[K]
Pc = 42.48;//[bar]
Pc = Pc*10^(5);
```

```
w = 0.152;

//(1)
// For van der Walls equation of state
a = (27*R^(2)*Tc^(2))/(64*Pc);//[Pa-m^(6)/mol^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]

// The cubic form of van der Walls equation of state
    is given by,
// V^(3) - (b + (R*T)/P)*V^(2) + (a/P)*V - (a*b)/P =
    0

// At state 1 (100 C and 6 bar)
deff('[y]=f(V)','y=V^(3)-(b+(R*T_1)/P_1)*V^(2)+(a/
    P_1)*V-(a*b)/P_1');
V_1_1 = fsolve(1,f);
V_1_2 = fsolve(10,f);
V_1_3 = fsolve(100,f);
// The largest root is considered because of molar
    volume of vapour phase is to determined
V_1 = V_1_1;//[m^(3)/mol]
// Thus compressibility factor is
Z_1 = (P_1*V_1)/(R*T_1);//compressibility factor

H_R_1 = (Z_1 - 1)*R*T_1 - (a/V_1);// [J/mol]
S_R_1 = R*log((P_1*(V_1-b))/(R*T_1));// [J/mol-K]

// At state 2 (500 C and 12 bar)
deff('[y]=f1(V)','y=V^(3)-(b+(R*T_2)/P_2)*V^(2)+(a/
    P_2)*V-(a*b)/P_2');
V_2_1 = fsolve(1,f1);
V_2_2 = fsolve(10,f1);
V_2_3 = fsolve(100,f1);
// The largest root is considered because of molar
    volume of vapour phase is to determined
V_2 = V_2_1;//[m^(3)/mol]
// Thus compressibility factor is
Z_2 = (P_2*V_2)/(R*T_2);//compressibility factor
```

```

H_R_2 = (Z_2 - 1)*R*T_2 - (a/V_2);// [J/mol]
S_R_2 = R*log((P_2*(V_2-b))/(R*T_2));// [J/mol-K]

// Ideal gas entropy change is given by
delta_S_ig = Cp_0*log(T_2/T_1) - R*log(P_2/P_1);//[J
    /mol-K]
// Entropy change is given by
delta_S = delta_S_ig + (S_R_2 - S_R_1);//[J/mol-k]

// Ideal gas enthalpy change is given by
delta_H_ig = Cp_0*(T_2 - T_1);//[J/mol]
// Enthalpy change is given by
delta_H = delta_H_ig + (H_R_2 - H_R_1);//[J/mol]

printf("(1).The change in enthalpy is %f J/mol\n",
    delta_H);
printf("    The change in entropy is %f J/mol-K\n\n"
    ,delta_S);

//(2)
// Virial equation of state

// At state 1 (372.15 K, 6 bar) let us calculate B
    and dB/dT
Tr = T_1/Tc;// Reduced temperature
B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));

//We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
B = ((B_0+(w*B_1))*(R*Tc))/Pc;//[m^(3)/mol]
dB0_dT = 0.422*1.6*Tc^(1.6)*T_1^(-2.6);// (dB_0/dT)
    at state 1
dB1_dT = 0.172*4.2*Tc^(4.2)*T_1^(-5.2);// (dB_1/dT)
    at state 1
dB_dT = ((R*Tc)/Pc)*((dB0_dT) + w*(dB1_dT));// (dB/
    dT) at state 1

```

```
H_R_1_2 = B*P_1 - P_1*T_1*dB_dT;//[J/mol] - Residual
    enthalpy at state 1
S_R_1_2 = -P_1*(dB_dT);//[J/mol-K] - Residual
   entropy at state 1

// At state 2 (773.15 K, 12 bar)
Tr_2 = T_2/Tc;// Reduced temperature
B_0_2 = 0.083-(0.422/(Tr_2)^(1.6));
B_1_2 = 0.139-(0.172/(Tr_2)^(4.2));

//We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
B_2 = ((B_0_2+(w*B_1_2))*(R*Tc))/Pc;//[m^(3)/mol]
dB0_dT_2 = 0.422*1.6*Tc^(1.6)*T_2^(-2.6);// (dB_0/dT
   ) at state 1
dB1_dT_2 = 0.172*4.2*Tc^(4.2)*T_2^(-5.2);// (dB_1/dT
   ) at state 1
dB_dT_2 = ((R*Tc)/Pc)*((dB0_dT_2) + w*(dB1_dT_2));//
    (dB/dT) at state 1

H_R_2_2 = B_2*P_2 - P_2*T_2*dB_dT_2;//[J/mol] -
   Residual enthalpy at state 1
S_R_2_2 = -P_2*(dB_dT_2);//[J/mol-K] - Residual
   entropy at state 1

delta_H_2 = delta_H_ig + (H_R_2_2 - H_R_1_2);//[J/
   mol]
delta_S_2 = delta_S_ig + (S_R_2_2 - S_R_1_2);//[J/
   mol]

printf("(2).The change in enthalpy is %f J/mol\n",
   delta_H_2);
printf("    The change in entropy is %f J/mol-K\n",
   delta_S_2);
```

Scilab code Exa 10.21 Calculation of fugacity

```
clear;
clc;

//Example - 10.21
//Page number - 362
printf("Example - 10.21 and Page number - 362\n\n");

//Given
P = 2.76*10^(6);//[N/m^(2)] - Pressure
T = 310.93;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant

// For n-butane
Tc = 425.18;//[K] - Critical temperature
Pc = 37.97;//[bar] - Critical pressure
Pc = Pc*10^(5);//[Pa]
w = 0.193;
den = 0.61;//[g/cm^(3)]
mol_wt = 58;//[g/mol] - Molecular weight of butane

// log(P_sat) = 15.7374 - 2151.63/(T-36.24)
P_sat = exp(15.7374 - 2151.63/(T-36.24));//[mm Hg]
P_sat = (P_sat/760)*101325;//[N/m^(2)]

//(1)
// Let us determine the second virial coefficient at
    310.93 K
Tr = T/Tc;// Reduced temperature
B_0 = 0.083-(0.422/(Tr)^(1.6));
B_1 = 0.139-(0.172/(Tr)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0 + (w*B_1)
B = ((B_0+(w*B_1))*(R*Tc))/Pc;//[m^(3)/mol]

// Fugacity under saturated conditions is given by
// log(f_sat/P_sat) = (B*P_sat)/(R*T)
f_sat = P_sat*(exp((B*P_sat)/(R*T)));//[N/m^(2)]

// The molar volume is given by
```

```
V_liq = (1/(den*1000))*(mol_wt/1000);//[m^(3)/mol]

f = f_sat*exp(V_liq*(P-P_sat)/(R*T));

printf(" (1).The fugacity of n-butane is %e N/m^(2)\
   n\n",f);

//(2)
// For van der Walls equation of state
a = (27*R^(2)*Tc^(2))/(64*Pc);//[Pa-m^(6)/mol^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]

// The cubic form of van der Walls equation of state
    is given by,
// V^(3) - (b + (R*T)/P)*V^(2) + (a/P)*V - (a*b)/P =
    0

// At 100 C and 1 atm
deff('[y]=f(V)','y=V^(3)-(b+(R*T)/P)*V^(2)+(a/P)*V-(
   a*b)/P');
V_1 = fsolve(0.1,f);
V_1 = fsolve(10,f);
V_1 = fsolve(100,f);
// The above equation has only 1 real root, other
   two roots are imaginary
V = V_1;//[m^(3)/mol]

// log(f/P) = log((R*T)/(P*(V-b))) + b/(V-b) -(2*a)
   /(R*T*V)
f_2 = P*(exp(log((R*T)/(P*(V-b))) + b/(V-b) -(2*a)/(
   R*T*V)));

printf(" (2).The fugacity of n-butane is %e N/m^(2)\
   n\n",f_2);
```

Scilab code Exa 10.22 Calculation of enthalpy change

```
clear;
clc;

//Example - 10.22
//Page number - 363
printf("Example - 10.22 and Page number - 363\n\n");

//Given
T = 50+273.15;//[K] - Temperature
P = 25*10^(3);//[Pa] - Pressure
y1 = 0.5;//[mol] - mole fraction of equimolar
    mixture
y2 = 0.5;
R = 8.314;//[J/mol*K] - Universal gas constant

//For component 1 (methyl ethyl ketone)
Tc_1  = 535.5;//[K] - Critical temperature
Pc_1 = 41.5*10^(5);//[N/m^(2)] - Critical pressure
Vc_1 = 267;//[cm^(3)/mol] - Critical volume
Zc_1 = 0.249;// Critical compressibility factor
w_1 = 0.323;// acentric factor

//For component 2 (toluene)
Tc_2 = 591.8;//[K]
Pc_2 = 41.06*10^(5);//[N/m^(2)]
Vc_2 = 316;//[cm^(3)/mol]
Zc_2 = 0.264;
w_2 = 0.262;

// For equation of state Z = 1 + B/V
//For component 1, let us calculate B and dB/dT
Tr_1 = T/Tc_1;//Reduced temperature
//At reduced temperature
B1_0 = 0.083-(0.422/(Tr_1)^(1.6));
B1_1 = 0.139-(0.172/(Tr_1)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
```

```
B_11 = ((B1_0+(w_1*B1_1))*(R*Tc_1))/Pc_1;// [m^(3)/
   mol-K]
dB0_dT_1 = 0.422*1.6*Tc_1^(1.6)*T^(-2.6);// [m^(3)/
   mol-K] - (dB_0/dT)
dB1_dT_1 = 0.172*4.2*Tc_1^(4.2)*T^(-5.2);// [m^(3)/
   mol-K] - (dB_1/dT)
dB_dT_1 = ((R*Tc_1)/Pc_1)*((dB0_dT_1) + w_1*(
   dB1_dT_1));//[m^(3)/mol-K] - (dB/dT)_

//Similarly for component 2
Tr_2 = T/Tc_2;//Reduced temperature
//At reduced temperature Tr_2,
B2_0 = 0.083 - (0.422/(Tr_2)^(1.6));
B2_1 = 0.139 - (0.172/(Tr_2)^(4.2));
B_22 = ((B2_0+(w_2*B2_1))*(R*Tc_2))/Pc_2;//[m^(3)/
   mol]
dB0_dT_2 = 0.422*1.6*Tc_2^(1.6)*T^(-2.6);// [m^(3)/
   mol-K] - (dB_0/dT)
dB1_dT_2 = 0.172*4.2*Tc_2^(4.2)*T^(-5.2);// [m^(3)/
   mol-K] - (dB_1/dT)
dB_dT_2 = ((R*Tc_2)/Pc_2)*((dB0_dT_2) + w_2*(
   dB1_dT_2));//[m^(3)/mol-K] - (dB/dT)_

//For cross coeffcient, let us calculate B and dB/dT
Tc_12 = (Tc_1*Tc_2)^(1/2);//[K]
w_12 = (w_1 + w_2)/2;
Zc_12 = (Zc_1 + Zc_2)/2;
Vc_12 = (((Vc_1)^(1/3) + (Vc_2)^(1/3))/2)^(3);//[cm
   ^(3)/mol]
Vc_12 = Vc_12*10^(-6);//[m^(3)/mol]
Pc_12 = (Zc_12*R*Tc_12)/Vc_12;//[N/m^(2)]

//Now we have,(B_12*Pc_12)/(R*Tc_12) = B_0+(w_12*B_1
   )
//where B_0 and B_1 are to be evaluated at Tr_12
Tr_12 = T/Tc_12;
//At reduced temperature Tr_12
B_0 = 0.083 - (0.422/(Tr_12)^(1.6));
```

```
B_1 = 0.139 - (0.172/(Tr_12)^(4.2));
B_12 = ((B_0 + (w_12*B_1))*R*Tc_12)/Pc_12;//[m^(3)/
    mol]
dB0_dT_12 = 0.422*1.6*Tc_12^(1.6)*T^(-2.6);// [m^(3)
    /mol-K] - (dB_0/dT)
dB1_dT_12 = 0.172*4.2*Tc_12^(4.2)*T^(-5.2);// [m^(3)
    /mol-K] - (dB_1/dT)
dB_dT_12 = ((R*Tc_12)/Pc_12)*((dB0_dT_12) + w_12*(
    dB1_dT_12));//[m^(3)/mol-K] - (dB/dT)_12

//For the mixture
B = y1^(2)*B_11 + 2*y1*y2*B_12 + y2^(2)*B_22;//[m
    ^(3)/moL]

// The equation of state can be written as
// V^(2) - ((R*T)/P) - (B*R*T)/P = 0
// V^(2) - 0.1075*V + 1.737*10^(-4) = 0
deff('[y]=f(V)','y=V^(2) - 0.1075*V + 1.737*10^(-4)'
    );
V1 = fsolve(0.1,f);
V2 = fsolve(1,f);
// We will consider the root which is near to R*T/P
V = V1;
// dB/dT = y_1^(2)*dB_11/dT + y_2^(2)*dB_22/dT + 2*
    y_1*y_2*dB_12/dT
dB_dT = y1^(2)*dB_dT_1 + y2^(2)*dB_dT_2 + 2*y1*y2*
    dB_dT_12;//[m^(3)/mol-K]

// For equation of state Z = 1 + B/V
H_R = (B*R*T)/V - ((R*T^(2))/V)*dB_dT;//[J/mol]

printf(" (1).The value of H_R for the mixture using
    virial equation of state is %f J/mol\n\n",H_R);

//(2)
//  For van der Walls equation of state
a_11 = (27*R^(2)*Tc_1^(2))/(64*Pc_1);//[Pa-m^(6)/mol
    ^(2)]
```

```
a_22 = (27*R^(2)*Tc_2^(2))/(64*Pc_2);//[Pa-m^(6)/mol
    ^(2)]
a_12 = (a_11*a_22)^(1/2);
b_1 = (R*Tc_1)/(8*Pc_1);//[m^(3)/mol]
b_2 = (R*Tc_2)/(8*Pc_2);//[m^(3)/mol]

// For the mixture
a = y1^(2)*a_11 + y2^(2)*a_22 + 2*y1*y2*a_12;//[Pa-m
    ^(6)/mol^(2)]
b = y1*b_1 + y2*b_2;//[m^(3)/mol]

// From the cubic form of van der Walls equation of
    state
deff('[y]=f1(V)','y=V^(3)-(b+(R*T)/P)*V^(2)+(a/P)*V
    -(a*b)/P');
V2_1 = fsolve(0.1,f1);
V2_2 = fsolve(10,f1);
V2_3 = fsolve(100,f1);
// The largest root is considered
V_2 = V2_1;

// The residual enthalpy is given by
H_R_2 = P*V_2 - R*T -a/V_2;//[J/mol]

printf(" (2).The value of H_R for the mixture using
    van der Walls equation of state is %f J/mol\n\n",
    H_R_2);
```

Scilab code Exa 10.23 Calculation of fugacity of water vapour

```
clear;
clc;

//Example - 10.23
//Page number - 366
```

```
printf("Example - 10.23 and Page number - 366\n\n");

//Given
T = 320 + 273.15;//[K]
R = 8.314;//[J/mol*K] - Universal gas constant

// For water
Tc = 647.1;//[K]
Pc = 220.55;//[bar]
Pc = Pc*10^(5);//[Pa]

// The cubic form of Redlich Kwong equation of state
    is given by,
// V^(3) - ((R*T)/P)*V^(2) - ((b_1^(2)) + ((b_1*R*T)
   /P) - (a/(T^(1/2)*P))*V - (a*b)/(T^(1/2)*P) = 0

// At 320 C and 70 bar pressure
P_1 = 70;//[bar]
P_1 = P_1*10^(5);//[Pa]

a = (0.42748*(R^(2))*(Tc^(2.5)))/Pc;//[Pa*m^(6)*K
   ^(1/2)/mol]
b = (0.08664*R*Tc)/Pc;//[m^(3)/mol]
// Solving the cubic equation
deff('[y]=f1(V)','y=V^(3)-((R*T)/P_1)*V^(2)-((b^(2))
   +((b*R*T)/P_1)-(a/(T^(1/2)*P_1)))*V-(a*b)/(T
   ^(1/2)*P_1)');
V1=fsolve(1,f1);
V2=fsolve(10,f1);
V3=fsolve(100,f1);
// The largest root is considered because at 320 C
   and 70 bar vapour phase exists.
V_1 = V1;//[m^(3)/mol]
// Thus compressibility factor is
Z_1 = (P_1*V_1)/(R*T);

// For Redlich-Kwong equation of state
// log(f/P) = Z - 1 - log(V_1/(V_1-b)) + (a/(b*R*(T
    ^(3/2))))*log(V/(V+b))
f_1 = P_1*(exp(Z_1-1-log(Z_1)+log(V_1/(V_1-b))+(a/(b
    *R*(T^(3/2))))*log(V_1/(V_1+b))));//[Pa]
f_1 = f_1*10^(-5);//[bar]

printf(" The fugacity of water vapour at 320 C and
    70 bar pressure is %f bar\n\n",f_1);

// At 320 C and 170 bar pressure, we have
P_2 = 170;//[bar]
P_2 = P_2*10^(5);//[Pa]

// Solving the cubic equation
deff('[y]=f2(V)','y=V^(3)-((R*T)/P_2)*V^(2)-((b^(2))
    +((b*R*T)/P_2)-(a/(T^(1/2)*P_2)))*V-(a*b)/(T
    ^(1/2)*P_2)');
V4 = fsolve(1,f2);
V5 = fsolve(10,f2);
V6 = fsolve(100,f2);
// The above equation has only 1 real root,other two
    roots are imaginary. Therefore,
V_2 = V6;//[m^(3)/mol]
// Thus compressibility factor is
Z_2 = (P_2*V_2)/(R*T);

// For Redlich-Kwong equation of state
// log(f/P) = Z - 1 - log(V_1/(V_1-b)) + (a/(b*R*(T
    ^(3/2))))*log(V/(V+b))
f_2 = P_2*(exp(Z_2-1-log(Z_2)+log(V_2/(V_2-b))+(a/(b
    *R*(T^(3/2))))*log(V_2/(V_2+b))));//[Pa]
f_2 = f_2*10^(-5);//[bar]

printf(" The fugacity of water vapour at 320 C and
    170 bar pressure is %f bar\n\n",f_2);
```

Scilab code Exa 10.24 Determination of change in internal energy

```
clear;
clc;

//Example - 10.24
//Page number - 367
printf("Example - 10.24 and Page number - 367\n\n");

//Given
Vol = 0.057;//[m^(3)] - Volume of car tyre
P_1 = 300;//[kPa] - Initial pressure
P_1 = P_1*10^(3);//[Pa]
T_1 = 300;//[K] - Initial temperature
P_2 = 330;//[kPa] - Finnal pressure
P_2 = P_2*10^(3);//[Pa]
R = 8.314;//[J/mol*K] - Universal gas constant
Cv_0 = 21;//[J/mol-K] - Heat capacity for air

// For oxygen
Tc_O2 = 154.6;//[K] - Critical temperature
Pc_O2 = 50.43;//[bar] - Critical pressure
Pc_O2 = Pc_O2*10^(5);//[Pa]
y1 = 0.21;// - Mole fraction of oxygen
// For nitrogen
Tc_N2 = 126.2;//[K] - Critical temperature
Pc_N2 = 34.00;//[bar] - Critical pressure
Pc_N2 = Pc_N2*10^(5);//[Pa]
y2 = 0.79;// - Mole fraction of nitrogen

// (1)
// Assuming ideal gas behaviour. The volume remains
   the same,therefore,we get
// P_1/T_1 = P_2/T_2
T_2 = P_2*(T_1/P_1);//[K]

n = (P_1*Vol)/(R*T_1);//[mol] - Number of moles
delta_U = n*Cv_0*(T_2-T_1);//[J]
```

```

printf(" (1).The change in internal energy (for
   ideal gas behaviour) is %f J\n\n",delta_U);

//(2)
//  For van der Walls equation of state
a_O2 = (27*R^(2)*Tc_O2^(2))/(64*Pc_O2);//[Pa-m^(6)/
   mol^(2)]
a_N2 = (27*R^(2)*Tc_N2^(2))/(64*Pc_N2);//[Pa-m^(6)/
   mol^(2)]
a_12 = (a_O2*a_N2)^(1/2);
b_O2 = (R*Tc_O2)/(8*Pc_O2);//[m^(3)/mol]
b_N2 = (R*Tc_N2)/(8*Pc_N2);//[m^(3)/mol]

// For the mixture
a = y1^(2)*a_O2 + y2^(2)*a_N2 + 2*y1*y2*a_12;//[Pa-m
   ^(6)/mol^(2)]
b = y1*b_O2 + y2*b_N2;//[m^(3)/mol]

// From the cubic form of van der Walls equation of
   state
// At 300 K and 300 kPa,
deff('[y]=f1(V)','y=V^(3)-(b+(R*T_1)/P_1)*V^(2)+(a/
   P_1)*V-(a*b)/P_1');
V_1 = fsolve(0.1,f1);
V_2 = fsolve(10,f1);
V_3 = fsolve(100,f1);
// The above equation has only 1 real root, other
   two roots are imaginary
V = V_1;//[m^(3)/mol]

// Now at P = 330 kPa and at molar volume V
// The van der Walls equation of state is
// (P + a/V^(2))*(V - b) = R*T
T_2_prime = ((P_2 + a/V^(2))*(V - b))/R;//[K]
n_prime = Vol/V;//[mol]

// Total change in internal energy is given by
// delta_U_prime = n_prime*delta_U_ig + n_prime*(
   U_R_2 - U_R_1)
// delta_U_prime = n_prime*Cv_0*(T_2_prime - T_1) +
   n_prime*a(1/V_2 - 1/V_1);
// Since V_1 = V_2 = V, therefore
delta_U_prime = n_prime*Cv_0*(T_2_prime - T_1);

printf(" (2).The change in internal energy (for van
   der Walls equation of state) is %f J\n\n",
   delta_U_prime);
```

Scilab code Exa 10.25 Calculation of enthalpy and entropy change

```
clear;
clc;
funcprot(0);

//Example - 10.25
//Page number - 369
printf("Example - 10.25 and Page number - 369\n\n");

//Given
T_1 = 150 + 273.15;//[K] - Initial emperature
T_2 = T_1;// Isothermal process
P_1 = 100*10^(3);//[Pa] - Initial pressure
P_2 = 450*10^(3);//[Pa] - Final pressure
R = 8.314;//[J/mol*K] - Universal gas constant
// For water
Tc = 647.1;//[K] - Critical temperature
Pc = 220.55;//[bar] - Critical pressure
Pc = Pc*10^(5);
w = 0.345;
Mol_wt = 18.015;//[g/mol] - Molecular weight of
   water
Cp_0 = 4.18;//[J/mol-K] - Standard heat capacity of
```

```
    water

// Both phases are superheated vapour phases because
     at 150 C the vapour pressure of steam is 4.67
    bar and both operating pressures are below
    saturated pressure.
// In Peng-Robinson equation of state
m = 0.37464 + 1.54226*w - 0.26992*w^(2);
// At T_1 and P_1, we have
Tr = T_1/Tc;
alpha = (1 + m*(1 - Tr^(1/2)))^(2);
a = ((0.45724*(R*Tc)^(2))/Pc)*alpha;//[Pa*m^(6)/mol
    ^(2)]
b = (0.07780*R*Tc)/Pc;//[m^(3)/mol]

// Cubuc form of Peng-Robinson equation of stste is
    given by
// V^(3)+(b-(R*T)/P)*V^(2)-((3*b^(2))+((2*R*T*b)/P)
    -(a/P))*V+b^(3)+((R*T*(b^(2))/P)-((a*b)/P)=0;
// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)+(b-(R*T_1)/P_1)*V^(2)-((3*b
    ^(2))+((2*R*T_1*b)/P_1)-(a/P_1))*V+b^(3)+((R*T_1
    *(b^(2)))/P_1)-((a*b)/P_1)');
V1 = fsolve(-1,f);
V2 = fsolve(0,f);
V3 = fsolve(1,f);
//The largest root is for vapour phase,
//The largest root is only considered as the
    systemis gas
V_1 = V3;//[m^(3)/mol]
// Thus compressibility factor is
Z_1 = (P_1*V_1)/(R*T_1);

// At T_2 and P_2, we have
// Cubuc form of Peng-Robinson equation of stste is
    given by
// V^(3)+(b-(R*T)/P)*V^(2)-((3*b^(2))+((2*R*T*b)/P)
    -(a/P))*V+b^(3)+((R*T*(b^(2))/P)-((a*b)/P)=0;
```

```
// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)+(b-(R*T_2)/P_2)*V^(2)-((3*b
    ^(2))+((2*R*T_2*b)/P_2)-(a/P_2))*V+b^(3)+((R*T_2
    *(b^(2)))/P_2)-((a*b)/P_2)');
V4 = fsolve(-1,f);
V5 = fsolve(0,f);
V6 = fsolve(1,f);
//The largest root is for vapour phase,
//The largest root is only considered as the
    systemis gas
V_2 = V6;//[m^(3)/mol]
// Thus compressibility factor is
Z_2 = (P_2*V_2)/(R*T_2);

// In the Peng-Robinson equation of stste
// da/dT = -(a*m)/((alpha*T*Tc)^(1/2))
// The residual enthalpy is given by
// H_R = R*T*(Z-1) + (((T*(da_dT))-a)/(2*2^(1/2)*b))
    *log((Z+(1+2^(1/2)*((P*b)/(R*T))))/(Z+(1-2^(1/2)
    *((P*b)/(R*T)))))

// At state 1
da_dT_1 = -(a*m)/((alpha*T_1*Tc)^(1/2));//[Pa*m^(6)/
    mol^(2)]
H_R_1 = R*T_1*(Z_1-1) + (((T_1*(da_dT_1))-a)
    /(2*(2^(1/2))*b))*log((Z_1+(1+2^(1/2))*((P_1*b)/(
    R*T_1)))/(Z_1+(1-2^(1/2))*((P_1*b)/(R*T_1))));

// At state 2
da_dT_2 = -(a*m)/((alpha*T_2*Tc)^(1/2));//[Pa*m^(6)/
    mol^(2)]
H_R_2 = R*T_2*(Z_2-1) + (((T_2*(da_dT_2))-a)
    /(2*2^(1/2)*b))*log((Z_2+(1+2^(1/2))*((P_2*b)/(R*
    T_1)))/(Z_2+(1-2^(1/2))*((P_2*b)/(R*T_1))));


// Since the temperature is the same,therefore ideal
     gas change in enthalpy is zero and thus
```

```
delta_H = H_R_2 - H_R_1;;//[J/mol]
delta_H = delta_H/Mol_wt;//[kJ/kg]

// The residual entropy relation for a substance
   following Peng - Robinson equation of state ia
// S_R = R*log(Z - (P*b)/(R*T)) + (da_dT/(2*2^(1/2)*
   b))*log((Z+(1+2^(1/2))*((P*b)/(R*T)))/(Z
   +(1-2^(1/2))*((P*b)/(R*T))))

// The residual entropy at state 1 is
S_R_1 = R*log(Z_1 - (P_1*b)/(R*T_1)) + (da_dT_1
   /(2*2^(1/2)*b))*log((Z_1+(1+2^(1/2))*((P_1*b)/(R*
   T_1)))/(Z_1+(1-2^(1/2))*((P_1*b)/(R*T_1))));

// The residual entropy at state 2 is
S_R_2 = R*log(Z_2 - (P_2*b)/(R*T_2)) + (da_dT_2
   /(2*2^(1/2)*b))*log((Z_2+(1+2^(1/2))*((P_2*b)/(R*
   T_2)))/(Z_2+(1-2^(1/2))*((P_2*b)/(R*T_2))));

delta_S_R = S_R_2 - S_R_1;//[J/mol-K]

// The ideal gas change in entropy is
delta_S_ig = Cp_0*log(T_2/T_1) - R*log(P_2/P_1);//[J
   /mol-K]

// Therefore
delta_S = delta_S_R + delta_S_ig;//[J/mol-K]

printf(" The enthalpy change is given by, delta_H =
   %f kJ/mol\n\n",delta_H);
printf(" The entropy change is given by, delta_S =
   %f J/mol-K\n\n",delta_S);
```

Scilab code Exa 10.26 Calculation of final temperature and pressure

```
clear;
clc;
funcprot(0);

//Example - 10.26
//Page number - 370
printf("Example - 10.26 and Page number - 370\n\n");

//Given
Vol = 0.15;//[m^(3)]
T_1 = 170;//[K] - Initial emperature
P_1 = 100;//[bar] - Initial pressure
P_1 = P_1*10^(5);//[Pa]
R = 8.314;//[J/mol*K] - Universal gas constant
// For nitrogen
Tc = 126.2;//[K] - Critical tempeature
Pc = 34;//[bar] - Critical pressure
Pc = Pc*10^(5);//[Pa]
w = 0.038;
// Cp_0 = 27.2+4.2*10^(-3)*T

//(1)
// For van der Walls equation of state
a = (27*R^(2)*Tc^(2))/(64*Pc);//[Pa-m^(6)/mol^(2)]
b = (R*Tc)/(8*Pc);//[m^(3)/mol]

// The cubic form of van der Walls equation of state
    is given by,
// V^(3) - (b + (R*T)/P)*V^(2) + (a/P)*V - (a*b)/P =
    0
// On simplification the equation changes to
// V^(3) - 1.799*10^(4)*V^(2) + 1.366*10^(-8)*V -
   5.269*10^(-13) = 0

// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)-1.799*10^(-4)*V^(2) +
   1.366*10^(-8)*V - 5.269*10^(-13)');
V1 = fsolve(1,f);
```

```
V2 = fsolve(10,f);
V3 = fsolve(100,f);
// The above equation has only 1 real root, other
   two roots are imagimnary
V_1 = V1;//[m^(3)/mol]
// Thus total number of moles is given by
n_1 = Vol/V_1;//[mol]

// After 500 mol are withdrawn, the final number of
   moles is given by
n_2 = n_1 - 500;//[mol]
// Thus molar volume at final state is
V_2 = Vol/n_2;//[m^(3)/mol]

// The ideal entropy change is guven by
// delta_S_ig = integrate('27.2+4.2*10^(-3)*T','T',
   T_1,T_2) - R*log(P_2/P_1);
// The residual entropy change is given by
// delta_S_R = R*log((P_2*(V_2-b))/(R*T_2)) - R*log
   ((P_1*(V_1-b))/(R*T_1))
// delta_S = delta_S_ig = delta_S_R
// delta_S = integrate('27.2+4.2*10^(-3)*T','T',T_1,
   T_2) + R*log((V_2-b)/(V_1-b));
// During discharging delta_S = 0, thus on
   simplification we get
// 18.886*log(T_2) + 4.2*10^(-3)*T_2 - 92.937 = 0
// Solving the above equation we get
deff('[y]=f1(T_2)','y=18.886*log(T_2) + 4.2*10^(-3)*
   T_2 - 92.937');
T_2 = fsolve(1,f1);

// Thus at T_2,
P_2 = (R*T_2)/(V_2-b) - a/V_2^(2);//[N/m^(2)]
P_2 = P_2*10^(-5);//[bar]

printf(" (1).The final temperature is %f K\n",T_2);
printf("     The final pressure is %f bar\n\n",P_2);

```

```
//(2)
// In Peng-Robinson equation of state
m = 0.37464 + 1.54226*w - 0.26992*w^(2);
// At T_1 and P_1, we have
Tr = T_1/Tc;
alpha = (1 + m*(1 - Tr^(1/2)))^(2);
a_2 = ((0.45724*(R*Tc)^(2))/Pc)*alpha;//[Pa*m^(6)/
    mol^(2)]
b_2 = (0.07780*R*Tc)/Pc;//[m^(3)/mol]

// Cubuc form of Peng-Robinson equation of stste is
    given by
// V^(3)+(b-(R*T)/P)*V^(2)-((3*b^(2))+((2*R*T*b)/P)
    -(a/P))*V+b^(3)+((R*T*(b^(2))/P)-((a*b)/P)=0;
// Solving the cubic equation
deff('[y]=f2(V)','y=V^(3)+(b_2-(R*T_1)/P_1)*V^(2)
    -((3*b_2^(2))+((2*R*T_1*b_2)/P_1)-(a_2/P_1))*V+
    b_2^(3)+((R*T_1*(b_2^(2)))/P_1)-((a_2*b_2)/P_1)')
    ;
V4 = fsolve(-1,f2);
V5 = fsolve(0,f2);
V6 = fsolve(0.006,f2);
//The above equation has only 1 real root,the other
    two roots are imaginary
V_1_2 = V6;//[m^(3)/mol]

// The number of moles in the initial state is given
     by
n_1_2 = Vol/V_1_2;//[mol]
// After 500 mol are withdrawn, the final number of
    moles is given by
n_2_2 = n_1_2 - 500;//[mol]
// Thus molar volume at final state is
V_2_2 = Vol/n_2_2;//[m^(3)/mol]

// At the final state the relation between pressure
    and temperature is
// P_2_2 = (R*T_2_2)/(V_2_2-b_2) - a_2/V_2_2^(2)
// P_2_2 = 7.23*10^(4)*T_2 - 3.93*10^(7)*a_2

// Now let us calculate the residual entropy at
   initial state
Z_1 = (P_1*V_1_2)/(R*T_1);
da_dT_1 = -(a*m)/((alpha*T_1*Tc)^(1/2));//[Pa*m^(6)/
   mol^(2)] - da/dT

// The residual entropy change for Peng-Robinson
   equatiob of state is given by
// S_R = R*log(Z-(P*b)/(R*T)) + (da_dT/(2*2^(1/2)*b)
   )*log((V+(1+2^(1/2))*b))/((V+(1-2^(1/2)*b))));
S_R_1 = R*(log(Z_1-(P_1*b_2)/(R*T_1))) + (da_dT_1
   /(2*2^(1/2)*b_2))*(log((V_1_2+(1+2^(1/2))*b_2)/(
   V_1_2+(1-2^(1/2))*b_2)));

// The total entropy change is given by
// delta_S = delta_S_ig + delta_S_R
// where, delta_S_ig = integrate('27.2+4.2*10^(-3)*T
   ','T',T_1,T_2_2) - R*log(P_2_2/P_1);
// and, P_2_2 = (R*T_2_2)/(V_2_2-b_2) - a_2/V_2_2
   ^(2)
// On simplification we get
// delta_S = 27.2*log(T_2_2-T_1) + 4.2*10^(-3)*(
   T_2_2-T_1) - R*log(P_2_2/P_1) + R*log(Z_2-(P_2_2*
   b)/(R*T_2_2)) + 6226*(da_dT_2) + 9.22

// Now we have the determine the value of T_2_2 such
    that delta_S = 0
// Starting with a temperature of 150 K
T_prime = 100;//[K]
error = 10;
while(error>0.1)
    Tr_prime = T_prime/Tc;
    alpha_prime = (1 + m*(1 - Tr_prime^(1/2)))^(2);
    a_prime = ((0.45724*(R*Tc)^(2))/Pc)*alpha_prime;
    P_prime = 7.23*10^(4)*T_prime - 3.93*10^(7)*
       a_prime;
    Z_prime = (P_prime*V_2_2)/(R*T_prime);
    da_dT_prime = -(a_prime*m)/((alpha_prime*T_prime
       *Tc)^(1/2));
    delta_S = 27.2*log(T_prime/T_1) + 4.2*10^(-3)*(
       T_prime-T_1) - R*log(P_prime/P_1) + R*log(
       Z_prime-((P_prime*b_2)/(R*T_prime))) + 6226*(
       da_dT_prime) + 9.22;
    error=abs(delta_S);
    T_prime = T_prime + 0.3;
end

T_2_2 = T_prime;//[K] - Final temperature
P_2_2 = P_prime*10^(-5);//[bar] - Final pressure

printf(" (2).The final temperature is %f K\n",T_2_2)
   ;
printf("     The final pressure is %f bar\n",P_2_2);
```

Scilab code Exa 10.27 Calculation of vapour pressure

```
clear;
clc;
funcprot(0);

//Example - 10.27
//Page number - 374
printf("Example - 10.27 and Page number - 374\n\n");

//Given
T = 373.15;//[K]
Tc = 562.16;//[K]
Pc = 48.98;//[bar]
Pc = Pc*10^(5);//[Pa]
R = 8.314;//[J/mol-K] - Universal gas constant

```

```
// The cubic form of Redlich Kwong equation of state
    is given by,
// V^(3) - ((R*T)/P)*V^(2) - ((b_1^(2)) + ((b_1*R*T)
   /P) - (a/(T^(1/2)*P))*V - (a*b)/(T^(1/2)*P) = 0

a = (0.42748*(R^(2))*(Tc^(2.5)))/Pc;//[Pa*m^(6)*K
   ^(1/2)/mol]
b = (0.08664*R*Tc)/Pc;//[m^(3)/mol]

// At 373.15 K, let us assume the pressure to be 2.5
    bar and under these conditions
P_1 = 2.5;//[bar]
P_1 = P_1*10^(5);//[bar]

// Putting the values in Redlich Kwong equation of
   state, the equation becomes
// V^(3) - 0.0124*V^(2) + 8.326*10^(-6)*V -
   7.74*10^(-10) = 0
// Solving the cubic equation

deff('[y]=f(V)','y=V^(3) - 0.0124*V^(2) +
   8.326*10^(-6)*V - 7.74*10^(-10)');
V1=fsolve(-9,f);
V2=fsolve(10,f);
V3=fsolve(0.1,f);
// The largest root and the smallest root is
   considered for liquid phase and vapour phase
   respectively.
V_liq = V1;//[m^(3)/mol] - Molar volume in liquid
   phase
V_vap = V3;//[m^(3)/mol] - Molar volume in vapour
   phase

// Let us calculate the fugacity of vapour phase
// log(f_vap/P) = b/(V-b) + log((R*T)/(P*(V-b))) - (
   a/(R*T^(1.5)))*(1/(V+b) - (1/b)*log(V/(V+b)))
f_vap = P_1*exp(b/(V_vap-b) + log((R*T)/(P_1*(V_vap-
   b))) - (a/(R*T^(1.5)))*(1/(V_vap+b) - (1/b)*log(
    V_vap/(V_vap+b))));//[Pa]

// Let us calculate the fugacity of the liquid phase
f_liq = P_1*exp(b/(V_liq-b) + log((R*T)/(P_1*(V_liq-
    b))) - (a/(R*T^(1.5)))*(1/(V_liq+b) - (1/b)*log(
    V_liq/(V_liq+b))));


// The two fugacities are not same; therefore
    another pressure is to be assumed. The new
    pressure is
P_new = P_1*(f_liq/f_vap);//[Pa]

// At P_new
deff('[y]=f1(V)','y=V^(3) - ((R*T)/P_new)*V^(2) - (b
    ^(2) + ((b*R*T)/P_new) - a/(T^(1/2)*P_new))*V - (
    a*b)/(T^(1/2)*P_new)');
V4=fsolve(-9,f1);
V5=fsolve(10,f1);
V6=fsolve(0.1,f1);
// The largest root and the smallest root is
    considered for liquid phase and vapour phase
    respectively.
V_liq_2 = V4;//[m^(3)/mol] - Molar volume in liquid
    phase
V_vap_2 = V6;//[m^(3)/mol] - Molar volume in vapour
    phase

f_vap_prime = P_new*exp(b/(V_vap_2-b) + log((R*T)/(
    P_new*(V_vap_2-b))) - (a/(R*T^(1.5)))*(1/(V_vap_2
    +b) - (1/b)*log(V_vap_2/(V_vap_2+b))));//[Pa]
f_liq_prime = P_new*exp(b/(V_liq_2-b) + log((R*T)/(
    P_new*(V_liq_2-b))) - (a/(R*T^(1.5)))*(1/(V_liq_2
    +b) - (1/b)*log(V_liq_2/(V_liq_2+b))));

// Since the fugacities of liquid and vapour
    phasesare almost same the assumed pressure may be
     taken as vapour pressure at 373.15 K
```

```
P_new = P_new*10^(-5);//[bar]

printf(" The vapour pressure of benzene using
   Redlich Kwong equation of state is %f bar\n",
   P_new);
```

Scilab code Exa 10.28 Determination of vapour pressure

```
clear;
clc;
funcprot(0);

//Example - 10.28
//Page number - 374
printf("Example - 10.28 and Page number - 375\n\n");

//Given
T = 150 + 273.15;//[K]
Tc = 647.1;//[K]
Pc = 220.55;//[bar]
Pc = Pc*10^(5);//[Pa]
w = 0.345;
R = 8.314;//[J/mol-K] - Universal gas constant

// Let us assume a pressure of 100 kPa.
P_1 = 100*10^(3);//[Pa]

// At 100 kPa and 423.15 K, from Peng-Robinson
   equation of stste
m = 0.37464 + 1.54226*w - 0.26992*w^(2);
Tr = T/Tc;
alpha = (1 + m*(1 - Tr^(1/2)))^(2);
a = ((0.45724*(R*Tc)^(2))/Pc)*alpha;//[Pa*m^(6)/mol
   ^(2)]
b = (0.07780*R*Tc)/Pc;//[m^(3)/mol]
```

```
// Cubic form of Peng-Robinson equation of ststе is
   given by
// V^(3)+(b-(R*T)/P)*V^(2)-((3*b^(2))+((2*R*T*b)/P)
   -(a/P))*V+b^(3)+((R*T*(b^(2))/P)-((a*b)/P)=0;
// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)+(b-(R*T)/P_1)*V^(2)-((3*b
   ^(2))+((2*R*T*b)/P_1)-(a/P_1))*V+b^(3)+((R*T*(b
   ^(2)))/P_1)-((a*b)/P_1)');
V1 = fsolve(-1,f);
V2 = fsolve(0,f);
V3 = fsolve(1,f);
// The largest root and the smallest root is
   considered for liquid phase and vapour phase
   respectively.
V_liq = V1;//[m^(3)/mol] - Molar volume in liquid
   phase
V_vap = V3;//[m^(3)/mol] - Molar volume in vapour
   phase

// The compressibility factor is given by
Z_vap = (P_1*V_vap)/(R*T);// For liquid phase
Z_liq = (P_1*V_liq)/(R*T);// For vapour phase

// The expression for fugacity of Peng Robinson
   equation is
// log(f/P) = (Z-1) - log(Z-((P*b)/(R*T))) - (a
   /(2*2^(1/2)*b*R*T))*log((Z+(1+2^(1/2))*((P*b)/(R*
   T)))/((Z+(1-2^(1/2))*((P*b)/(R*T)))
// For vapour phase
f_P_vap = exp((Z_vap-1) - log(Z_vap-((P_1*b)/(R*T)))
    - (a/(2*2^(1/2)*b*R*T))*log((Z_vap+(1+2^(1/2))
   *((P_1*b)/(R*T)))/(Z_vap+(1-2^(1/2))*((P_1*b)/(R*
   T)))));
// For liquid phase
f_P_liq = exp((Z_liq-1) - log(Z_liq-((P_1*b)/(R*T)))
    - (a/(2*2^(1/2)*b*R*T))*log((Z_liq+(1+2^(1/2))
   *((P_1*b)/(R*T)))/(Z_liq+(1-2^(1/2))*((P_1*b)/(R*
   T)))));
```

```

// Therefore f_liq/f_vap can be calculated as
fL_fV = (f_P_liq/f_P_vap);

// The two values (f/P)_vap and (f/P)_vap are not
   same [ (f_P_liq/f_P_vap) >1 ]; therefore another
   pressure is to be assumed. The new pressure be
P_new = P_1*(f_P_liq/f_P_vap);//[Pa]

// At P_new and 423.15 K, from Peng-Robinson
   equation of stste

// V^(3)+(b-(R*T)/P)*V^(2)-((3*b^(2))+((2*R*T*b)/P)
   -(a/P))*V+b^(3)+((R*T*(b^(2))/P)-((a*b)/P)=0;
// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)+(b-(R*T)/P_new)*V^(2)-((3*b
   ^(2))+((2*R*T*b)/P_new)-(a/P_new))*V+b^(3)+((R*T
   *(b^(2)))/P_new)-((a*b)/P_new)');
V4 = fsolve(-1,f);
V5 = fsolve(0,f);
V6 = fsolve(1,f);
// The largest root and the smallest root is
   considered for liquid phase and vapour phase
   respectively.
V_liq_2 = V4;//[m^(3)/mol] - Molar volume in liquid
   phase
V_vap_2 = V6;//[m^(3)/mol] - Molar volume in vapour
   phase

// The compressibility factor is given by
Z_vap_2 = (P_new*V_vap_2)/(R*T);// For liquid phase
Z_liq_2 = (P_new*V_liq_2)/(R*T);// For vapour phase

// For vapour phase
f_P_vap_2 = exp((Z_vap_2-1) - log(Z_vap_2-((P_new*b)
   /(R*T))) - (a/(2*2^(1/2)*b*R*T))*log((Z_vap_2
   +(1+2^(1/2))*((P_new*b)/(R*T)))/(Z_vap_2
   +(1-2^(1/2))*((P_new*b)/(R*T)))));
```

```
// For liquid phase
f_P_liq_2 = exp((Z_liq_2-1) - log(Z_liq_2-((P_new*b)
   /(R*T))) - (a/(2*2^(1/2)*b*R*T))*log((Z_liq_2
   +(1+2^(1/2))*((P_new*b)/(R*T)))/(Z_liq_2
   +(1-2^(1/2))*((P_new*b)/(R*T)))));

// Therefore f_liq/f_vap can be calculated as
fL_fV_2 = (f_P_liq_2/f_P_vap_2);

// And new pressure is given by
P_new_prime = P_new*(f_P_liq_2/f_P_vap_2);//[Pa]
P_new_prime = P_new_prime*10^(-5);

// Since the change in pressure is small, so we can
   take this to be the vapour pressure at 150 C

printf(" The vapour pressure of water using Peng-
   Robinson equation of stste  is %f bar\n",
   P_new_prime);
```

Chapter 11

Properties of a Component in a Mixture

Scilab code Exa 11.1 Determination of volumes of ethanol and water

```
clear;
clc;

//Example - 11.1
//Page number - 385
printf("Example - 11.1 and Page number - 385\n\n");

//Given
Vol_total = 3;//[m^(3)] - Total volume of solution
x_ethanol = 0.6;//Mole fraction of ethanol
x_water = 0.4;//Mole fraction of water

//The partial molar volumes of the components in the
    mixture are
V_ethanol_bar = 57.5*10^(-6);//[m^(3)/mol]
V_water_bar = 16*10^(-6);//[m^(3)/mol]

//The molar volumes of the pure components are
V_ethanol = 57.9*10^(-6);//[m^(3)/mol]
```

```
V_water = 18*10^(-6);//[m^(3)/mol]

//The molar volume of the solution is
V_sol = x_ethanol*V_ethanol_bar + x_water*
   V_water_bar;//[m^(3)/mol]
//Total number of moles can be calculated as
n_total = Vol_total/V_sol;//[mol]

//Moles of the components are
n_ethanol = n_total*x_ethanol;//[mol]
n_water = n_total*x_water;//[mol]

//Finally the volume of the pure components required
    can be calculated as
Vol_ethanol = V_ethanol*n_ethanol;
Vol_water = V_water*n_water;

printf("Required volume of ethanol is %f cubic metre
   \n\n",Vol_ethanol);
printf("Required volume of water is %f cubic metre",
   Vol_water);
```

Scilab code Exa 11.2 Developing an expression

```
clear;
clc;

//Example - 11.2
//Page number - 385
printf("Example - 11.2 and Page number - 385\n\n");

//Given
T = 25+273.15;//[K] - Temperature
P = 1;//[atm]
//Component 1 = water
```

```
//component 2 = methanol
a = -3.2;//[cm^(3)/mol] - A constant
V2 = 40.7;//[cm^(3)/mol] - Molar volume of pure
    component 2 (methanol)
//V1_bar = 18.1 + a*x_2^(2)

//From Gibbs-Duhem equation at constant temperature
    and pressure we have
//x_1*dV1_bar + x_2*dV2_bar = 0
//dV2_bar = -(x_1/x_2)*dV1_bar = -(x_1/x_2)*a*2*x_2*
    dx_2 = -2*a*x_1*dx_2 = 2*a*x_1*dx_1

//At x_1 = 0: x_2 = 1 and thus V2_bar = V2
//Integrating the above equation from x_1 = 0 to x_1
     in the RHS, and from V2_bar = V2 to V2 in the
    LHS, we get
//V2_bar = V2 + a*x_1^(2) -  Molar volume of
    component 2(methanol) in the mixture

printf("The expression for the partial molar volume
    of methanol(2) is\nV2_bar = V2 + a*x_1^(2) [cm
    ^(3)/mol]\n\n");

//At infinite dilution , x_2 approach 0 and thus x_1
    approach 1, therefore
x_1 = 1;// Mole fraction of component 1(water) at
    infinite dilution
V2_bar_infinite = V2 + a*(x_1^(2));//[cm^(3)/mol]

printf("The partial molar volume of methanol at
    infinite dilution is %f cm^(3)/mol",
    V2_bar_infinite);
```

Scilab code Exa 11.3 Determination of partial molar volume

```
clear;
clc;

//Example - 11.3
//Page number - 386
printf("Example - 11.3 and Page number - 386\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 11.3 on page
   number 386 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 11.3 on
   page number 386 of the book.")
```

Scilab code Exa 11.4 Calculation of enthalpies

```
clear;
clc;

//Example - 11.4
//Page number - 387
printf("Example - 11.4 and Page number - 387\n\n");

//Given
//H = a*x_1 + b*x_2 +c*x_1*x_2

//The values of the constants are
a = 15000;//[J/mol]
b = 20000;//[J/mol]
c = -2000;//[J/mol]

//(1)
```

```
//Enthalpy of pure component 1 = H1 is obtained at
   x_2 = 0, thus
x_2 = 0;
x_1 = 1;
H1 = a*x_1 + b*x_2 +c*x_1*x_2;//[J/mol]
printf("(a).The enthalpy of pure component 1 is %f J
   /mol\n",H1);

//Similarly for component 2,
//Enthalpy of pure component 2 = H2 is obtained at
   x_1 = 0, thus
x_1_prime = 0;
x_2_prime = 1;
H2 = a*x_1_prime + b*x_2_prime +c*x_1_prime*
   x_2_prime;//[J/mol]
printf("     The enthalpy of pure component 2 is %f J
   /mol\n\n",H2);

//(b)
//This part involves proving a relation in which no
   mathematics and no calculations are involved.
//For prove refer to this example 11.4 on page
   number 387 of the book.

//(c)
//From part (b), we have the relation
//H1_bar = a + c*(x_2^(2))
//H2_bar = b + c*(x_1^(2))

//For enthalpy of component 1 at infinite dilution,
   x_1 approach 0 and thus x_2 approach 1, therefore
x_1_c = 0;
x_2_c = 1;
H1_infinite = a + c*(x_2_c^(2));//[cm^(3)/mol]
printf("(C).The enthalpy of componenet 1 at infinite
    dilution (at x_1 = 0) is %f J/mol\n",H1_infinite
   );

```

```
//At x_1 = 0.2
x_1_c1 = 0.2;
x_2_c1 = 0.8;
H1_bar_c1 = a + c*(x_2_c1^(2));//[J/mol]
printf("    The enthalpy of componenet 1 at (at x_1
    = 0.2) is %f J/mol\n",H1_bar_c1);

//At x_1 = 0.8
x_1_c2 = 0.8;
x_2_c2 = 0.2;
H1_bar_c2 = a + c*(x_2_c2^(2));//[J/mol]
printf("    The enthalpy of componenet 1 at (at x_1
    = 0.8) is %f J/mol",H1_bar_c2);

//As x_1 increases , 'H1_bar' approaches the value of
     'H1'

//(d)
//This part involves proving a relation in which no
    mathematics and no calculations are involved.
//For prove refer to this example 11.4 on page
    number 387 of the book.
```

Scilab code Exa 11.5 Developing an expression and calculation for enthalpy change of mixture

```
clear;
clc;

//Example - 11.5
//Page number - 389
printf("Example - 11.5 and Page number - 389\n\n");

//This problem involves proving a relation in which
    no mathematical components are involved.
```

```
//For prove refer to this example 11.5 on page
   number 389 of the book.
printf(" This problem involves proving a relation in
    which no mathematical components are involved.\n
   \n");
printf(" For prove refer to this example 11.5 on
   page number 389 of the book.")
```

Scilab code Exa 11.6 Proving a mathematical relation

```
clear;
clc;

//Example - 11.6
//Page number - 390
printf("Example - 11.6 and Page number - 390\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 11.6 on page
   number 390 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 11.6 on
   page number 390 of the book.")
```

Scilab code Exa 11.7 Proving a mathematical relation

```
clear;
clc;

//Example - 11.7
```

```
//Page number - 393
printf("Example - 11.7 and Page number - 393\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 11.7 on page
   number 393 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 11.7 on
   page number 393 of the book.")
```

Scilab code Exa 11.8 Proving a mathematical relation

```
clear;
clc;

//Example - 11.8
//Page number - 394
printf("Example - 11.8 and Page number - 394\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 11.8 on page
   number 394 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 11.8 on
   page number 394 of the book.")
```

Scilab code Exa 11.9 Calculation of minimum work required

```
clear;
clc;

//Example - 11.9
//Page number - 395
printf("Example - 11.9 and Page number - 395\n\n");

//Given
n = 1*10^(3);//[mol] - No of moles
P = 0.1;//[MPa] - Pressure of the surrounding
T = 300;//[K] - Temperature of the surrounding
x_1 = 0.79;//Mole fraction of N2 in the air
x_2 = 0.21;//Mole fraction of O2 in the air
R=8.314;//[J/mol*K]

//Change in availability when x_1 moles of component
    1 goes from pure state to that in the mixture is
//x_1*(si_1 - si_2) = x_1*[H1 - H1_bar - T_0*(S1 -
   S1_bar)]
//Similarly change in availability of x_2 moles of
   component 2 is
//x_2*(si_1 - si_2) = x_2*[H2 - H2_bar - T_0*(S2 -
   S2_bar)]

//and thus total availability change when 1 mol of
   mixture is formed from x_1 mol of component 1 and
    x_2 mol of component 2 is equal to reversible
   work
//W_rev = x_1*[H1 - H1_bar - T_0*(S1 - S1_bar)] +
   x_2*[H2 - H2_bar - T_0*(S2 - S2_bar)]
//W_rev = -[x_1*(H1_bar - H1) + x_2*(H2_bar - H2)] +
    T_0*[x_1*(S1_bar - S1) + x_2*(S2_bar - S2)]
//W_rev =  -[delta_H_mix] +T_0*[delta_S_mix]

//If T = T_0 that is,temperature of mixing is same
   as that of surroundings, W_rev = -delta_G_mix.
//W_rev = -delta_G_mix = R*T*(x_1*log(x_1) + x_2*log
   (x_2))
```

```
W_rev = R*T*(x_1*log(x_1) + x_2*log(x_2));//[J/mol]

//Therefore total work transfer is given by
W_min = (n*W_rev)/1000;//[kJ]

printf("The minimum work required is %f kJ",W_min);
```

Scilab code Exa 11.10 Calculation of fugacity 0f the mixture

```
clear;
clc;

//Example - 11.10
//Page number - 400
printf("Example - 11.10 and Page number - 400\n\n");

//Given
x_A = 0.20;// Mole fraction of A
x_B = 0.35;// Mole fraction of B
x_C = 0.45;// Mole fraction of C

phi_A = 0.7;// Fugacity coefficient of A
phi_B = 0.6;// Fugacity coefficient of B
phi_C = 0.9;// Fugacity coefficient of C

P = 6.08;//[MPa] - Pressure
T = 384;//[K] - Temperature

//We know that
//log(phi) = x_1*log(phi_) + x_2*log(phi_2) + x_3*
   log(phi_3)
log_phi = x_A*log(phi_A) + x_B*log(phi_B) + x_C*log(
   phi_C);// Fugacity coefficient
phi = exp(log_phi);

```

```
//Thus fugacity is given by,
f_mixture = phi*P;//[MPa]

printf("The fugacity of the mixture is %f MPa",
   f_mixture);
```

Scilab code Exa 11.11 Proving a mathematical relation

```
clear;
clc;

//Example - 11.11
//Page number - 400
printf("Example - 11.11 and Page number - 400\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 11.11 on page
   number 400 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 11.11 on
   page number 400 of the book.")
```

Scilab code Exa 11.12 Proving a mathematical relation

```
clear;
clc;

//Example - 11.12
//Page number - 401
printf("Example - 11.12 and Page number - 401\n\n");
```

```

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 11.12 on page
   number 401 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 11.12 on
   page number 401 of the book.")
```

Scilab code Exa 11.13 Proving a mathematical relation

```
clear;
clc;

//Example - 11.13
//Page number - 405
printf("Example - 11.13 and Page number - 405\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 11.13 on page
   number 405 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 11.13 on
   page number 405 of the book.")
```

Chapter 12

Partial Molar Volume and Enthalpy from Experimental Data

Scilab code Exa 12.1 Calculation of partial molar volume

```
clear;
clc;

//Example - 12.1
//Page number - 419
printf("Example - 12.1 and Page number - 419\n\n");

//Given
//component 1 = methanol
//component 2 = water
T = 0 + 273.15;//[K] - Temperature
P = 1;//[atm] - Pressure
x_methanol = 0.5;//Mole fraction of methanol at
    which molar volume is to be calculated
x_water = 0.5;//Mole fraction at which molar volume
    is to be calculated

```

```
//V = V1 at x1 = 1 and V = V2 at x1 = 0, therefore
V1 = 40.7;//[cm^(3)/mol] - Molar volume of pure
    component 1
V2 = 18.1;//[cm^(3)/mol] - Molar volume of pure
    component 2

x1=[0.114,0.197,0.249,0.495,0.692,0.785,0.892];//
    Values of mole fraction of component 1
V=[20.3,21.9,23.0,28.3,32.9,35.2,37.9];// Values of
    molar volume
x2=zeros(1,7);// Mole fraction of component 2
x_V=zeros(1,7);// x_V = x1*V_1 + x2*V_2
V_mix=zeros(1,7);// V_mix = V - x1*V_1 - x2*V_2
del_V=zeros(1,7);//del_V = V_mix/(x1*x2)

for i=1:7;
    x2(1,i)=1-x1(i);
    x_V(1,i)=x1(i)*V1 + x2(i)*V2;
    V_mix(1,i)=V(i)-x1(i)*V1- x2(i)*V2;
    del_V(1,i)=V_mix(i)/(x1(i)*x2(i));
end

//Now employing the concept of quadratic regression
    of the data ( x1 , del_V ) to solve the equation
    of the type
//y = a0 + a1*x + a2*x^(2)
//Here the above equation is in the form of
//del_V = V_mix/(x1*x2) = a0 + a1*x1 + a2*x1^(2)

//From the matrix method to solve simultaneous
    linear equations, we have
a=[7 sum(x1) sum(x1^2);sum(x1) sum(x1^2) sum(x1^3);
    sum(x1^2) sum(x1^3) sum(x1^4)];
b=[sum(del_V);sum(x1.*del_V);sum((x1^2).*del_V)];
soln=a\b;
a0=soln(1);
a1=soln(2);
a2=soln(3);
```

```

//del_V = V_mix/(x1*x2) = a0 + a1*x1 + a2*x1^(2)
//V_mix = (a0 + a1*x1 + a2*x1^(2))*(x1*(1 - x1))
// For x1 = 0.5
x1  = 0.5;
V_mix_prime = (a0+(a1*x1)+(a2*x1^2))*(x1*(1-x1));//[
   cm^(3)/mol]

//Now differentiating the above equation with
   respect to x we get
//d/dx(V_mix) = (-4*a2*x1^3) + (3*(a2-a1)*x1^2) +
   (2*(a1-a0)*x1)+ a0
//Again for x1 = 0.5
x1_prime = 0.5;
del_V_mix_prime = (-4*a2*x1_prime^3)+(3*(a2-a1)*
   x1_prime^2)+(2*(a1-a0)*x1_prime)+a0;

//Finally,calculating the partial molar volumes
V1_bar = V1 + V_mix_prime + x_water*del_V_mix_prime;
   //[cm^(3)/mol]
V2_bar = V2 + V_mix_prime - x_methanol*
   del_V_mix_prime;//[cm^(3)/mol]

printf("The partial molar volume of methanol (
   component 1) is %f cm^(3)/mol\n\n",V1_bar);
printf("The partial molar volume of water (component
    2) is %f cm^(3)/mol",V2_bar);
```

Scilab code Exa 12.2 Determination of volume of the mixture

```
clear;
clc;

//Example - 12.2
//Page number - 421
```

```
printf("Example - 12.2 and Page number - 421\n\n");

//Given
//component 1 = water
//component 2 = methanol
T = 25 + 273.15;//[K] - Temperature

//delta_V_mix = x_1*x_2*(-3.377 - 2.945*x_1 + 3.31*
   x_1^(2))
V1 = 18.0684;//[cm^(3)/mol] - Molar volume of pure
   component 1
V2 = 40.7221;//[cm^(3)/mol] - Molar volume of pure
   component 2
Vol_1 = 1000;//[cm^(3)] - Volume of pure component 1
Vol_2 = 1000;//[cm^(3)] - Volume of pure component 2

//Moles of the componenets can be calculated as
n_1 = Vol_1/V1;//[mol]
n_2 = Vol_2/V2;//[mol]

//Mole fraction of the components
x_1 = n_1/(n_1 + n_2);
x_2 = n_2/(n_1 + n_2);

delta_V_mix = x_1*x_2*(-3.377 - 2.945*x_1 + 3.31*x_1
   ^(2));//[cm^(3)/mol]

//Differentiating the above equation, we get
//d/dx(delta_V_mix) = (1 - 2*x_1)*(-3.377 - 2.945*
   x_1 + 3.31*x_1^(2)) + (x_1 - x_1^(2))*(-2.945 +
   6.62*x_1)
del_delta_V_mix = (1 - 2*x_1)*(-3.377 - 2.945*x_1 +
   3.31*x_1^(2)) + (x_1 - x_1^(2))*(-2.945 + 6.62*
   x_1);//[cm^(3)/mol]

//Now calculating the partial molar volumes
V1_bar = V1 + delta_V_mix + x_1*del_delta_V_mix;//[
   cm^(3)/mol]
```

```
V2_bar = V2 + delta_V_mix - x_2*del_delta_V_mix;//[
   cm^(3)/mol]

//Finally molar volume of the solution is given by
V_sol = x_1*V1_bar + x_2*V2_bar;//[cm^(3)/mol]

// Total volume of the solution is given by
V_total = (n_1 + n_2)*V_sol;//[cm^(3)]

printf("The molar volume of the solution is %f cm
   ^(3)/mol\n\n",V_sol);
printf("The total volume of the solution is %f cm
   ^(3)",V_total);
```

Scilab code Exa 12.3 Determination of volumes

```
clear;
clc;

//Example - 12.3
//Page number - 422
printf("Example - 12.3 and Page number - 422\n\n");

//Given
//component 1 = methanol
//component 2 = water
Vol = 20;//[cm^(3)] - Volume of the solution
T = 22 + 273.15;//[K] - Temperature
W_bottle = 11.5485;//[g] - Weight of density bottle
Mol_meth = 32.04;//Molecular weight of methanol
Mol_water = 18.015;// Molecular weight of water

//Density of pure components can be found out at 0%
   and 100% of volume percent.
den_meth = 0.7929;//[cm^(3)/mol] - Density of pure
```

```
methanol
den_water = 0.9937;//[cm^(3)/mol] - Density of pure water


Vol_perc=[5,10,20,30,40,50,60,70,80,90,95];// Volumes percent of component 1 (methanol)
W_total=[31.2706,31.1468,30.8907,30.6346,30.3396,30.0053,29.5865,29.1453
// Weight of solution + weight of density bottle

W_sol=zeros(1,11);// Weight of 20 cm^(3) of solution
den=zeros(1,11);// density of the solution
x1=zeros(1,11);// Mole fraction of methanol
x2=zeros(1,11);// Mole fraction of water

for i=1:11;
    W_sol(1,i)=W_total(i)-W_bottle;
    den(1,i)=W_sol(i)/Vol;
    x1(1,i)=((Vol_perc(i)*den_meth)/Mol_meth)/(((Vol_perc(i)*den_meth)/Mol_meth)+(((100-Vol_perc(i))*den_water)/Mol_water));
    x2(1,i)=1-x1(1,i);
end

//Again we have,
V_kg=zeros(1,11);//[cm^(3)] - Volume of 1 kg of solution
n_mol=zeros(1,11);//[mol] - Number of moles in 1 kg of solution
V_mol=zeros(1,11);//[cm^(3)/mol] - Volume of 1 mol of solution
x_V=zeros(1,11);//[cm^(3)/mol] - x_V = x1*V_meth + x2*V_water
V_mix=zeros(1,11);//[cm^(3)/mol] - V_mix = V_mol - x1*V_meth - x2*V_water
del_V=zeros(1,11);// [cm^(3)/mol] - del_V = V_mix/(x1*x2)
```

```

//V_mol = V_meth at x1 = 1 and V_mol = V_water at x1
    = 0, therefore
V_meth = 40.4114;//[cm^(3)/mol] - Molar volume of
    pure component 1 (methanol)
V_water = 18.1286;//[cm^(3)/mol] - Molar volume of
    pure component 2 (water)

for i=1:11;
    V_kg(1,i)=1000/den(i);
    n_mol(1,i)=1000/(x1(i)*Mol_meth+x2(i)*Mol_water)
        ;
    V_mol(1,i)=V_kg(i)/n_mol(i);
    x_V(1,i)=V_meth*x1(i)+V_water*x2(i);
    V_mix(1,i)=V_mol(i)-x1(i)*V_meth-x2(i)*V_water;
    del_V(1,i)=V_mix(i)/(x1(i)*x2(i));
end

//Now employing the concept of quadratic regression
    of the data ( x1 , del_V ) to solve the equation
    of the type
//y = a0 + a1*x + a2*x^(2)
//Here the above equation is in the form of
//del_V = V_mix/(x1*x2) = a0 + a1*x1 + a2*x1^(2)

//From the matrix method to solve simultaneous
    linear equations, we have
a=[11 sum(x1) sum(x1^2);sum(x1) sum(x1^2) sum(x1^3);
    sum(x1^2) sum(x1^3) sum(x1^4)];
b=[sum(del_V);sum(x1.*del_V);sum((x1^2).*del_V)];
soln=a\b;
a0=soln(1);
a1=soln(2);
a2=soln(3);

//del_V = V_mix/(x1*x2) = a0 + a1*x1 + a2*x1^(2)
//V_mix = (a0 + a1*x1 + a2*x1^(2))*(x1*(1 - x1))
//Solving the above equation for x1,
deff('[y]=f(x1)','y=(a0+(a1*x1)+(a2*x1^2))*(x1*(1-x1))');

//Now differentiating the above equation with respect to x we get
//d/dx(V_mix) = (-4*a2*x1^3) + (3*(a2-a1)*x1^2) + (2*(a1-a0)*x1)+ a0
//Again solving it for x1
deff('[y]=f1(x1)','y=(-4*a2*x1^3)+(3*(a2-a1)*x1^2)+(2*(a1-a0)*x1)+a0');

//Now

x1_prime=[0,0.25,0.50,0.75,1.0];
V_mix_prime=zeros(1,5);//[cm^(3)/mol] -  V_mix = V - x1*V_meth - x2*V_water
del_V_prime=zeros(1,5);//[cm^(3)/mol] - del_V = V_mix/(x1*x2)
V1_bar=zeros(1,5);//[cm^(3)/mol] - Partial molar volume of component 1
V2_bar=zeros(1,5);//[cm^(3)/mol] - Partial molar volume of component 1

for j=1:5;
    V_mix_prime(j)=f(x1_prime(j));
    del_V_prime(j)=f1(x1_prime(j));
    V1_bar(j)=V_meth+V_mix_prime(j)+(1-x1_prime(j))*del_V_prime(j);
    V2_bar(j)=V_water+V_mix_prime(j)-x1_prime(j)*del_V_prime(j);
    printf("For x1 = %f\n",x1_prime(j));
    printf("The partial molar volume of methanol (component 1) is %f cm^(3)/mol\n",V1_bar(j));
    printf("The partial molar volume of water (component 2) is %f cm^(3)/mol\n\n",V2_bar(j));
end
```

Scilab code Exa 12.4 Determination of partial molar volumes

```
clear;
clc;

//Example - 12.4
//Page number - 424
printf("Example - 12.4 and Page number - 424\n\n");

//Given
//component 1 = formic acid
//component 2 = water
T = 20 + 273.15;//[K] - Temperature
Mol_form = 46.027;//Molecular weight of formic acid
Mol_water = 18.015;// Molecular weight of water

Wt_perc=[10,18,30,50,72,78];//Weight percent of
    formic acid
den=[1.0246,1.0441,1.0729,1.1207,1.1702,1.1818];//[g
    /cm^(3)] - Density of solution

V_g=zeros(1,6);//[cm^(3)/g] - Volume of 1 g of
    solution
x1=zeros(1,6);// Mole fraction of component 1
x2=zeros(1,6);// Mole fraction of component 2
n=zeros(1,6);// Number of moles in 1 g
V_mol=zeros(1,6);//[cm^(3)/mol] - Volume of 1 mol of
     solution
x_V=zeros(1,6);//[cm^(3)/mol] - x_V = x1*V_form + x2
    *V_water
V_mix=zeros(1,6);//[cm^(3)/mol] -  V_mix = V - x1*
    V_form - x2*V_water
del_V=zeros(1,6);// [cm^(3)/mol] - del_V = V_mix/(x1
    *x2)
```

```

//V_mol = V_form at x1 = 1 and V_mol = V_water at x1
    = 0, therefore
V_form = 37.737;//[cm^(3)/mol] - Molar volume of
   pure formic acid (component 1)
V_water = 18.050;//[cm^(3)/mol] - Molar volume of
   pure water (component 2)

for i=1:6;
    V_g(i)=1/den(i);
    x1(1,i)=(Wt_perc(i)/Mol_form)/((Wt_perc(i)/
       Mol_form)+((100-Wt_perc(i))/Mol_water));
    x2(1,i)=1-x1(i);
    n(1,i)=((Wt_perc(i)/100)/Mol_form)+(((100-
       Wt_perc(i))/100)/Mol_water);
    V_mol(1,i)=V_g(i)/n(i);
    x_V(1,i)=V_form*x1(i)+V_water*x2(i);
    V_mix(1,i)=V_mol(i)-x1(i)*V_form-x2(i)*V_water;
    del_V(1,i)=V_mix(i)/(x1(i)*x2(i));
end

//Now employing the concept of quadratic regression
   of the data ( x1 , del_V ) to solve the equation
   of the type
//y = a0 + a1*x + a2*x^(2)
//Here the above equation is in the form of
//del_V = V_mix/(x1*x2) = a0 + a1*x1 + a2*x1^(2)

//From the matrix method to solve simultaneous
   linear equations , we have
a=[11 sum(x1) sum(x1^2);sum(x1) sum(x1^2) sum(x1^3);
   sum(x1^2) sum(x1^3) sum(x1^4)];
b=[sum(del_V);sum(x1.*del_V);sum((x1^2).*del_V)];
soln=a\b;
a0=soln(1);
a1=soln(2);
a2=soln(3);

```

```
//del_V = V_mix/(x1*x2) = a0 + a1*x1 + a2*x1^(2)
//V_mix = (a0 + a1*x1 + a2*x1^(2))*(x1*(1 - x1))
//Solving the above equation for x1,
deff('[y]=f(x1)','y=(a0+(a1*x1)+(a2*x1^2))*(x1*(1-x1
    ))');

//Now differentiating the above equation with
    respect to x we get
//d/dx(V_mix) = (-4*a2*x1^3) + (3*(a2-a1)*x1^2) +
    (2*(a1-a0)*x1)+ a0
//Again solving it for x1
deff('[y]=f1(x1)','y=(-4*a2*x1^3)+(3*(a2-a1)*x1^2)
    +(2*(a1-a0)*x1)+a0');

//At 15 Wt% of formic acid, x1 is given by
x1_prime_1 = (15/Mol_form)/((15/Mol_form)+((100-15)/
    Mol_water));
//Similarly at 75 Wt% of formic acid, x1 is given by
x1_prime_2 = (75/Mol_form)/((75/Mol_form)+((100-75)/
    Mol_water));

Wt_perc_prime=[15,75];
x1_prime=[x1_prime_1,x1_prime_2];
V_mix_prime=zeros(1,2);//[cm^(3)/mol] -  V_mix = V -
     x1*V_meth - x2*V_water
del_V_prime=zeros(1,2);//[cm^(3)/mol] - del_V =
    V_mix/(x1*x2)
V1_bar=zeros(1,2);//[cm^(3)/mol] - Partial molar
    volume of component 1
V2_bar=zeros(1,2);//[cm^(3)/mol] - Partial molar
    volume of component 1

for j=1:2;
    V_mix_prime(j)=f(x1_prime(j));
    del_V_prime(j)=f1(x1_prime(j));
    V1_bar(j)=V_form+V_mix_prime(j)+(1-x1_prime(j))*
        del_V_prime(j);
    V2_bar(j)=V_water+V_mix_prime(j)-x1_prime(j)*
```

```
        del_V_prime(j);
    printf("For weight percent of formic acid = %f
        percent\n",Wt_perc_prime(j));
    printf("The partial molar volume of formic acid
        (component 1) is %f cm^(3)/mol\n",V1_bar(j));
    printf("The partial molar volume of water (
        component 2) is %f cm^(3)/mol\n\n",V2_bar(j))
        ;
end
```

Scilab code Exa 12.5 Determination of enthalpy

```
clear;
clc;

//Example - 12.5
//Page number - 426
printf("Example - 12.5 and Page number - 426\n\n");

//Given
T = 40 + 273.15;//[K] - Temperature

x1
   =[0.083,0.176,0.268,0.353,0.428,0.720,0.780,0.850,0.900];
   // Mole fraction of component 1
delta_H_mix
   =[0.250,0.488,0.670,0.790,0.863,0.775,0.669,0.510,0.362];
   //[kJ/mol] - Enthalpy of the solution

x2=zeros(1,9);// Mole fraction of component 2
del_H=zeros(1,9);//[kJ/mol] - del_H = delta_H_mix/(
   x1*x2)

for i=1:9;
    x2(1,i)=1-x1(i);
    del_H(1,i)=delta_H_mix(i)/(x1(i)*x2(i));
end

//Now employing the concept of quadratic regression
   of the data ( x1 , del_H ) to solve the equation
   of the type
//y = a0 + a1*x + a2*x^(2)
//Here the above equation is in the form of
//del_H = delta_H_mix/(x1*x2) = a0 + a1*x1 + a2*x1
   ^(2)

//From the matrix method to solve simultaneous
   linear equations , we have
a=[9 sum(x1) sum(x1^2);sum(x1) sum(x1^2) sum(x1^3);
   sum(x1^2) sum(x1^3) sum(x1^4)];
b=[sum(del_H);sum(x1.*del_H);sum((x1^2).*del_H)];
soln=a\b;
a0=soln(1);
a1=soln(2);
a2=soln(3);

//del_H = delta_H_mix/(x1*x2) = a0 + a1*x1 + a2*x1
   ^(2)
//delta_H_mix = (a0 + a1*x1 + a2*x1^(2))*(x1*(1 - x1
   ))
//At x1 = 0.25,
x_1 = 0.25;//[mol]
delta_H_mix = (a0+(a1*x_1)+(a2*x_1^2))*(x_1*(1-x_1))
   ;//[kJ/mol]

//Now differentiating the above equation with
   respect to x we get
//d/dx(delta_H_mix) = del_delta_H_mix = (-4*a2*x1^3)
    + (3*(a2-a1)*x1^2) + (2*(a1-a0)*x1)+ a0
//Again  for x1 = 0.25
x_1_prime = 0.25;//[mol]
del_delta_H_mix = (-4*a2*x_1_prime^3)+(3*(a2-a1)*
   x_1_prime^2)+(2*(a1-a0)*x_1_prime)+a0;//[kJ/mol]
```

```

//We have the relation
// H1_bar - H1 = delta_H_mix + x2*del_delta_H_mix ,
   and
// H2_bar - H2 = delta_H_mix - x1*del_delta_H_mix

//Let us suppose
//k_1 = H1_bar - H1 , and
//k_2 = H2_bar - H2

k_1 = delta_H_mix + (1-x_1_prime)*del_delta_H_mix;//
   [kJ/mol]
k_2 = delta_H_mix - x_1_prime*del_delta_H_mix;//[kJ/
   mol]


printf("The value of (H1_bar - H1) at x1 = 0.25 is
   %f kJ/mol\n\n",k_1);
printf("The value of (H2_bar - H2) at x1 = 0.25 is
   %f kJ/mol",k_2);
```

Chapter 13

Fugacity of a Component in a Mixture by Equations of State

Scilab code Exa 13.1 Proving a mathematical relation

```
clear;
clc;

//Example - 13.1
//Page number - 432
printf("Example - 13.1 and Page number - 432\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 13.1 on page
   number 432 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 13.1 on
   page number 432 of the book.")
```

Scilab code Exa 13.2 Caculation of fugacity coefficients

```
clear;
clc;

//Example - 13.2
//Page number - 433
printf("Example - 13.2 and Page number - 433\n\n");

//Given
T = 310.93;//[K] - Temperature
P = 2.76*10^(6);//[Pa] - Pressure
y1 = 0.8942;//[mol] - mole fraction of component 1
y2 = 1 - y1;//[mol] - mole fraction of component 2
R=8.314;//[J/mol*K] - Universal gas constant

//For component 1 (methane)
Tc_1 = 190.6;//[K] - Critical temperature
Pc_1 = 45.99*10^(5);//[N/m^(2)] - Critical pressure
Vc_1 = 98.6;//[cm^(3)/mol] - Critical molar volume
Zc_1 = 0.286;// - Critical compressibility factor
w_1 = 0.012;// - Critical acentric factor
//Similarly for component 2 (n-Butane)
Tc_2 = 425.1;//[K]
Pc_2 = 37.96*10^(5);//[N/m^(2)]
Vc_2 = 255;//[cm^(3)/mol]
Zc_2=0.274;
w_2=0.2;

//For component 1
Tr_1 = T/Tc_1;//Reduced temperature
//At reduced temperature
B1_0 = 0.083-(0.422/(Tr_1)^(1.6));
B1_1 = 0.139-(0.172/(Tr_1)^(4.2));
//We know,(B*Pc)/(R*Tc) = B_0+(w*B_1)
B_11 = ((B1_0+(w_1*B1_1))*(R*Tc_1))/Pc_1;//[m^(3)/
    mol]

```

```
//Similarly for component 2
Tr_2 = T/Tc_2;//Reduced temperature
//At reduced temperature Tr_2,
B2_0 = 0.083-(0.422/(Tr_2)^(1.6));
B2_1 = 0.139-(0.172/(Tr_2)^(4.2));
B_22 = ((B2_0+(w_2*B2_1))*(R*Tc_2))/Pc_2;//[m^(3)/
   mol]

//For cross coeffcient
Tc_12 = (Tc_1*Tc_2)^(1/2);//[K]
w_12 = (w_1+w_2)/2;
Zc_12 = (Zc_1+Zc_2)/2;
Vc_12 = (((Vc_1)^(1/3)+(Vc_2)^(1/3))/2)^(3);//[cm
   ^(3)/mol]
Vc_12 = Vc_12*10^(-6);//[m^(3)/mol]
Pc_12 = (Zc_12*R*Tc_12)/Vc_12;//[N/m^(2)]

//Given, Z = 1 + (B*P)/(R*T)
//Now we have,(B_12*Pc_12)/(R*Tc_12) = B_0 + (w_12*
   B_1)
//where B_0 and B_1 are to be evaluated at Tr_12
Tr_12 = T/Tc_12;
//At reduced temperature Tr_12
B_0 = 0.083-(0.422/(Tr_12)^(1.6));
B_1 = 0.139-(0.172/(Tr_12)^(4.2));
B_12 = ((B_0+(w_12*B_1))*R*Tc_12)/Pc_12;//[m^(3)/mol
   ]
//For the mixture
B = y1^(2)*B_11+2*y1*y2*B_12+y2^(2)*B_22;//[m^(3)/
   mol]

//Now del_12 can be calculated as,
del_12 = 2*B_12 - B_11 - B_22;//[m^(3)/mol]

//We have the relation, log(phi_1) = (P/(R*T))*(B_11
    + y2^(2)*del_12), therefore
phi_1 = exp((P/(R*T))*(B_11 + y2^(2)*del_12));
//Similarly for component 2
```

```
phi_2 = exp((P/(R*T))*(B_22 + y1^(2)*del_12));

printf(" The value of fugacity coefficient of
   component 1 (phi_1) is %f\n\n",phi_1);
printf(" The value of fugacity coefficient of
   component 2 (phi_2) is %f\n\n",phi_2);

//Finally fugacity coefficient of the mixture is
   given by
//log(phi) = y1*log(phi_1) + y2*log(phi_2);
phi = exp(y1*log(phi_1) + y2*log(phi_2));

printf(" The value of fugacity coefficient of the
   mixture (phi) is %f ",phi);
//The fugacity coefficient of the mixture can also
   be obtained using
//log(phi) = (B*P)/(R*T)
```

Scilab code Exa 13.3 Proving a mathematical relation

```
clear;
clc;

//Example - 13.3
//Page number - 435
printf("Example - 13.3 and Page number - 435\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 13.3 on page
   number 435 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 13.3 on
   page number 435 of the book.")
```

Scilab code Exa 13.4 Proving a mathematical relation

```
clear;
clc;

//Example - 13.4
//Page number - 436
printf("Example - 13.4 and Page number - 436\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 13.4 on page
   number 436 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 13.4 on
   page number 436 of the book.")
```

Scilab code Exa 13.5 Proving a mathematical relation

```
clear;
clc;

//Example - 13.5
//Page number - 442
printf("Example - 13.5 and Page number - 442\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
```

```
//For prove refer to this example 13.5 on page
   number 442 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 13.5 on
   page number 442 of the book.")
```

Scilab code Exa 13.6 Proving a mathematical relation

```
clear;
clc;

//Example - 13.6
//Page number - 446
printf("Example - 13.6 and Page number - 446\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 13.6 on page
   number 446 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 13.6 on
   page number 446 of the book.")
```

Scilab code Exa 13.7 Calculation of fugacity coefficients

```
clear;
clc;

// Example - 13.7
// Page number - 447
printf("Example - 13.7 and Page number - 447\n\n");

//Given
T = 460;//[K] - Temperature
P = 40*10^(5);//[Pa] - Pressure
R=8.314;//[J/mol*K] - Universal gas constant
// component 1 = nitrogen
// component 2 = n-Butane
y1 = 0.4974;// Mole percent of nitrogen
y2 = 0.5026;// Mole percent of n-Butane
Tc_nit = 126.2;//[K]
Pc_nit = 34.00*10^(5);//[Pa]
Tc_but = 425.1;//[K]
Pc_but = 37.96*10^(5);//[Pa]

// (1). van der Walls equation of state

// The fugacity coefficient of component 1 in a
   binary mixture following van der Walls equation
   of state is given by,
// log(phi_1) = b_1/(V-b) - log(Z-B) -2*(y1*a_11 +
   y2*a_12)/(R*T*V)
// and for component 2 is given by,
// log(phi_2) = b_2/(V-b) - log(Z-B) -2*(y1*a_12 +
   y2*a_22)/(R*T*V)
// Where B = (P*b)/(R*T)

// For componenet 1 (nitrogen)
a_1 = (27*R^(2)*Tc_nit^(2))/(64*Pc_nit);//[Pa-m^(6)/
   mol^(2)]
b_1 = (R*Tc_nit)/(8*Pc_nit);//[m^(3)/mol]

// Similarly for componenet 2 (n-Butane)
a_2 = (27*R^(2)*Tc_but^(2))/(64*Pc_but);//[Pa-m^(6)/
   mol^(2)]
b_2 = (R*Tc_but)/(8*Pc_but);//[m^(3)/mol]

```

```
// Here
a_11 = a_1;
a_22 = a_2;
// For cross coefficient
a_12 = (a_1*a_2)^(1/2);//[Pa-m^(6)/mol^(2)]

// For the mixture
a = y1^(2)*a_11 + y2^(2)*a_22 + 2*y1*y2*a_12;//[Pa-m
   ^(6)/mol^(2)]
b = y1*b_1 + y2*b_2;//[m^(3)/mol]

// The cubic form of the van der Walls equation of
   state is given by,
// V^(3) - (b+(R*T)/P)*V^(2) + (a/P)*V - (a*b)/P = 0
// Substituting the value and solving for V, we get
// Solving the cubic equation
deff('[y]=f(V)','y=V^(3)-(b+(R*T)/P)*V^(2)+(a/P)*V-(
   a*b)/P');
V_1=fsolve(-1,f);
V_2=fsolve(0,f);
V_3=fsolve(1,f);
// The molar volume V = V_3, the other two roots are
    imaginary
V = V_3;//[m^(3)/mol]

// The comprssibility factor of the mixture is
Z = (P*V)/(R*T);
// And B can also be calculated as
B = (P*b)/(R*T);

// The fugacity coefficient of component 1 in the
   mixture is
phi_1 = exp(b_1/(V-b) - log(Z-B) -2*(y1*a_11 + y2*
   a_12)/(R*T*V));
// Similarly fugacity coefficient of component 2 in
   the mixture is
phi_2 = exp(b_2/(V-b) - log(Z-B) -2*(y1*a_12 + y2*
   a_22)/(R*T*V));
```

```

// The fugacity coefficient of the mixture is given
   by,
// log(phi) = y1*log(phi_1) + y2*log(phi_2)
phi = exp(y1*log(phi_1) + y2*log(phi_2));

// Also the fugacity coefficient of the mixture
   following van der Walls equation of state is
   given by,
// log(phi) = b/(V-b) - log(Z-B) -2*a/(R*T*V)
phi_dash = exp(b/(V-b) - log(Z-B) -2*a/(R*T*V));
// The result is same as obtained above

printf(" (1)van der Walls equation of state\n");
printf("  The value of fugacity coefficient of
   component 1 (nitrogen) is %f\n",phi_1);
printf("  The value of fugacity coefficient of
   component 2 (n-butane) is %f\n",phi_2);
printf("  The value of fugacity coefficient of the
   mixture is %f\n",phi);
printf("  Also the fugacity coefficient of the
   mixture from van der Walls equation of state is
   %f (which is same as above)\n\n",phi_dash);

// (2). Redlich-Kwong equation of state

// For component 1,
a_1_prime = (0.42748*R^(2)*Tc_nit^(2.5))/Pc_nit;//[
   Pa-m^(6)/mol^(2)]
b_1_prime = (0.08664*R*Tc_nit)/Pc_nit;//[m^(3)/mol]

//similarly for component 2,
a_2_prime = (0.42748*R^(2)*Tc_but^(2.5))/Pc_but;//[
   Pa-m^(6)/mol^(2)]
b_2_prime = (0.08664*R*Tc_but)/Pc_but;//[m^(3)/mol]

// For cross coefficient
a_12_prime = (a_1_prime*a_2_prime)^(1/2);//[Pa-m^(6)
```

```
   /mol^(2)]
// For the mixture
a_prime = y1^(2)*a_1_prime + y2^(2)*a_2_prime +2*y1*
   y2*a_12_prime;//[Pa-m^(6)/mol^(2)]
b_prime = y1*b_1_prime +y2*b_2_prime;//[m^(3)/mol]


//The cubic form of Redlich Kwong equation of state
   is given by,
//  V^(3)-((R*T)/P)*V^(2)-((b^(2))+((b*R*T)/P)-(a/(T
   ^(1/2)*P))*V-(a*b)/(T^(1/2)*P)=0
// Solving the cubic equation
deff('[y]=f1(V)','y=V^(3)-((R*T)/P)*V^(2)-((b_prime
   ^(2))+((b_prime*R*T)/P)-(a_prime/(T^(1/2)*P)))*V
   -(a_prime*b_prime)/(T^(1/2)*P)');
V_4=fsolve(1,f1);
V_5=fsolve(0,f1);
V_6=fsolve(-1,f1);
// The molar volume V = V_4, the other two roots are
    imaginary
V_prime = V_4;//[m^(3)/mol]

// The compressibility factor of the mixture is
Z_prime = (P*V_prime)/(R*T);
// And B can also be calculated as
B_prime = (P*b_prime)/(R*T);

// The fugacity coefficient of component 1 in the
   binary mixture is given by
// log(phi_1) = b_1/(V-b) - log(Z-B) + ((a*b_1)/((b
   ^(2)*R*T^(3/2))))*(log((V+b)/V)-(b/(V+b)))-(2*(y1
   *a_1+y2*a_12)/(R*T^(3/2)b))*(log(V+b)/b)

phi_1_prime = exp((b_1_prime/(V_prime-b_prime))-log(
   Z_prime-B_prime)+((a_prime*b_1_prime)/((b_prime
   ^(2))*R*(T^(3/2))))*(log((V_prime+b_prime)/
   V_prime)-(b_prime/(V_prime+b_prime)))-(2*(y1*
   a_1_prime+y2*a_12_prime)/(R*(T^(3/2))*b_prime))*(
   log((V_prime+b_prime)/V_prime)));


// Similarly fugacity coefficient of component 2 in
   the mixture is
phi_2_prime = exp((b_2_prime/(V_prime-b_prime))-log(
   Z_prime-B_prime)+((a_prime*b_2_prime)/((b_prime
   ^(2))*R*(T^(3/2))))*(log((V_prime+b_prime)/
   V_prime)-(b_prime/(V_prime+b_prime)))-(2*(y1*
   a_12_prime+y2*a_2_prime)/(R*(T^(3/2))*b_prime))*(
   log((V_prime+b_prime)/V_prime)));

// The fugacity coefficient of the mixture is given
   by,
// log(phi) = y1*log(phi_1) + y2*log(phi_2)
phi_prime = exp(y1*log(phi_1_prime) + y2*log(
   phi_2_prime));

// Also the fugacity coefficient for the mixture
   following Redlich-kwong equation of state is also
    given by
// log(phi) = b/(V-b) - log(Z-B) - (a/(R*T^(3/2)))
   *(1/(V+b)+(1/b)*log((V+b)/V))
phi_prime_dash = exp(b_prime/(V_prime-b_prime) - log
   (Z_prime-B_prime) - (a_prime/(R*T^(3/2)))*(1/(
   V_prime+b_prime)+(1/b_prime)*log((V_prime+b_prime
   )/                      V_prime)));

printf(" (2)Redlich-Kwong equation of state\n");
printf("  The value of fugacity coefficient of
   component 1 (nitrogen) is %f\n",phi_1_prime);
printf("  The value of fugacity coefficient of
   component 2 (n-butane) is %f\n",phi_2_prime);
printf("  The value of fugacity coefficient of the
   mixture is %f\n",phi_prime);
printf("  Also the fugacity coefficient for the
   mixture from Redlich-kwong equation of state is
   %f (which is same as above)\n\n",phi_prime_dash);
```

Chapter 14

Activity Coefficients Models for Liquid Mixtures

Scilab code Exa 14.1 Determination of expression for activity coefficients

```
clear;
clc;

//Example - 14.1
//Page number - 455
printf("Example - 14.1 and Page number - 455\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 14.1 on page
   number 455 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.1 on
   page number 455 of the book.")
```

Scilab code Exa 14.2 Proving a mathematical relation

```
clear;
clc;

//Example - 14.2
//Page number - 456
printf("Example - 14.2 and Page number - 456\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 14.2 on page
   number 456 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.2 on
   page number 456 of the book.")
```

Scilab code Exa 14.3 Proving a mathematical relation

```
clear;
clc;

//Example - 14.3
//Page number - 458
printf("Example - 14.3 and Page number - 458\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 14.3 on page
   number 458 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
```

```
printf(" For prove refer to this example 14.3 on
    page number 458 of the book.")
```

Scilab code Exa 14.4 Determination of value of Gibbs free energy change and enthalpy change

```
clear;
clc;

//Example - 14.4
//Page number - 461
printf("Example - 14.4 and Page number - 461\n\n");

//Given ,
T = 300;//[K] - Temperature
b = 100;//[cal/mol]
R = 1.987;//[cal/mol*K] - Universal gas constant
// R*T*log(Y_1) = b*x_2^(2)
// R*T*log(Y_2) = b*x_1^(2)

//For equimolar mixture
x_1 = 0.5;//Mole fraction of component 1
x_2 = 0.5;//Mole fraction of component 2

//The excess Gibbs free energy is given by
// G_excess = R*T*(x_1*log(Y_1) + x_2*log(Y_2)) = b*
    x_1*x_2^(2) + b*x_2*x_1^(2) = b*x_1*(x_1 + x_2) =
     b*x_1*x_2
G_excess = b*x_1*x_2;//[cal/mol]

//The ideal Gibbs free energy change of mixing is
    given by,
delta_G_id_mix = R*T*(x_1*log(x_1)+x_2*log(x_2));//[
    cal/mol]

```

```
//The Gibbs free energy of mixing is given by
delta_G_mix = delta_G_id_mix + G_excess;//[cal/mol]

//It is given that entropy change of mixing is that
    of ideal mixture,therefore
// delta_S_mix = delta_S_id_mix = - R*sum(x_i*log(
    x_i))

//delta_G_mix = delta_H_mix - T*delta_S_mix =
    delta_H_mix + R*T*(x_1*log(x_1)+x_2*log(x_2))
delta_H_mix = b*x_1*x_2;//[cal/mol]

printf("The value of Gibbs free energy change for
    equimolar mixture formation is %f cal/mol\n\n",
    delta_G_mix);
printf("The value of enthalpy change for equimolar
    mixture formation is %f cal/mol\n\n",delta_H_mix)
    ;

//Work required for separation of mixture into pure
    components is
W = delta_G_mix;
printf("The least amount of work required for
    separation at 300 K is %f cal/mol\n\n",W);
```

Scilab code Exa 14.5 Proving a mathematical relation

```
clear;
clc;

//Example - 14.5
//Page number - 461
printf("Example - 14.5 and Page number - 461\n\n");

//This problem involves proving a relation in which
```

```
   no mathematics and no calculations are involved.
//For prove refer to this example 14.5 on page
   number 461 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.5 on
   page number 461 of the book.")
```

Scilab code Exa 14.6 Proving a mathematical relation

```
clear;
clc;

//Example - 14.6
//Page number - 463
printf("Example - 14.6 and Page number - 463\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 14.6 on page
   number 463 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.6 on
   page number 463 of the book.")
```

Scilab code Exa 14.7 Proving a mathematical relation

```
clear;
clc;

```

```
//Example - 14.7
//Page number - 464
printf("Example - 14.7 and Page number - 464\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 14.7 on page
   number 464 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.7 on
   page number 464 of the book.")
```

Scilab code Exa 14.8 Proving a mathematical relation

```
clear;
clc;

//Example - 14.8
//Page number - 465
printf("Example - 14.8 and Page number - 465\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 14.8 on page
   number 465 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.8 on
   page number 465 of the book.")
```

Scilab code Exa 14.9 Proving a mathematical relation

```
clear;
clc;

//Example - 14.9
//Page number - 465
printf("Example - 14.9 and Page number - 465\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 14.9 on page
   number 465 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.9 on
   page number 465 of the book.")
```

Scilab code Exa 14.10 Comparision of Margules and van Laar eqations

```
clear;
clc;

//Example - 14.10
//Page number - 466
printf("Example - 14.10 and Page number - 466\n\n")

//Given,
T = 25 +173.15;//[K] - Temperature
x_1
   =[0.0115,0.0160,0.0250,0.0300,0.0575,0.1125,0.1775,0.2330,0.4235,

y_1
   =[8.0640,7.6260,7.2780,7.2370,5.9770,4.5434,3.4019,2.8023,1.7694,
y_2
   =[1.0037,1.0099,1.0102,1.0047,1.0203,1.0399,1.1051,1.1695,1.4462,

x_2 = zeros(1,15);// x_2 = (1 - x_1)
G_RT = zeros(1,15);// G_RT = G_excess/(R*T)
x1x2_G_RT = zeros(1,15);// x1x2_G_RT = (x_1*x_2/(
   G_excess/(R*T)))
G_RTx1x2 = zeros(1,15);// G_RTx1x1 = G_excess/(R*T*
   x_1*x_2)

for i=1:15;
    x_2(1,i)=(1-x_1(i));
    G_RT(1,i)=(x_1(i)*log(y_1(i)))+(x_2(i)*log(y_2(i
       )));
    x1x2_G_RT(1,i)=(x_1(i)*x_2(i))/G_RT(i);
    G_RTx1x2(1,i)=1/x1x2_G_RT(i);

end

//From Van Laar equation
// G_RTx1x2 = (x_1*x_2/(G_excess/(R*T))) = 1/A + (1/
   B -1/A)*x_1
//slope = (1/B -1/A) and intercept = 1/A

//Now employing the concept of linear regression of
   the data ( x_1 , x1x2_G_RT ) to find the value of
    intercept and slope of the above equation
//Let slope = slop and intercept = intr

[slop,intr,sig]=reglin(x_1,x1x2_G_RT);

A = 1/intr;
B = 1/(slop+(1/A));
printf(" The value of van Laar parameters are\n A =
   %f and B = %f\n\n",A,B);

```

```
// Now from Margules equation
// G_RTx1x2 = G_excess/(R*T*x_1*x_2) = B1*x_1 + A1*
   x_1 = A1 + (B1 - A1)*x_1
//slope = (B1 - A1) and intercept = A1

// Again employing the concept of linear regression
   of the data ( x_1 , G_RTx1x2 ) to find the value
   of intercept and slope of the above equation
//Let slope = slop1 and intercept = intr1

[slop1,intr1,sig1]=reglin(x_1,G_RTx1x2);

A1 = intr1;
B1 = slop1 + A1;
printf(" The value of Margules parameters are\n A =
   %f and B = %f\n\n",A1,B1);

printf(" Because of the higher value of correlation
   factor for Van Laar model, it fits the data
   better.");
```

Scilab code Exa 14.11 Calculation of activity coefficients

```
clear;
clc;

//Example - 14.11
//Page number - 470
printf("Example - 14.11 and Page number - 470\n\n")

//Given,
T = 60 + 273.15;//[K] - Temperature
R = 1.987;//[cal/mol*K] - Universal gas constant
//component 1 = acetone
//component 2 = water
```

```
x_1 = 0.3;// Mole fraction of component 1
x_2 = 1 - x_1;//Mole fraction of component 2
V_mol_1 = 74.05;//[cm^(3)/mol] - Molar volume of
    pure component 1
V_mol_2 = 18.07;//[cm^(3)/mol] - Molar volume of
    pure component 2

//for Wilson equation
a_12 = 291.27;//[cal/mol]
a_21 = 1448.01;//[cal/mol]

//For NRTL
b_12 = 631.05;//[cal/mol]
b_21 = 1197.41;//[cal/mol]
alpha = 0.5343;

//Froom wilson equation
A_12=(V_mol_2/V_mol_1)*(exp(-a_12/(R*T)));
A_21 = (V_mol_1/V_mol_2)*(exp(-a_21/(R*T)));
Beta = A_12/(x_1+x_2*A_12) - A_21/(x_2+x_1*A_21);
//log(Y1) = -log(x_1 + x_2*A_12) + x_2*Beta;
Y1 = exp(-log(x_1+x_2*A_12)+x_2*Beta);
//similarly for Y2
Y2 = exp(-log(x_2+x_1*A_21)-x_1*Beta);
printf("The value of activity coefficients for
    Wilson equation are\n Y1 = %f \t and \t Y2 = %f\n
    \n",Y1,Y2);

//From NRTL equation,
t_12 = b_12/(R*T);
t_21 = b_21/(R*T);
G_12 = exp(-alpha*t_12);
G_21 = exp(-alpha*t_21);

//log(Y1) = x_1^(2)*[t_12*(G_12/(x_1+x_2*G_12))^(2)
    + (G_12*t_12)/((G_12/(x_1+x_2*G_12))^(2))]
Y1_prime = exp(x_2^(2)*(t_21*(G_21/(x_1+x_2*G_21))
    ^(2)+(G_12*t_12)/(((x_2+x_1*G_12))^(2))));
```

```
//Similarly for Y2
Y2_prime = exp(x_1^(2)*(t_12*(G_12/(x_2+x_1*G_12))
   ^(2)+(G_21*t_21)/(((x_1+x_2*G_21))^(2))));

printf("The value of activity coefficients for NRTL
   equation are\n Y1 = %f \t and \t Y2 = %f\n\n",
   Y1_prime,Y2_prime);
```

Scilab code Exa 14.12 Calculation of the value of activity coefficients

```
clear;
clc;

//Example - 14.12
//Page number - 474
printf("Example - 14.12 and Page number - 474\n\n");

//Given
T = 307;//[K]
x_1 = 0.047;
x_2 = 1 - x_1;

// The subgroups in the  two components are
// Acetone (1) : 1 CH3, 1 CH3CO
// n-pentane (2) : 2 CH3, 3 CH2

//Group volume (Rk) and surface area (Qk) parameters
    of the subgroup are
k_CH3 = 1;
k_CH2 = 2;
k_CH3CO = 19;
Rk_CH3 = 0.9011;
Rk_CH2 = 0.6744;
Rk_CH3CO = 1.6724;
Qk_CH3 = 0.848;
```

```
Qk_CH2 = 0.540;
Qk_CH3CO = 1.4880;

// Interaction parameters of the subgroups in kelvin
    (K) are
a_1_1 = 0;
a_1_2 = 0;
a_1_19 = 476.40;
a_2_1 = 0;
a_2_2 = 0;
a_2_19 = 476.40;
a_19_1 = 26.76;
a_19_2 = 26.76;
a_19_19 = 0;

r_1 = 1*Rk_CH3 + 1*Rk_CH3CO;
r_2 = 2*Rk_CH3 + 3*Rk_CH2;
q_1 = 1*Qk_CH3 + 1*Qk_CH3CO;
q_2 = 2*Qk_CH3 + 3*Qk_CH2;

J_1 = r_1/(r_1*x_1+r_2*x_2);
J_2 = r_2/(r_1*x_1+r_2*x_2);
L_1 = q_1/(q_1*x_1+q_2*x_2);
L_2 = q_2/(q_1*x_1+q_2*x_2);
t_1_1 = exp(-a_1_1/T);
t_1_2 = exp(-a_1_2/T);
t_1_19 = exp(-a_1_19/T);
t_2_1 = exp(-a_2_1/T);
t_2_2 = exp(-a_2_2/T);
t_2_19 = exp(-a_2_19/T);
t_19_1 = exp(-a_19_1/T);
t_19_2 = exp(-a_19_2/T);
t_19_19 = exp(-a_19_19/T);

e_1_1 = 1*Qk_CH3/q_1;
e_2_1 = 0;
e_19_1 = (1*Qk_CH3CO/q_1);
e_1_2 = 2*Qk_CH3/q_2;
```

```
e_2_2 = 3*Qk_CH2/q_2;
e_19_2 = 0;

B_1_1 = e_1_1*t_1_1 + e_2_1*t_2_1 + e_19_1*t_19_1;
B_1_2 = e_1_1*t_1_2 + e_2_1*t_2_2 + e_19_1*t_19_2;
B_1_19 = e_1_1*t_1_19 + e_2_1*t_2_19 + e_19_1*
   t_19_19;
B_2_1 = e_1_2*t_1_1 + e_2_2*t_2_1 + e_19_2*t_19_1;
B_2_2 = e_1_2*t_1_2 + e_2_2*t_2_2 + e_19_2*t_19_2;
B_2_19 = e_1_2*t_1_19 + e_2_2*t_2_19 + e_19_2*
   t_19_19;

theta_1 = (x_1*q_1*e_1_1 + x_2*q_2*e_1_2)/(x_1*q_1 +
    x_2*q_2);
theta_2 = (x_1*q_1*e_2_1 + x_2*q_2*e_2_2)/(x_1*q_1 +
    x_2*q_2);
theta_19 = (x_1*q_1*e_19_1 + x_2*q_2*e_19_2)/(x_1*
   q_1 + x_2*q_2);

s_1 = theta_1*t_1_1 + theta_2*t_2_1 + theta_19*
   t_19_1;
s_2 = theta_1*t_1_2 + theta_2*t_2_2 + theta_19*
   t_19_2;
s_19 = theta_1*t_1_19 + theta_2*t_2_19 + theta_19*
   t_19_19;

// log(Y1_C) = 1 - J_1 + log(J_1) - 5*q_1*(1- (J_1/
   L_1) + log(J_1/L_1))
// log(Y2_C) = 1 - J_2 + log(J_2) - 5*q_2*(1- (J_2/
   L_2) + log(J_2/L_2))
Y1_C = exp(1 - J_1 + log(J_1) - 5*q_1*(1- (J_1/L_1)
   + log(J_1/L_1)));
Y2_C = exp(1 - J_2 + log(J_2) - 5*q_2*(1- (J_2/L_2)
   + log(J_2/L_2)));

// For species 1
summation_theta_k_1 = theta_1*(B_1_1/s_1) + theta_2
   *(B_1_2/s_2) + theta_19*(B_1_19/s_19);
```

```
summation_e_ki_1 = e_1_1*log(B_1_1/s_1) + e_2_1*log(
    B_1_2/s_2) + e_19_1*log(B_1_19/s_19);

// For species 2
summation_theta_k_2 = theta_1*(B_2_1/s_1) + theta_2
    *(B_2_2/s_2) + theta_19*(B_2_19/s_19);
summation_e_ki_2 = e_1_2*log(B_2_1/s_1) + e_2_2*log(
    B_2_2/s_2) + e_19_2*log(B_2_19/s_19);

// log(Y1_R) = q_1(1 - summation_theta_k_1 +
    summation_e_ki_1)
// log(Y2_R) = q_2(1 - summation_theta_k_2 +
    summation_e_ki_2)
Y1_R = exp(q_1*(1 - summation_theta_k_1 +
    summation_e_ki_1));
Y2_R = exp(q_2*(1 - summation_theta_k_2 +
    summation_e_ki_2));

// log(Y1) = log(Y1_C) + log(Y1_R)
// log(Y2) = log(Y2_C) + log(Y2_R)
Y1 = exp(log(Y1_C) + log(Y1_R));
Y2 = exp(log(Y2_C) + log(Y2_R));

printf(" The activity coefficients are Y1 = %f  and
     Y2 = %f\n",Y1,Y2);
```

Scilab code Exa 14.13 Calculation of the value of activity coefficients

```
clear;
clc;

//Example - 14.15
//Page number - 481
printf("Example - 14.15 and Page number - 481\n\n")

```

```
//Given,
T = 25 + 273.15;//[K] - Temperature
R = 1.987;//[cal/mol*K] - Universal gas constant
//component 1 = chloroform
//component 2 = carbon tetrachloride
x_1 = 0.5;//Mole fraction of component 1 //Equimolar
     mixture
x_2 = 0.5;//Mole fraction of component 2
V_mol_1 = 81;//[cm^(3)/mol] - Molar volume of pure
    component 1
V_mol_2 = 97;//[cm^(3)/mol] - Molar volume of pure
    component 2
del_1 = 9.2;//[(cal/cm^(3))^(1/2)] - Mole fraction
    of component 1
del_2 = 8.6;//[(cal/cm^(3))^(1/2)] - Mole fraction
    of component 2

//Scatchard - Hilderbrand model
phi_1 = (x_1*V_mol_1)/(x_1*V_mol_1+x_2*V_mol_2);
phi_2 = (x_2*V_mol_2)/(x_1*V_mol_1+x_2*V_mol_2);

//log(Y1) = (V_mol_1/(R*T))*phi_1^(2)*(del_1-del_2)
    ^(2)
Y1 = exp((V_mol_1/(R*T))*(phi_1^(2))*((del_1-del_2)
    ^(2)));

//Similarly, for Y2
Y2 = exp((V_mol_2/(R*T))*(phi_2^(2))*((del_1-del_2)
    ^(2)));

printf("The value of activity coefficients for
    Scatchard-Hilderbrand model are\n Y1 = %f \t and
    \t Y2 = %f\n\n",Y1,Y2);
```

Scilab code Exa 14.14 Calculation of the value of activity coefficients

```
clear;
clc;

//Example - 14.14
//Page number - 485
printf("Example - 14.14 and Page number - 485\n\n")

//Given,
T = 25 + 273.15;//[K] - Temperature
mol_HCl = 0.001;//[mol/kg] - Molality of HCl
A = 0.510;//[(kg/mol)^(1/2)]
Z_positive = 1;//Stoichiometric coefficient of 'H'
   ion
Z_negative = -1;//Stoichiometric coefficient of 'Cl'
    ion
m_H_positive = mol_HCl;//
m_Cl_negative = mol_HCl;

// I = 1/2*[((Z_positive)^(2))*m_H_positive + ((
   Z_negative)^(2))*m_Cl_negative]
I = 1/2*(((Z_positive)^(2))*m_H_positive + ((
   Z_negative)^(2))*m_Cl_negative);

//Using Debye-Huckel limiting law wee get,
// log(Y1) = -A*(abs(Z_positive*Z_negative))*(I
   ^(1/2)))
Y = 10^(-A*(abs(Z_positive*Z_negative))*(I^(1/2)));
printf("The mean activity coefficient at 25 C using
   Debye-Huckel limiting law is Y = %f\n\n",Y);

//Using Debye-Huckel extended model we get
//log(Y_prime) = (-A*(abs(Z_positive*Z_negative))*(I
   ^(1/2)))/(1 + (I^(1/2)));
Y_prime = 10^((-A*(abs(Z_positive*Z_negative))*(I
   ^(1/2)))/(1 + (I^(1/2))));
printf("The mean activity coefficient at 25 C using
   Debye-Huckel extended model is Y = %f\n\n",
   Y_prime);
```

Scilab code Exa 14.15 Calculation of the value of activity coefficients

```
clear;
clc;

//Example - 14.15
//Page number - 485
printf("Example - 14.15 and Page number - 485\n\n")

//Given,
T = 25 + 273.15;//[K] - Temperature
mol_CaCl2 = 0.001;//[mol/kg] - Molality of HCl
A = 0.510;//[(kg/mol)^(1/2)]
Z_positive = 2;//Stoichiometric coefficient of 'Ca'
    ion
Z_negative = -1;//Stoichiometric coefficient of 'Cl'
     ion
m_Ca_positive = mol_CaCl2;
m_Cl_negative = 2*mol_CaCl2;

// I = 1/2*[((Z_positive)^(2))*m_Ca_positive + ((
    Z_negative)^(2))*m_Cl_negative]
I = 1/2*(((Z_positive)^(2))*m_Ca_positive + ((
    Z_negative)^(2))*m_Cl_negative);

//Using Debye-Huckel limiting law wee get,
// log(Y1) = -A*(abs(Z_positive*Z_negative))*(I
    ^(1/2))
Y = 10^(-A*(abs(Z_positive*Z_negative))*(I^(1/2)));
printf("The mean activity coefficient at 25 C using
    Debye-Huckel limiting law is Y = %f\n\n",Y);
```

Scilab code Exa 14.16 Proving a mathematical relation

```
clear;
clc;

//Example - 14.16
//Page number - 486
printf("Example - 14.16 and Page number - 486\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 14.16 on page
   number 486 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.16 on
   page number 486 of the book.")
```

Scilab code Exa 14.17 Calculation of pressure

```
clear;
clc;

//Example - 14.17
//Page number - 488
printf("Example - 14.17 and Page number - 488\n\n");

//Given,
T = 50 + 273.15;//[K] - Temperature
R=8.314;//[J/mol*K] - Universal gas constant
x_1 = 0.3;// Mole fraction of component 1
x_2 = (1-x_1);// Mole fraction of component 2
//Increment of 1% means Y2 = 1.01*Y1

```

```
//Excess volume of the mixture is given by,
V_excess = 4*x_1*x_2;//[cm^(3)/mol]
//Amd threfore
V_1_excess = 4*x_2*x_2*10^(-6);//[m^(3)/mol] - Exces
    volume of component 1
V_2_excess = 4*x_1*x_1*10^(-6);//[m^(3)/mol] - Exces
    volume of component 2

//We have from equation 14.89 of the book,
//V_i_excess/(R*T) = (del_log(Y_i)/del_P)_T,x

//Rearranging above equation
//d(log(Y1)) = (V1_excess/(R*T))dP
//Integrating the above equation at constant 'T' and
    'x' in the limit from 'Y1' to '1.01*Y1' in the
   LHS and from 'P' to 'P+delta_P ' in the RHS
//On simplification we get
//log(1.01*Y1/Y1) = (V_1_exces/(R*T))*delta_P
delta_P = log(1.01)/(V_1_excess/(R*T));//[N/m^(2)]
delta_P = delta_P*10^(-6);//[MPa]

printf("The required pressure to increase the
   activity coefficient by 1%% is %f MPa",delta_P);
```

Scilab code Exa 14.18 Proving a mathematical relation

```
clear;
clc;

//Example - 14.18
//Page number - 488
printf("Example - 14.18 and Page number - 488\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
```

```
//For prove refer to this example 14.18 on page
   number 488 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 14.18 on
   page number 488 of the book.")
```

Scilab code Exa 14.19 Determination of enthalpy

```
clear;
clc;

//Example - 14.19
//Page number - 489
printf("Example - 14.19 and Page number - 489\n\n");

//given
P = 2;//[bar] - Pressure
T = 310;//[K] - Temperature
R=8.314;//[J/mol*K] - Universal gas constant
A = (0.1665 + 233.74/T);// Margules parameter
B = (0.5908 + 197.55/T);// Margules parameter

//two parameter Margules equation is given by
//G_excess/(R*T*x_1*x_2) = B*x_1 + A*x_2
//On simplification and putting the values we get
//G_excess = ((0.5908 + 197.55/T)*x_1^(2)*x_2 +
   (0.1665 + 233.74/T)*x_2^(2)*x_1)


//H_excess/(R*T^(2)) = -[d/dT(G_excess/(R*T*x_1*x_2)
   )]_P,x
//On simplification and putting the values we get
//H_excess/(R*T^(2)) = (197.55/T^(2))*x_1^(2)*x_2 +
    (233.74/T^(2))*x_1*x_2^(2)

//We know that enthalpy change of mixing is given by
// delta_H_mix = H - x_1*H_1 - x_2*H_2 =
    delta_H_id_mix + H_excess

//But, delta_H_id_mix = 0 and H_excess is positive ,
     therefore enthalpy of muxture
// H > (x_1*H_1 + x_2*H_2)
//Therefore heat is required during the formation of
     mixture

printf("Since enthalpy of mixture formation (H)
    comes out to be positive , threfore steam is
    required to maintain the constant temperature.");
```

Scilab code Exa 14.20 Determination of an expression

```
clear;
clc;

//Example - 14.20
//Page number - 490
printf("Example - 14.20 and Page number - 490\n\n");

T = 40 + 273.15;//[K]
P = 101.3;//[kPa]

// G_E/(R*T) = A*x_1*x_2

// The parameter A at 101.3 kPa and various
    temperatures are
A_35 = 0.479;// A at 35 C
A_40 = 0.458;// A at 40 C
A_45 = 0.439;// A at 45 C
```

```

// At 40 C, G_E/(R*T) is given by
// G_E/(R*T) = A_40*x1*x2
// Therefore log(Y1) = A_40*x2^(2) and log(Y2) =
   A_40*x1^(2)

dA_dT = (A_45-A_35)/(45-35);//[K^(-1)] - dA/dT
// H_E/(R*T^(2)) = -[del(G_E/(R*T))/del(T)]_P,x = (
   dA/dT)*x1*x2
// H_E/(R*T) = -T*(dA/dT)*x1*x2 = 1.25*x1*x2

// S_E = (H_E - G_E)/T = (-R*T^(2)*(dA/dT)*x1*x2 - A
   *R*T*x1*x2)/T = -(R*T*(dA/dT) + A*R)*x1*x2
// Thus S_E/R = -(T*(dA/dT) + A)*x1*x2 = 0.795*x1*x2

printf(" The expressions are  H_E/(R*T) = 1.25*x1*x2
    \n\t\t    S_E/R = 0.795*x1*x2");
```

Scilab code Exa 14.21 Calculation of enthalpy entropy and Gibbs free energy

```
clear;
clc;

//Example - 14.21
//Page number - 490
printf("Example - 14.21 and Page number - 490\n\n");

//given
T = 293.15;//[K] - Temperature
R=8.314;//[J/mol*K] - Universal gas constant
A = 1280;//[J/mol]

//(dA/dT)_P,x = del_A (say)
dA_dT = -7.0;//[J/mol-K]
```

```

//For equilomar mixture,
x_1 = 0.5;// Mole fraction of component 1
x_2 = 0.5;// Mole fraction of component 2

//log(Y1) =  (A/(R*T))*x_2^(2)
//log(Y2) =  (A/(R*T))*x_1^(2)
Y1 = exp((A/(R*T))*x_2^(2));
Y2 = exp((A/(R*T))*x_1^(2));

//G_excess/(R*T*) = x_1*log(Y1) + x_2*log(Y2) = (A/(
   R*T))*x_1*x_2
G_excess = A*x_1*x_2;//[J/mol] - Excess Gibbs free
    energy

//H_excess/(R*T^(2)) = -[d/dT(G_excess/(R*T))]_P,x
//H_excess/(R*T^(2)) = -((x_1*x_2)/R)*[d/dT(A/T)]_P,
    x
//On simplification  and putting the values we get
H_excess = A*x_1*x_2 - T*dA_dT*x_1*x_2;//[J/mol] -
    Excess enthalpy

//Now excess entropy is given by
S_excess = (H_excess - G_excess)/T;//[J/mol-K] -
    Excess entropy

printf("For equimolar mixture\n\n");
printf("Excess Gibbs free energy (G_excess) is %f J/
    mol\n\n",G_excess);
printf("Excess enthalpy (H_excess) is %f J/mol\n\n",
    H_excess);
printf("Excess entropy (S_excess) is %f J/mol\n\n",
    S_excess);
printf("The value of activity coefficient Y1 is %f\n
    \n",Y1);
printf("The value of activity coefficient Y2 is %f\n
    \n",Y2);
```

Scilab code Exa 14.22 Determination of Gibbs free energy and enthalpy change

```
clear;
clc;

// Example - 14.22
// Page number - 491
printf("Example - 14.22 and Page number - 491\n\n");

// Given
T = 60 + 273.15;//[K] - Temperature
R = 8.314;//[J/mol*K] - Universal gas constant

// log(Y1_inf) = log(Y2_inf) = 0.15 + 10/T

// Since the two liquids are slightly dissimilar ,
   we assume the activity coeffiecients to follow
   van Laar equation
// From van Laaar equation
// A = log(Y1_inf) and B = log(Y2_inf) and since it
   is given that log(Y1_inf) = log(Y2_inf),
   therefore A = B
//(x_1*x_2)/(G_excess/R*T) = x_1/B + x_2/A = X_1/A +
    x_2/A = 1/A
// G_excess/(R*T) = A*x_1*x_2

// For equilomar mixture,
x_1 = 0.5;// Mole fraction of component 1
x_2 = 0.5;// Mole fraction of component 2

// Expression for A can be written as
// A = 0.15 + 10/T, where T is in C. Therefore
A = 0.15 + 10/(T - 273.15);
```

```
// Differentiating it with respect to temprature we
   get
dA_dT = - 10/((T-273.15)^(2));

// The excess Gibbs free energy can be calculated as
G_excess = A*x_1*x_2*(R*T);//[J/mol]

// The ideal Gibbs free energy change can  be
   calculated as
delta_G_id_mix = R*T*(x_1*log(x_1) + x_2*log(x_2));
   //[J/mol]

// Finally we have,
delta_G_mix = G_excess + delta_G_id_mix;//[J/mol]

printf("The Gibbs free energy change of mixing for
   equimolar mixture is %f J/mol\n\n",delta_G_mix);


// Now let us determine the excess enthalpy. We know
    that
// H_excess/(R*T^(2)) = -[d/dT(G_excess/R*T)]_P,x =
  - x_1*x_2*[d/dT(A)]_P,x
// Therefore at 'T' = 60 C the excess enthalpy is
   given by
H_excess = -R*(T^(2))*x_1*x_2*dA_dT;//[J/mol]

delta_H_id_mix = 0;//[J/mol] - Enthalpy change of
   mixing for ideal solution is zero.

//Thus enthalpy change of mixing for an equimolar
   mixture at 333.15 K is given by
delta_H_mix = delta_H_id_mix + H_excess;//[J/mol]


printf("The enthalpy change of mixing for equimolar
   mixture is %f J/mol",delta_H_mix);
```

Chapter 15

Vapour Liquid Equilibria

Scilab code Exa 15.1 Calculation of number of moles in liquid and vapour phase

```
clear;
clc;

//Example - 15.1
//Page number - 503
printf("Example - 15.1 and Page number - 503\n\n");

//Given
T = 90+ 273.15;//[K] - Temperature
P = 1;//[atm] - Pressure
x_1 = 0.5748;// Equilibrium composition of liquid
    phase
y_1 = 0.7725;// Equilibrium composition of vapour
    phase

// We start with 1 mol of mixture of composition z_1
    = 0.6, from marterial balance we get
// (L + V)*z_1 = L*x_1 + V*y_1
// Since total number of moles is 1, we get
// z_1 = L*x_1 + (1-L)*y_1
```

```

z_1_1 = 0.6;// - Mole fraction of benzene
L_1 = (z_1_1 - y_1)/(x_1 - y_1);
V_1 = 1 - L_1;

printf(" For z_1 = 0.6\n");
printf(" The moles in the liquid phase is %f mol\n",
   L_1);
printf(" The moles in the vapour phase is %f mol\n\n
   ",V_1);

z_1_2 = 0.7;// - Mole fraction of benzene
L_2 = (z_1_2 - y_1)/(x_1 - y_1);
V_2 = 1 - L_2;

printf(" For z_1 = 0.7\n");
printf(" The moles in the liquid phase is %f mol\n",
   L_2);
printf(" The moles in the vapour phase is %f mol\n\n
   ",V_2);


// For z = 0.8
// The feed remains vapour and the liquid is not
   formed at all as it is outside the two-phase
   region (x_1 = 0.5748 and y_1 = 0.7725).
printf(" For z_1 = 0.8\n");
printf(" The feed remains vapour and the liquid is
   not formed at all as it is outside the two-phase
   region (x_1 = 0.5748 and y_1 = 0.7725)")
```

Scilab code Exa 15.2 Calculation of pressure temperature and composition

```
clear;
```

```
clc;
funcprot(0);

//Example - 15.2
//Page number - 515
printf("Example - 15.2 and Page number - 515\n\n");

//Given
// log(P_1_sat) = 14.39155 - 2795.817/(t + 230.002)
// log(P_2_sat) = 16.26205 - 3799.887/(t + 226.346)

//(a)
x_1_a =0.43;// Equilibrium composition of liquid
    phase
t_a = 76;//[C] - Temperature
x_2_a = 1 - x_1_a;

// Since liquid phase composition is given we use
    the relation
// P = x_1*P_1_sat + x_2*P_2_sat
// At t = 76 C
P_1_sat_a = exp(14.39155 - 2795.817/(t_a + 230.002))
    ;
P_2_sat_a = exp(16.26205 - 3799.887/(t_a + 226.346))
    ;
// Therefore total pressure is
P_a = x_1_a*P_1_sat_a + x_2_a*P_2_sat_a;//[kPa]
y_1_a = (x_1_a*P_1_sat_a)/(P_a);
y_2_a = (x_2_a*P_2_sat_a)/(P_a);

printf("(a).The system pressure is, P = %f kPa\n",
    P_a);
printf("    The vapour phase composition is, y_1 =
    %f\n\n",y_1_a);

//(b)
y_1_b = 0.43;// Equilibrium composition of vapour
    phase
```

```
y_2_b = 1 - y_1_b;
t_b = 76;//[C] - Temperature

P_1_sat_b = exp(14.39155 - 2795.817/(t_b + 230.002))
    ;
P_2_sat_b = exp(16.26205 - 3799.887/(t_b + 226.346))
    ;

// Since vapour phase composition is given ,the
    system pressure is given by
// 1/P = y_1/P_1_sat + y_2/P_2_sat
P_b = 1/(y_1_b/P_1_sat_b + y_2_b/P_2_sat_b);

x_1_b = (y_1_b*P_b)/P_1_sat_b;
x_2_b = (y_2_b*P_b)/P_2_sat_b;

printf("(b).The system pressure is, P = %f kPa\n",
    P_b);
printf("     The liquid phase composition is, x_1 =
    %f\n\n",x_1_b);

//(c)
x_1_c = 0.32;// Equilibrium composition of liquid
    phase
x_2_c = 1 - x_1_c;
P_c = 101.33;//[kPa] - Pressure of the system

// We have,  P = x_1*P_1_sat + x_2*P_2_sat
t_1_sat = 2795.817/(14.39155 - log(P_c)) - 230.002;
t_2_sat = 3799.887/(16.26205 - log(P_c)) - 226.346;
t = x_1_c*t_1_sat + x_2_c*t_2_sat;

error = 10;
while(error>0.1)
    P_1_sat = exp(14.39155 - 2795.817/(t + 230.002))
        ;
    P_2_sat = exp(16.26205 - 3799.887/(t + 226.346))
        ;
    P = x_1_c*P_1_sat + x_2_c*P_2_sat;
    error=abs(P - P_c);
    t = t - 0.1;
end

P_1_sat_c = exp(14.39155 - 2795.817/(t + 230.002));
P_2_sat_c = exp(16.26205 - 3799.887/(t + 226.346));

y_1_c = (x_1_c*P_1_sat_c)/(P_c);
y_2_c = 1 - y_1_c;

printf("(c).The system temperature is, t = %f C\n",t
   );
printf("     The vapour phase composition is, y_1 =
   %f\n\n",y_1_c);

//(d)
y_1_d = 0.57;// Vapour phase composition
y_2_d = 1 - y_1_d;
P_d = 101.33;//[kPa] - Pressure of the system

// Since vapour phase composition is given, we can
   use the relation
// 1/P = y_1/P_1_sat + y_2/P_2_sat
t_1_sat_d = 2795.817/(14.39155 - log(P_d)) -
   230.002;
t_2_sat_d = 3799.887/(16.26205 - log(P_d)) -
   226.346;
t_d = y_1_d*t_1_sat_d + y_2_d*t_2_sat_d;

fault = 10;
while(fault>0.1)
    P_1_sat_prime = exp(14.39155 - 2795.817/(t_d +
       230.002));
    P_2_sat_prime = exp(16.26205 - 3799.887/(t_d +
       226.346));
    P_prime = 1/(y_1_d/P_1_sat_prime + y_2_d/
       P_2_sat_prime);
    fault=abs(P_prime - P_d);
    t_d = t_d + 0.01;
end

P_1_sat_d = exp(14.39155 - 2795.817/(t_d + 230.002))
    ;
P_2_sat_d = exp(16.26205 - 3799.887/(t_d + 226.346))
    ;

x_1_d = (y_1_d*P_d)/(P_1_sat_d);
x_2_d = 1 - x_1_d;

printf("(d).The system temperature is, t = %f C\n",
    t_d);
printf("     The liquid phase composition is, x_1 =
    %f\n\n",x_1_d);
```

Scilab code Exa 15.3 Calculation of pressure temperature and composition

```
clear;
clc;
funcprot(0);

//Example - 15.3
//Page number - 516
printf("Example - 15.3 and Page number - 516\n\n");

//Given
// log(P_1_sat) = 14.3916 - 2795.82/(t + 230.00)
// log(P_2_sat) = 14.2724 - 2945.47/(t + 224.00)
// log(P_3_sat) = 14.2043 - 2972.64/(t + 209.00)

//(a)
x_1_a = 0.25;// Equilibrium composition of liquid
```

```
    phase
x_2_a = 0.35;
x_3_a = 1 - x_1_a - x_2_a;
t_a = 80;//[C] - Temperature

// At t = 80 C
P_1_sat_a = exp(14.3916 - 2795.82/(t_a + 230.00));
P_2_sat_a = exp(14.2724 - 2945.47/(t_a + 224.00));
P_3_sat_a = exp(14.2043 - 2972.64/(t_a + 209.00));

// Since liquid phase composition is given we use
    the relation
P_a = x_1_a*P_1_sat_a + x_2_a*P_2_sat_a + x_3_a*
    P_3_sat_a;//[kPa]

y_1_a = (x_1_a*P_1_sat_a)/(P_a);
y_2_a = (x_2_a*P_2_sat_a)/(P_a);
y_3_a = (x_3_a*P_3_sat_a)/(P_a);

printf("(a).The system pressure is, P = %f kPa\n",
    P_a);
printf("     The vapour phase composition is given by
    , y_1 = %f, y_2 = %f and y_3 = %f \n\n",y_1_a,
    y_2_a,y_3_a);

//(2)
y_1_b = 0.50;// Equilibrium composition of liquid
    phase
y_2_b = 0.30;
y_3_b = 1 - y_1_a - y_2_a;
t_b = 85;//[C] - Temperature

// At t = 80 C
P_1_sat_b = exp(14.3916 - 2795.82/(t_b + 230.00));
P_2_sat_b = exp(14.2724 - 2945.47/(t_b + 224.00));
P_3_sat_b = exp(14.2043 - 2972.64/(t_b + 209.00));

// Since vapour phase composition is given we use
```

```
    the relation
P_b = 1/(y_1_b/P_1_sat_b + y_2_b/P_2_sat_b + y_3_b/
    P_3_sat_b);//[kPa]

// Therefore we have
x_1_b = (y_1_b*P_b)/P_1_sat_b;
x_2_b = (y_2_b*P_b)/P_2_sat_b;
x_3_b = (y_3_b*P_b)/P_3_sat_b;

printf("(b).The system pressure is, P = %f kPa\n",
    P_b);
printf("     The liquid phase composition is given by
    , x_1 = %f, x_2 = %f and x_3 = %f \n\n",x_1_b,
    x_2_b,x_3_b);

//(c)
x_1_c = 0.30;// Equilibrium composition of liquid
    phase
x_2_c = 0.45;
x_3_c = 1 - x_1_c - x_2_c;
P_c = 80;//[kPa] - Pressure of the system

// We have,  P = x_1*P_1_sat + x_2*P_2_sat + x_3*
    P_3_sat
t_1_sat = 2795.82/(14.3916 - log(P_c)) - 230.00;//[C
    ]
t_2_sat = 2945.47/(14.2724 - log(P_c)) - 224.00;//[C
    ]
t_3_sat = 2972.64/(14.2043 - log(P_c)) - 209.00;//[C
    ]
t = x_1_c*t_1_sat + x_2_c*t_2_sat + x_3_c*t_3_sat;

error = 10;
while(error>0.5)
    P_1_sat = exp(14.3916 - 2795.82/(t + 230.00));
    P_2_sat = exp(14.2724 - 2945.47/(t + 224.00));
    P_3_sat = exp(14.2043 - 2972.64/(t + 209.00));
    P = x_1_c*P_1_sat + x_2_c*P_2_sat + x_3_c*
```

```
        P_3_sat;
    error=abs(P - P_c);
    t = t - 0.2;
end

P_1_sat_c = exp(14.3916 - 2795.82/(t + 230.00));
P_2_sat_c = exp(14.2724 - 2945.47/(t + 224.00));
P_3_sat_c = exp(14.2043 - 2972.64/(t + 209.00));

y_1_c = (x_1_c*P_1_sat_c)/(P_c);
y_2_c = (x_2_c*P_2_sat_c)/(P_c);
y_3_c = 1 - y_1_c - y_2_c;

printf("(c).The system temperature is, t = %f C\n",t
   );
printf("     The vapour phase composition is, y_1 =
   %f, y_2 %f and y_3 = %f\n\n",y_1_c,y_2_c,y_3_c);

//(d)
y_1_d = 0.6;// Vapour phase composition
y_2_d = 0.2;
y_3_d = 1 - y_1_d - y_2_d;
P_d = 90;//[kPa] - Pressure of the system

// Since vapour phase composition is given, we can
   use the relation
// 1/P = y_1/P_1_sat + y_2/P_2_sat + y_3/P_3_sat
t_1_sat_d = 2795.82/(14.3916 - log(P_d)) - 230.00;
t_2_sat_d = 2945.47/(14.2724 - log(P_d)) - 224.00;
t_3_sat_d = 2972.64/(14.2043 - log(P_d)) - 209.00;
t_d = y_1_d*t_1_sat_d + y_2_d*t_2_sat_d + y_3_d*
   t_3_sat_d;

fault = 10;
while(fault>0.5)
    P_1_sat_prime = exp(14.3916 - 2795.82/(t_d +
        230.00));
    P_2_sat_prime = exp(14.2724 - 2945.47/(t_d +
        224.00));
    P_3_sat_prime = exp(14.2043 - 2972.64/(t_d +
        209.00));
    P_prime = 1/(y_1_d/P_1_sat_prime + y_2_d/
        P_2_sat_prime + y_3_d/P_3_sat_prime);
    fault=abs(P_prime - P_d);
    t_d = t_d + 0.2;
end

P_1_sat_d = exp(14.3916 - 2795.82/(t_d + 230.00));
P_2_sat_d = exp(14.2724 - 2945.47/(t_d + 224.00));
P_3_sat_d = exp(14.2043 - 2972.64/(t_d + 209.00));

x_1_d = (y_1_d*P_d)/(P_1_sat_d);
x_2_d = (y_2_d*P_d)/(P_2_sat_d);
x_3_d = 1 - x_1_d - x_2_d;

printf("(d).The system temperature is, t = %f C\n",
   t_d);
printf("    The liquid phase composition is, x_1 =
   %f, x_2 = %f and x_3 = %f\n\n",x_1_d,x_2_d,x_3_d)
   ;
```

Scilab code Exa 15.4 Calculation of pressure and composition

```
clear;
clc;

//Example - 15.4
//Page number - 519
printf("Example - 15.4 and Page number - 519\n\n");

//Given
T = 120;//[C] - Temperature
P_1 = 1;//[atm] - Initial pressure
```

```
P_1 = P_1*101.325;//[kPa]
R = 8.314;//[J/mol*K] - Universal gas constant

y_1 = 0.3;// Mole fraction of propane
y_2 = 0.5;// Mole fraction of butane
y_3 = 0.2;// Mole fraction of hexane

// log(P_1_sat) = 13.7713 - 1892.47/(t + 248.82)
// log(P_2_sat) = 13.7224 - 2151.63/(t + 236.91)
// log(P_3_sat) = 13.8216 - 2697.55/(t + 224.37)

//(a)
P_1_sat = exp(13.7713 - 1892.47/(T + 248.82));
P_2_sat = exp(13.7224 - 2151.63/(T + 236.91));
P_3_sat = exp(13.8216 - 2697.55/(T + 224.37));

// Since vapour phase composition is given we can
   use the relation ,
P_2 = 1/(y_1/P_1_sat + y_2/P_2_sat + y_3/P_3_sat);//
   [kPa]

printf(" (a).The pressure of the mixture when first
   drop condenses is given by, P = %f kPa\n\n",P_2);

//(b)
x_1 = (y_1*P_2)/P_1_sat;
x_2 = (y_2*P_2)/P_2_sat;
x_3 = (y_3*P_2)/P_3_sat;

printf(" (b).The liquid phase composition is given
   by, x_1 (propane) = %f, x_2 (butane) = %f and x_3
    (hexane) = %f \n\n",x_1,x_2,x_3);

// (c)
// Work transfer per mol is given by
// W = integral(P*dV) = integral((R*T/V)*dV) = R*T*
   log(V_2/V_1) = R*T*log(P_1/P_2)
w = R*(T+273.15)*log(P_1/P_2);//[J/mol]
```

```
// For 100 mol
W = w*100;//[J]
W = W*10^(-3);//[kJ]
printf(" (c).The work required for 100 mol of
   mixture handeled is %f kJ",W);
```

Scilab code Exa 15.5 Calculation of pressure temperature and composition

```
clear;
clc;
funcprot(0);

//Example - 15.5
//Page number - 520
printf("Example - 15.5 and Page number - 520\n\n");

//Given
T = 27;//[C] - Temperature

// log(P_1_sat) = 13.8216 - 2697.55/(t + 224.37)
// log(P_2_sat) = 13.8587 - 2911.32/(t + 216.64)

//(a)
x_1_a = 0.2;
x_2_a = 1 - x_1_a;

// At t = 27 C
P_1_sat = exp(13.8216 - 2697.55/(T + 224.37));//[kPa
   ]
P_2_sat = exp(13.8587 - 2911.32/(T + 216.64));//[kPa
   ]
P_a = x_1_a*P_1_sat + x_2_a*P_2_sat;//[kPa]

y_1_a = x_1_a*P_1_sat/P_a;
```

```
y_2_a = x_2_a*P_2_sat/P_a;

printf("(a).The total pressure is, P = %f kPa\n",P_a
    );
printf("     The vapour phase composition is given by
    , y_1 = %f and y_2 = %f\n\n",y_1_a,y_2_a);

//(b)
y_1_b = 0.2;
y_2_b = 1 - y_1_b;
// Since vapour phase composition is given we can
    use the relation
P_b = 1/(y_1_b/P_1_sat + y_2_b/P_2_sat);//[kPa]

// Therefore we have
x_1_b = (y_1_b*P_b)/P_1_sat;
x_2_b = (y_2_b*P_b)/P_2_sat;

printf("(b).The total pressure is, P = %f kPa\n",P_b
    );
printf("     The liquid phase composition is given by
    , x_1 = %f and x_2 = %f\n\n",x_1_b,x_2_b);

//(c)
P_c = 30;//[kPa] - Total pressure
x_1_c = 0.2;
x_2_c = 1 - x_1_c;

// We have, P = x_1*P_1_sat + x_2*P_2_sat
t_1_sat = 2697.55/(13.8216 - log(P_c)) - 224.37;
t_2_sat = 2911.32/(13.8587 - log(P_c)) - 216.64;
t = x_1_c*t_1_sat + x_2_c*t_2_sat;

fault = 10;
while(fault>0.3)
    P_1_sat = exp(13.8216 - 2697.55/(t + 224.37));
    P_2_sat = exp(13.8587 - 2911.32/(t + 216.64));
    P = x_1_c*P_1_sat + x_2_c*P_2_sat;
    fault = abs(P - P_c);
    t = t - 0.1;
end

P_1_sat_c = exp(13.8216 - 2697.55/(t + 224.37));
P_2_sat_c = exp(13.8587 - 2911.32/(t + 216.64));

y_1_c = (x_1_c*P_1_sat_c)/(P_c);
y_2_c = 1 - y_1_c;

printf("(c).The system temperature is, t = %f C\n",t
   );
printf("    The vapour phase composition is, y_1 =
   %f and y_2 = %f \n\n",y_1_c,y_2_c);
```

Scilab code Exa 15.6 Determinatin of DPT and BPT

```
clear;
clc;
funcprot(0);

//Example - 15.6
//Page number - 521
printf("Example - 15.6 and Page number - 521\n\n");

//Given
P = 90;//[kPa] - Pressure
R = 8.314;//[J/mol*K] - Universal gas constant

// log(P_sat) = A - B/(t + C)

// For benzene
A_1 = 13.8594;
B_1 = 2773.78;
C_1 = 220.07;
```

```
// For ethyl benzene
A_2 = 14.0045;
B_2 = 3279.47;
C_2 = 213.20;

x_1 = 0.5;// Equimolar mixture
x_2 = 0.5;

// The bubble point temperature equation is
// P = x_1*P_1_sat + x_2*P_2_sat

t_1_sat = B_1/(A_1 - log(P)) - C_1;
t_2_sat = B_2/(A_2 - log(P)) - C_2;
t = x_1*t_1_sat + x_2*t_2_sat;

fault = 10;
while(fault>0.3)
    P_1_sat = exp(A_1 - B_1/(t + C_1));
    P_2_sat = exp(A_2 - B_2/(t + C_2));
    P_net = x_1*P_1_sat + x_2*P_2_sat;
    fault=abs(P_net - P);
    t = t - 0.1;
end

printf(" The bubble point temperature is %f C\n\n",t
   );

// Now let us determine dew point temperature for
   y_1 = 0.5, and P = 90 kPa
y_1 = 0.5;// Equimolar mixture
y_2 = 0.5;

// 1/P = y_1/P_1_sat + y_2/P_2_sat
// Let us statrt with t = 104.07
t_old = 104.07;
error = 10;
while(error>0.3)
    P_1_sat_prime = exp(A_1 - B_1/(t_old + C_1));
    P_2_sat_prime = exp(A_2 - B_2/(t_old + C_2));
    P_net_prime = 1/(y_1/P_1_sat_prime + y_2/
       P_2_sat_prime);
    error=abs(P_net_prime - P);
    t_old = t_old + 0.1;
end

printf(" The dew point temperature is %f C",t_old);
```

Scilab code Exa 15.7 Determination of range of temperature for which two phase exists

```
clear;
clc;
funcprot(0);

//Example - 15.7
//Page number - 522
printf("Example - 15.7 and Page number - 522\n\n");

//Given
P = 1;//[bar] - Pressure
P = P*10^(2);//[kPa]

// log(P_1_sat) = 13.8594 - 2773.78/(t + 220.07)
// log(P_2_sat) = 14.0098 - 3103.01/(t + 219.79)

// The bubble point equation is
// P = x_1*P_1_sat + x_2*P_2_sat;

t_1_sat = 2773.78/(13.8594 - log(P)) - 220.07;
t_2_sat = 3103.01/(14.0098 - log(P)) - 219.79;

// For x_1 = 0.1
// t = x_1_1*t_1_sat + x_2_1*t_2_sat;
```

```
x_1 = [0.1,0.5,0.9];

for i=1:3
x_2(i) = 1 - x_1(i);
t = x_1(i)*t_1_sat + x_2(i)*t_2_sat;
fault = 10;
while(fault>0.3)
    P_1_sat = exp(13.8594 - 2773.78/(t + 220.07));
    P_2_sat = exp(14.0098 - 3103.01/(t + 219.79));
    P_net = x_1(i)*P_1_sat + x_2(i)*P_2_sat;
    fault=abs(P_net - P);
    t = t - 0.1;
end

printf(" The bubble point temperature (for x_1 = %f)
    is %f C\n",x_1(i),t);

end

printf("\n\n");

// Now let us determine dew point temperature
// 1/P = y_1/P_1_sat + y_2/P_2_sat

y_1 = [0.1,0.5,0.9];

for j=1:3
y_2(j) = 1 - y_1(j);
t_prime = y_1(j)*t_1_sat + y_2(j)*t_2_sat;

error = 10;
while(error>0.3)
    P_1_sat = exp(13.8594 - 2773.78/(t_prime +
        220.07));
    P_2_sat = exp(14.0098 - 3103.01/(t_prime +
        219.79));
    P_net = 1/(y_1(j)/P_1_sat + y_2(j)/P_2_sat);
    error=abs(P_net - P);
    t_prime = t_prime + 0.1;
end

printf(" The dew point temperature (for y_1 = %f) is
    %f C\n",y_1(j),t_prime);

end

printf("\n\n");

//Therefore
printf(" The temperature range for (z_1 = 0.1) which
    two phase exist is 105.61 C to 108.11 C\n");
printf(" The temperature range for (z_1 = 0.5) which
    two phase exist is 91.61 C to 98.40 C\n");
printf(" The temperature range for (z_1 = 0.9) which
    two phase exist is 81.71 C to 84.51 C\n");
```

Scilab code Exa 15.8 Calculation of DPT and BPT

```
clear;
clc;
funcprot(0);

//Example - 15.8
//Page number - 524
printf("Example - 15.8 and Page number - 524\n\n");

//Given
x_1 = 0.20;
x_2 = 0.45;
x_3 = 0.35;
P = 10;//[atm]
P = P*101325*10^(-3);//[kPa]

```

```
// log(P_1_sat) = 13.7713 - 1892.47/(t + 248.82)
// log(P_2_sat) = 13.7224 - 2151.63/(t + 236.91)
// log(P_3_sat) = 13.8183 - 2477.07/(t + 233.21)

//(a)
// The bubble point equation is
// P = x_1*P_1_sat + x_2*P_2_sat + x_3*P_3_sat

t_1_sat = 1892.47/(13.7713 - log(P)) - 248.82;
t_2_sat = 2151.63/(13.7224 - log(P)) - 236.91;
t_3_sat = 2477.07/(13.8183 - log(P)) - 233.21;
t = x_1*t_1_sat + x_2*t_2_sat + x_3*t_3_sat;

fault = 10;
while(fault>0.1)
    P_1_sat = exp(13.7713 - 1892.47/(t + 248.82));
    P_2_sat = exp(13.7224 - 2151.63/(t + 236.91));
    P_3_sat = exp(13.8183 - 2477.07/(t + 233.21));
    P_net = x_1*P_1_sat + x_2*P_2_sat + x_3*P_3_sat;
    fault=abs(P_net - P);
    t = t - 0.003;
end

BPT = t;
printf(" (a).The bubble point temperature is %f C\n\
   n",BPT);

// (b)
// Now let us determine dew point temperature for
   y_1 = 0.5, and P = 90 kPa
y_1 = 0.20;
y_2 = 0.45;
y_3 = 0.35;

// 1/P = y_1/P_1_sat + y_2/P_2_sat + y_3/P_3_sat

t_old = 90;//[C]
error = 10;
```

```
while(error>0.1)
    P_1_sat_prime = exp(13.7713 - 1892.47/(t_old +
       248.82));
    P_2_sat_prime = exp(13.7224 - 2151.63/(t_old +
       236.91));
    P_3_sat_prime = exp(13.8183 - 2477.07/(t_old +
       233.21));
    P_net_prime = 1/(y_1/P_1_sat_prime + y_2/
       P_2_sat_prime + y_3/P_3_sat_prime);
    error=abs(P_net_prime - P);
    t_old = t_old + 0.003;
end

DPT = t_old;
printf(" (b).The dew point temperature is %f C\n\n",
   DPT);

// (c)
// For the given composition and pressure the two
   phase region exists in  the temperature range of
    DPT and BPT
// Therefore at 82 C two phase exists
// At 82 C and P = 1013.25 kPa pressure
T_c = 82;//[C]
P_c = 1013.25;//[kPa]
z_1 = 0.20;
z_2 = 0.45;
z_3 = 0.35;

P_1_sat_c = exp(13.7713 - 1892.47/(T_c + 248.82));
P_2_sat_c = exp(13.7224 - 2151.63/(T_c + 236.91));
P_3_sat_c = exp(13.8183 - 2477.07/(T_c + 233.21));

K_1 = P_1_sat_c/P_c;
K_2 = P_2_sat_c/P_c;
K_3 = P_3_sat_c/P_c;

// We have to find such a V that the following
```

```
    equation is satisfied.
// summation(y_i) = K_i*z_i/(1-V+V*K_i) = 1
// K_1*z_1/(1-V+V*K_1) + K_2*z_2/(1-V+V*K_2) + K_3*
    z_3/(1-V+V*K_3) = 1

deff('[y]=f1(V)','y= K_1*z_1/(1-V+V*K_1) + K_2*z_2
    /(1-V+V*K_2) + K_3*z_3/(1-V+V*K_3)-1');
V = fsolve(0.4,f1);

// Therefore now we have
y_1_c = K_1*z_1/(1-V+V*K_1);
y_2_c = K_2*z_2/(1-V+V*K_2);
y_3_c = K_3*z_3/(1-V+V*K_3);
x_1_c = y_1_c/K_1;
x_2_c = y_2_c/K_2;
x_3_c = y_3_c/K_3;

printf(" (c).The proportion of vapour is given by, V
     = %f\n\n",V);
printf("      The composition of vapour foemed is
    given by, y_1 = %f, y_2 = %f and y_3 = %f \n\n",
    y_1_c,y_2_c,y_3_c);
printf("      The composition of liquid formed is
    given by, x_1 = %f, x_2 = %f and x_3 = %f \n\n",
    x_1_c,x_2_c,x_3_c);
```

Scilab code Exa 15.9 Calculation of range of pressure for which two phase exists

```
clear;
clc;

//Example - 15.9
//Page number - 526
printf("Example - 15.9 and Page number - 526\n\n");
```

```

//Given
T = 27;//[C] - Temperature
z_1 = 0.4;
z_2 = 0.3;
z_3 = 0.3;

// log(P_sat) = A - B/(t + C)

// For propane
A_1 = 13.7713;
B_1 = 1892.47;
C_1 = 248.82;
// For i-butane
A_2 = 13.4331;
B_2 = 1989.35;
C_2 = 236.84;
// For n-butane
A_3 = 13.7224;
B_3 = 2151.63;
C_3 = 236.91;

//(a)
// The pressure range for the existence of two phase
    region lies between dew point and bubble point
   pressures.
// At the dew point the whole feed lies in the
   vapour phase and a drop of liquid is formed,
   therefore
y_1 = z_1;
y_2 = z_2;
y_3 = z_3;

// At 27 C,
P_1_sat = exp(A_1 - B_1/(T + C_1));
P_2_sat = exp(A_2 - B_2/(T + C_2));
P_3_sat = exp(A_3 - B_3/(T + C_3));

```

```
// The dew point pressure is given by
P_1 = 1/(y_1/P_1_sat + y_2/P_2_sat + y_3/P_3_sat);

// At the bubble point the whole feed lies in the
   liquid phase and an infinitesimal amount of
   vapour is formed, therefore
x_1 = z_1;
x_2 = z_2;
x_3 = z_3;

// The bubble point pressure is given by
P_2 = x_1*P_1_sat + x_2*P_2_sat + x_3*P_3_sat;

printf(" (a).The two phase region exists between %f
   and %f kPa\n\n",P_1,P_2);

//(b)
// The mean of the two-phase pressure range is given
    by
P_mean = (P_1 + P_2)/2;

// Now let us calculate the K values of the
   components
K_1 = P_1_sat/P_mean;
K_2 = P_2_sat/P_mean;
K_3 = P_3_sat/P_mean;

// summation of y_i = 1, gives
// (K_1*z_1)/(1-V-K_1*V) + (K_2*z_2)/(1-V-K_2*V) + (
   K_3*z_3)/(1-V-K_3*V) = 1
// Solving we get
deff('[y]=f(V)','y=(K_1*z_1)/(1-V+K_1*V) + (K_2*z_2)
   /(1-V+K_2*V) + (K_3*z_3)/(1-V+K_3*V)-1');
V = fsolve(0.1,f);

y_1_prime = (z_1*K_1)/(1-V+K_1*V);

printf(" (b).The mole fraction of propane in vapour
   phase is %f whereas in the feed is %f and
   fraction of vapour in the system is %f",y_1_prime
   ,z_1,V);
```

Scilab code Exa 15.10 Determination of vapour and liquid phase composition

```
clear;
clc;

//Example - 15.10
//Page number - 527
printf("Example - 15.10 and Page number - 527\n\n");

//Given
T = 50;//[C] - Temperature
P = 64;//[kPa] - Pressure
z_1 = 0.7;
z_2 = 0.3;

// log(P_sat) = A - B/(t + C)

// For acetone
A_1 = 14.37824;
B_1 = 2787.498;
C_1 = 229.664;
// For acetonitrile
A_2 = 14.88567;
B_2 = 3413.099;
C_2 = 250.523;

// At 50 C,
P_1_sat = exp(A_1 - B_1/(T + C_1));//[kPa]
P_2_sat = exp(A_2 - B_2/(T + C_2));//[kPa]

```

```
// Now let us calculate the K values of the
   components
K_1 = P_1_sat/P;
K_2 = P_2_sat/P;

// summation of y_i = 1, gives
// (K_1*z_1)/(1-V-K_1*V) + (K_2*z_2)/(1-V-K_2*V) = 1
// Solving we get
deff('[y]=f(V)','y=(K_1*z_1)/(1-V+K_1*V) + (K_2*z_2)
   /(1-V+K_2*V) -1');
V = fsolve(0.1,f);
L = 1 - V;
// Therefore
y_1 = (K_1*z_1)/(1-V+K_1*V);
y_2 = (K_2*z_2)/(1-V+K_2*V);

x_1 = y_1/K_1;
x_2 = y_2/K_2;

printf(" The value of V = %f\n",V);
printf(" The value of L = %f\n\n",L);
printf(" The liquid phase composition is, x_1 = %f
   and x_2 = %f\n",x_1,x_2);
printf(" The vapour phase composition is, y_1 = %f
   and y_2 = %f",y_1,y_2);
```

Scilab code Exa 15.11 Calculation of temperature

```
clear;
clc;
funcprot(0);

//Example - 15.11
//Page number - 528
printf("Example - 15.11 and Page number - 528\n\n");
```

```

//Given
P = 12.25*101325*10^(-3);//[kPa]
z_1 = 0.8;
z_2 = 1 - z_1;
V = 0.4;
// log(P_1_sat) = 13.7713 - 2892.47/(T + 248.82)
// log(P_2_sat) = 13.7224 - 2151.63/(T + 236.91)

// P_1_sat = exp(13.7713 - 21892.47/(T + 248.82));
// P_2_sat = exp(13.7224 - 2151.63/(T + 236.91));

// Let the total mixture be 1 mol
// We have to assume a temperature such that,
// y_1 + y_2 = (K_1*z_1)/(1-V-K_1*V) + (K_2*z_2)/(1-
   V-K_2*V) = 1

// To assume a temperature we have to determine the
   BPT and DPT and take a temperature in between the
    range BPT to DPT

// At the bubble point the whole feed lies in the
   liquid phase and an infinitesimal amount of
   vapour is formed, therefore
x_1 = z_1;
x_2 = z_2;

// The bubble point pressure is given by
// P = x_1*(exp(13.7713 - 21892.47/(T + 248.82)) +
   x_2*(exp(13.7224 - 2151.63/(T + 236.91)));

deff('[y]=f(T)','y=x_1*(exp(13.7713 - 1892.47/(T +
   248.82))) + x_2*(exp(13.7224 - 2151.63/(T +
   236.91))) - P');
T_1 = fsolve(0.1,f);
BPT = T_1;

// At the dew point the whole feed lies in the
```

```
    vapour phase and a drop of liquid is formed,
    therefore
y_1 = z_1;
y_2 = z_2;

// The dew point equation is given by
// 1/P = y_1/P_1_sat + y_2/P_2_sat
deff('[y]=f1(T)','y=1/(y_1/(exp(13.7713 - 1892.47/(T
     + 248.82))) + y_2/(exp(13.7224 - 2151.63/(T +
    236.91)))) - P');
T_2 = fsolve(0.1,f1);
DPT = T_2;

// Now the assumed temperature should be in the
    range of BPT and DPT
// Let the assumed temperature be 47 C
T = 47;//[C]
error = 10;
while(error>0.001)
    P_1_sat = exp(13.7713 - 1892.47/(T + 248.82));
    P_2_sat = exp(13.7224 - 2151.63/(T + 236.91));
    K_1 = P_1_sat/P;
    K_2 = P_2_sat/P;
    y1 = (K_1*z_1)/(1-V+K_1*V);
    y2 = (K_2*z_2)/(1-V+K_2*V);
    y = y1 + y2;
    error=abs(y - 1);
    T = T - 0.0001;
end

printf(" The temperature when 40 mol %% of mixture
    is in the vapour is %f C",T);
```

Scilab code Exa 15.12 Determination of number of moles in liquid and vapour phase

```
clear;
clc;

//Example - 15.12
//Page number - 529
printf("Example - 15.12 and Page number - 529\n\n");

//Given
T = 105;//[C]
P = 1.5;//[atm]
P = P*101325*10^(-3);//[kPa]
z = [0.4,0.3667,0.2333];// Feed composition
x = [0.3,0.3,0.4];// Equilibrium liquid phase
    composition
y = [0.45,0.40,0.15];// Equilibrium vapour phase
    composition

// From the material balance equation of component
    1, we get
// (L + V)*z_1 = L*x_1 + V*y_1

// Since total moles are one, therefore  L + V = 1
    and thus
// z_1 = L*x_1 + (1-L)*y_1

for i=1:3;
    L = (z(i) - y(i))/(x(i) - y(i));
    V = 1 - L;
    printf(" The number of moles in liquid phase (z
        = %f) is given by ,L = %f\n",z(i),L);
    printf(" The number of moles in vapour phase (z
        = %f) is given by ,V = %f\n\n",z(i),V);
end
```

Scilab code Exa 15.13 Determination of vapour and liquid phase composition

```
clear;
clc;

//Example - 15.13
//Page number - 530
printf("Example - 15.13 and Page number - 530\n\n");

//Given
T = 90;//[C]
P = 1;//[atm]
P = P*101325*10^(-3);//[kPa]
z_1 = [0.1,0.5,0.8];

// log(P_1_sat) = 13.8594 - 2773.78/(t + 220.07)
// log(P_2_sat) = 14.0098 - 3103.01/(t + 219.79)

//At T = 90 C
P_1_sat = exp(13.8594 - 2773.78/(T + 220.07));
P_2_sat = exp(14.0098 - 3103.01/(T + 219.79));
K_1 = P_1_sat/P;
K_2 = P_2_sat/P;

// For z_1 = 0.1
// y1 = (K_1*z_1(i))/(1-V+K_1*V);
// y2 = (K_2*z_2)/(1-V+K_2*V);
// We do not get a value between 0 and 1 such that,
   y = y1 + y2 = 1;
// This means that at z_1 = 0.1 two phases do not
   exist.
// At given temperature and pressure, let us
   determine the equilibrium liquid and vapour
   phase compositions

x_1 = (P - P_2_sat)/(P_1_sat - P_2_sat);
y_1 = (x_1*P_1_sat)/(P);
```

```

// For two phase region to exist at 90 C and 101.325
    kPa, z_1 sholud lie between x_1 and y_1

printf(" For two phase region to exist at 90 C and
   101.325 kPa, z_1 sholud lie between %f and %f\n\n
   ",x_1,y_1);
printf(" For z_1 = 0.1 and z_1 = 0.5, only liquid
   phase exists (V = 0).\n\n");
printf(" For z_1 = 0.8, only vapour exists (V = 1).\
   n")
```

Scilab code Exa 15.14 Preparation of table having composition and pressure data

```
clear;
clc;

//Example - 15.14
//Page number - 531
printf("Example - 15.14 and Page number - 531\n\n");

//Given
T = 90;//[C]
P = 1;//[atm]
P = P*101325*10^(-3);//[kPa]
z_1 = [0.1,0.5,0.8];

// log(P_1_sat) = 13.8594 - 2773.78/(t + 220.07)
// log(P_2_sat) = 14.0098 - 3103.01/(t + 219.79)

//(a)
//At T = 90 C
P_1_sat = exp(13.8594 - 2773.78/(T + 220.07));
P_2_sat = exp(14.0098 - 3103.01/(T + 219.79));
```

```
K_1 = P_1_sat/P;

x_1 = [0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0];
P_prime = zeros(1,11);
x_2 = zeros(11);
y_1 = zeros(11);

printf(" (a).\n\n");
printf(" x_1     \t\t  P    \t\t    y_1        \n\n");

for i=1:11;
    x_2(i) = 1 - x_1(i);
    P_prime(i) = x_1(i)*P_1_sat + x_2(i)*P_2_sat;
    y_1(i) = (x_1(i)*P_1_sat)/P_prime(i);
    printf(" %f \t  %f  \t    %f  \n",x_1(i),P_prime(
       i),y_1(i));
end

//(b)
T_1_sat = 2773.78/(13.8594-log(P)) - 220.07;//[C]
T_2_sat = 3103.01/(14.0098-log(P)) - 219.79;//[C]

T_prime =
   [110.62,107,104,101,98,95,92,89,86,83,80.09];

P1_sat = zeros(11);
P2_sat = zeros(11);
x_1 = zeros(11);
y_1 = zeros(11);

printf(" \n\n (b).\n\n");
printf(" T(C)  \t\t  P_1_sat (kPa)  \t\t P_2_sat (
   kPa) \t\t  x_1  \t\t  y_1 \n\n");

for j=1:11;
    P1_sat(j) = exp(13.8594 - 2773.78/(T_prime(j) +
       220.07));
    P2_sat(j) = exp(14.0098 - 3103.01/(T_prime(j) +
        219.79));
    x_1(j) = (P-P2_sat(j))/(P1_sat(j)-P2_sat(j));
    y_1(j) = (x_1(j)*P1_sat(j))/P;
    printf(" %f \t    %f     \t   %f     \t        %f   \
        t         %f \n",T_prime(j),P1_sat(j),P2_sat(j)
        ,x_1(j),y_1(j));
end
```

Scilab code Exa 15.15 Calculation of DPT and BPT

```
clear;
clc;
funcprot(0);

//Example - 15.15
//Page number - 533
printf("Example - 15.15 and Page number - 533\n\n");

//Given
// log(Y1) = 0.95*x_2^(2)
// log(Y2) = 0.95*x_1^(2)
P_1_sat = 79.80;//[kPa]
P_2_sat = 40.45;//[kPa]

//(1)
T = 373.15;//[K]
x_1 = 0.05;
x_2 = 1 - x_1;
Y1 = exp(0.95*x_2^(2));
Y2 = exp(0.95*x_1^(2));

// The total pressure of the system is given by
P = x_1*Y1*P_1_sat + x_2*Y2*P_2_sat;//[kPa]
y_1 = x_1*Y1*P_1_sat/P;
y_2 = x_2*Y2*P_2_sat/P;
```

```

printf(" (1).The first bubble is formed at %f kPa
   and the composition , y_1 = %f\n\n",P,y_1);

//(2)
T = 373.15;//[K]
y_1_prime = 0.05;
y_2_prime = 1 - y_1_prime;

// Let us assume a value of x_1 ,
x_1_prime = 0.0001;

error = 10;
while(error>0.001)
    x_2_prime = 1 - x_1_prime;
    Y1_prime = exp(0.95*x_2_prime^(2));
    Y2_prime = exp(0.95*x_1_prime^(2));
    P_prime = x_1_prime*Y1_prime*P_1_sat + x_2_prime
       *Y2_prime*P_2_sat;
    x_1 = (y_1_prime*P_prime)/(Y1_prime*P_1_sat);
    error=abs(x_1_prime - x_1);
    x_1_prime = x_1_prime + 0.00001;
end

P_2 = x_1_prime*Y1_prime*P_1_sat + x_2_prime*
   Y2_prime*P_2_sat;

printf(" (2).The first drop is formed at %f kPa and
   has the composition , x_1 = %f",P_2,x_1_prime);
```

Scilab code Exa 15.16 Calculation of pressure

```
clear;
clc;
funcprot(0);
```

```

//Example - 15.16
//Page number - 534
printf("Example - 15.16 and Page number - 534\n\n");

//Given
T = 78.15;//[C]
P_1_sat = 755;//[mm Hg]
P_2_sat = 329;//[mm Hg]

z_1 = 0.3;
V = 0.5;

// log(Y1) = 0.845/(1 + 0.845*(x_1/x_2))^(2)
// log(Y2) = 1/(1 + 1.183*(x_2/x_1))^(2)

// A value of x_1 is to determined for which  V =
   0.5
// Let us assume a value of x_1, say x_1 = 0.150
x_1 = 0.150;

error = 10;
while(error>0.001)
    x_2 = 1 - x_1;
    Y1 = exp(0.845/(1 + 0.845*(x_1/x_2))^(2));
    Y2 = exp(1/(1 + 1.183*(x_2/x_1))^(2));
    P = x_1*Y1*P_1_sat + x_2*Y2*P_2_sat;
    y_1 = (x_1*Y1*P_1_sat)/P;
    V_prime = (z_1 - x_1)/(y_1 - x_1);
    error=abs(V_prime - V);
    x_1 = x_1 + 0.00001;
end

P_prime = x_1*Y1*P_1_sat + x_2*Y2*P_2_sat;//[mm hg]

// At x_1 , V = 0.5,
// Therefore when the mixture is 50 % vaporized at
   78.15 C the mole fraction of component 1 in the
```

```
    liquid phase is x_1 and the system pressure is
    P_prime

printf(" The required pressure is %f mm Hg\n\n",
    P_prime);
printf(" and the mole fraction of component 1 in the
     liquid phase for this pressure is x_1 = %f\n\n",
    x_1);
```

Scilab code Exa 15.17 Calculation of van Laar activity coefficient parameters

```
clear;
clc;

//Example - 15.17
//Page number - 536
printf("Example - 15.17 and Page number - 536\n\n");

//Given
T = 25;//[C] - Temperature
P =
    [118.05,124.95,137.90,145.00,172.90,207.70,227.70,237.85,253.90,2
    //[mm Hg]
x_1 =
    [0.0115,0.0160,0.0250,0.0300,0.0575,0.1125,0.1775,0.2330,0.4235,0

y_1 =
    [0.1810,0.2250,0.3040,0.3450,0.4580,0.5670,0.6110,0.6325,0.6800,0


// Pressure value for which x_1 = y_1 = 0,
    corresponds to P_2_sat ,therefore
P_2_sat = 97.45;//[mm Hg]
// Pressure value for which x_1 = y_1 = 1,
    corresponds to P_1_sat , therefore
P_1_sat = 230.40;//[mm Hg]

x_2 = zeros(1,15);
y_2 = zeros(1,15);
Y1 = zeros(1,15);
Y2 = zeros(1,15);
GE_RT = zeros(1,15);
x1x2_GE_RT = zeros(1,15);
for i=1:15;
    x_2(1,i) = 1 - x_1(i);
    y_2(1,i) = 1 - y_1(i);
    Y1(1,i) = (y_1(i)*P(i))/(x_1(i)*P_1_sat);
    Y2(1,i) = (y_2(i)*P(i))/(x_2(i)*P_2_sat);
    GE_RT(1,i) = x_1(i)*log(Y1(i)) + x_2(i)*log(Y2(i
       ));// G_E/(R*T)
    x1x2_GE_RT(1,i) = (x_1(i)*x_2(i))/GE_RT(i);
end

[M,N,sig]=reglin(x_1,x1x2_GE_RT);

// Linear regression between x_1 and x_1*x_2/(G_E/R*
   T) gives intercept = N and slope = M

// van Laar equation is x_1*x_2/(G_E/R*T) = 1/A +
   (1/B - 1/A)
// 1/A = N
A = 1/N;
B = 1/(M + 1/A);

printf(" The value of Van Laar coefficient A =  %f\n
   \n",A);
printf(" The value of Van Laar coefficient B =  %f\n
   ",B);
```

Scilab code Exa 15.18 Prediction of azeotrrope formation

```
clear;
clc;

//Example - 15.18
//Page number - 541
printf("Example - 15.18 and Page number - 541\n\n");

//Given
T = 343.15;//[K] - Temperature
// At 343.15 K
// log(Y1) = 0.95*x_2^(2)
// log(Y2) = 0.95*x_1^(2)
P_1_sat = 79.80;//[kPa]
P_2_sat = 40.50;//[kPa]

// At x_1 = 0,
Y1_infinity = exp(0.95);
alpha_12_x0 = (Y1_infinity*P_1_sat)/(P_2_sat);
// At x_1 = 1,
Y2_infinity = exp(0.95);
alpha_12_x1 = (P_1_sat)/(Y2_infinity*P_2_sat);

// Within the range alpha_12_x0 and alpha_12_x1 , the
     relative volatility  continuously decrease and
    thus a value of 1.0 is obtained and thus
    azeotrope is formed.
// At azeotrope , Y1*P1_sat = Y2*P2_sat
// Y2/Y1 = P_1_sat/P_2_sat
// Taking logarithm of both sides we get
// log(Y2) - log(Y1) = log(P_1_sat/P_2_sat)
// 0.95*x_1^(2) - 0.95*x_2^(2) = log(P_1_sat/P_2_sat
    )
// Solving the above equation
deff('[y]=f(x_1)','y=0.95*x_1^(2) - 0.95*(1-x_1)^(2)
     - log(P_1_sat/P_2_sat)');
x_1 = fsolve(0.1,f);
```

```

// At x_1
x_2 = 1 - x_1;
Y1 = exp(0.95*x_2^(2));
Y2 = exp(0.95*x_1^(2));
P = x_1*Y1*P_1_sat + x_2*Y2*P_2_sat;//[kPa] -
    Azeotrope pressure
y_1 = (x_1*Y1*P_1_sat)/P;

// Since x_1 = y_1, (almost equal) ,the above
    condition is of azeotrope formation

// Since alpha_12 is a continuous curve and in
    between a value of alpha_12 = 1, shall come and
    at this composition the azeotrope shall get
    formed.

printf(" Since (alpha_12_x=0) = %f and (alpha_12_x
    =1) = %f \n",alpha_12_x0,alpha_12_x1);
printf(" and since alpha_12 is a continuous curve
    and in between a value of alpha_12 = 1, shall
    come and at this composition the azeotrope shall
    get formed.\n\n")
printf(" The azeotrope composition is x_1 = y_1 = %f
    \n\n",x_1);
printf(" The azeotrope presssure is %f kPa\n",P);
```

Scilab code Exa 15.19 Tabulation of activity coefficients relative volatility and compositions

```
clear;
clc;

//Example - 15.19
//Page number - 541
```

```
printf("Example - 15.19 and Page number - 541\n\n");

//Given
T = 45;//[C] - Temperature

x_1 =
   [0.0455,0.0940,0.1829,0.2909,0.3980,0.5069,0.5458,0.5946,0.7206,0

y_1 =
   [0.1056,0.1818,0.2783,0.3607,0.4274,0.4885,0.5098,0.5375,0.6157,0

P =
   [31.957,33.553,35.285,36.457,36.996,37.068,36.978,36.778,35.792,3


// Pressure value for which x_1 = y_1 = 0,
   corresponds to P_2_sat,therefore
P_2_sat = 29.819;//[kPa]
// Pressure value for which x_1 = y_1 = 1,
   corresponds to P_1_sat,therefore
P_1_sat = 27.778;//[kPa]

x_2 = zeros(1,12);
y_2 = zeros(1,12);
Y1 = zeros(1,12);
Y2 = zeros(1,12);
alpha_12 = zeros(1,12);
GE_RT = zeros(1,12);
x1x2_GE_RT = zeros(1,12);

printf(" x_1  \t\t  y_1    \t   P   \t\t     Y1   \t\
   tY2  \t     alpha_12     \t   G_E/RT \t  x1*x2/(G_E/
   RT)\n\n");

for i=1:12;
    x_2(1,i) = 1 - x_1(i);
    y_2(1,i) = 1 - y_1(i);
    Y1(1,i) = (y_1(i)*P(i))/(x_1(i)*P_1_sat);
    Y2(1,i) = (y_2(i)*P(i))/(x_2(i)*P_2_sat);
    alpha_12(1,i) = (y_1(i)/x_1(i))/(y_2(i)/x_2(i));
    GE_RT(1,i) = x_1(i)*log(Y1(i)) + x_2(i)*log(Y2(i
        ));// G_E/(R*T)
    x1x2_GE_RT(1,i) = (x_1(i)*x_2(i))/GE_RT(i);
    printf(" %f\t   %f\t   %f \t       %f   \t    %f \t
           %f\t    %f   \t%f\n\n",x_1(i),y_1(i),P(i),
        Y1(i),Y2(i),alpha_12(i),GE_RT(i),x1x2_GE_RT(i
        ));
end

[M,N,sig]=reglin(x_1,x1x2_GE_RT);

// Linear regression between x_1 and x_1*x_2/(G_E/R*
    T) gives intercept = N and slope = M

// Now let us assume the system to follow van Laar
    activity coefficient model.
// x_1*x_2/(G_E/(R*T)) = x_1/B + x_2/A =  x_1/B + (1
    - x_1)/A  =  1/A + (1/B - 1/A)*x_1 = N + M*x_1

// 1/A = N
A = 1/N;
// (1/B - 1/A) = M
B = 1/(M + 1/A);

printf("\n\n")
printf(" The value of van Laar parameters are, A =
    %f and B = %f \n\n",A,B);

Y1_infinity = exp(A);
Y2_infinity = exp(B);


// Azeotrope is formed when maxima ( or mainina) in
    pressure is observed and relative volatility
    becomes 1.
// This is the case for x_1 between 0.2980 and
```

```
   0.5458.
// The ezeotropr os maximum pressure  (and thus
   minimum boiling) because at azeotrope the system
   pressure is greater than vapour pressure of pure
   components.

// Now let us calculate the azeotrope composition.
// At azeotrope, Y1*P1_sat = Y2*P2_sat
// log(Y1/Y2) = log(P_2_sat/P_1_sat)
// From van Laar model we get
// log(P_2_sat/P_1_sat) = (A*B^(2)*2*x_2^(2))/(A*x_1
    + B*x_2)^(2) + (B*A^(2)*2*x_1^(2))/(A*x_1 + B*
   x_2)^(2)
// Solving the above equation
deff('[y]=f(x_1)','y= log(P_2_sat/P_1_sat) - (A*B
   ^(2)*(1-x_1)^(2))/(A*x_1 + B*(1-x_1))^(2) + (B*A
   ^(2)*x_1^(2))/(A*x_1 + B*(1-x_1))^(2)');
x_1 = fsolve(0.1,f);

printf(" The azeotrope composition is given by x_1 =
    y_1 = %f\n",x_1);
```

Scilab code Exa 15.20 Tabulation of partial pressure and total pressure data of components

```
clear;
clc;

//Example - 15.20
//Page number - 541
printf("Example - 15.20 and Page number - 543\n\n");

//Given
T = 25;//[C] - Temperature
P_1_sat = 230.4;//[mm Hg]
```

```
P_2_sat = 97.45;//[mm Hg]
Y1_infinity = 8.6;
Y2_infinity = 6.6;

// Assuming ideal vpour behaviour means that phi = 1
    and since system pressure is low, therefore
// f_i = P_i_sat
// Assuming the activity coefficients to follow van
   Laar model we get
A = log(Y1_infinity);
B = log(Y2_infinity);

//log(Y1) = A/(1+ (A*x_1)/(B*x_2))^(2)
// log(Y2) = B/(1+ (B*x_2)/(A*x_1))^(2)

x_1 = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9];

x_2 = zeros(9);
Y1 = zeros(9);
Y2 = zeros(9);
y1_P = zeros(9);
y2_P = zeros(9);
P = zeros(9);
y_1 = zeros(9);

printf(" (a).\n\n");
printf(" x_1         \t\t     Y1      \t\t     Y2      \t\t
      y1*P      \t\t      y2*P    \t\t    P      \t\t     y_1
   \n\n");

for i=1:9;
    x_2(i) = 1 - x_1(i);
    Y1(i) = exp(A/(1+ (A*x_1(i))/(B*x_2(i)))^(2));
    Y2(i) = exp(B/(1+ (B*x_2(i))/(A*x_1(i)))^(2));
    y1_P(i) = x_1(i)*Y1(i)*P_1_sat;
    y2_P(i) = x_2(i)*Y2(i)*P_2_sat;
    P(i) = x_1(i)*Y1(i)*P_1_sat + x_2(i)*Y2(i)*
       P_2_sat;
    y_1(i) = (x_1(i)*Y1(i)*P_1_sat)/P(i);
    printf(" %f\t\t %f\t\t %f \t\t %f \t\t %f \t\t
       %f \t   %f\n\n",x_1(i),Y1(i),Y2(i),y1_P(i),
       y2_P(i),P(i),y_1(i));

end

//(b)
// The total system pressure versus x_1 shows a
   maxima and thus azeotrope is formed by the VLE
   system
// The maxima occurs in the range of x_1 = 0.6 to
   0.8, so an  azeotrope is formed in this
   composition range

// At the azeotrope  point, Y1*P1_sat = Y2*P2_sat
// log(Y1) - log(Y2) = log(P_2_sat/P_1_sat)
// On putting the values and then solving the above
   equation we get
deff('[y]=f(x_1)','y= A/(1+1.14*x_1/(1-x_1))^(2)- B
   /(1+0.877*(1-x_1)/x_1)^(2) - log(P_2_sat/P_1_sat)
   ');
x_1_prime = fsolve(0.1,f);

// At x_1
x_2_prime = 1 - x_1_prime;
Y1_prime = exp(A/(1+ (A*x_1_prime)/(B*x_2_prime))
   ^(2));
Y2_prime = exp(B/(1+ (B*x_2_prime)/(A*x_1_prime))
   ^(2));
P_prime = x_1_prime*Y1_prime*P_1_sat + x_2_prime*
   Y2_prime*P_2_sat;//[kPa] - Azeotrope pressure
y_1_prime = (x_1_prime*Y1_prime*P_1_sat)/P_prime;

// Since x_1 = y_1,azeotrope formation will take
   place
printf(" (b)\n\n");
printf(" The total system pressure versus x_1 shows
   a maxima and thus azeotrope is formed by the VLE
   system\n\n");
printf(" The azeotrope composition is x_1 = y_1 = %f
   \n\n",x_1_prime);
printf(" The azeotrope presssure is %f mm Hg\n",
   P_prime);
```

Scilab code Exa 15.21 Determination of azeotrope formation

```
clear;
clc;

//Example - 15.21
//Page number - 544
printf("Example - 15.21 and Page number - 544\n\n");

//Given
T = 50;//[C]
// At 50 C
P_1_sat = 0.67;//[atm]
P_2_sat = 0.18;//[atm]
Y1_infinity = 2.5;
Y2_infinity = 7.2;

//(1)
// alpha_12 = (y_1/x_1)/(y_2/x_2) = (Y1*P_1_sat)/((
   Y2*P_2_sat))
// At  x_1 tending to zero,
alpha_12_x0 = (Y1_infinity*P_1_sat)/(P_2_sat);
// At  x_1 tending to 1,
alpha_12_x1 = (P_1_sat)/((Y2_infinity*P_2_sat));

// Since alpha_12 is a continuous curve and in
   between a value of alpha_12 = 1, shall come and
   at this composition the azeotrope shall get
```

```
    formed.
printf(" (1).Since (alpha_12_x=0) = %f and (
    alpha_12_x=1) = %f \n",alpha_12_x0,alpha_12_x1);
printf("      and since alpha_12 is a continuous
    curve and in between a value of alpha_12 = 1,
    shall come and at this composition azeotrope
    shall get formed.\n\n")

//(b)
// Since the activity coefficient values are greater
     than 1 ,therefore the deviations from Roult's
    law is positive
// and the azeotrope is maximum pressure (or minimum
     boiling)
printf(" (2).Since the activity coefficient values
    are greater than 1 ,therefore the deviations from
     Roults law is positive\n");
printf("      and the azeotrope is maximum pressure (
    or minimum boiling)\n\n");

//(3)
// Let us assume the system to follow van Laar
    activity coefficient model
A = log(Y1_infinity);
B = log(Y2_infinity);

// log(Y1) = A/(1+ (A*x_1)/(B*x_2))^(2)
// log(Y2) = B/(1+ (B*x_2)/(A*x_1))^(2)

// At the azeotrope  point, Y1*P1_sat = Y2*P2_sat
// log(Y1) - log(Y2) = log(P_2_sat/P_2_sat)
// On putting the values and then solving the above
    equation
deff('[y]=f(x_1)','y= A/(1+ (A*x_1)/(B*(1-x_1)))^(2)
    - B/(1+ (B*(1-x_1))/(A*x_1))^(2) - log(P_2_sat/
    P_1_sat)');
x_1 = fsolve(0.1,f);

```

```
// At x_1
x_2 = 1 - x_1;
Y1 = exp(A/(1+ (A*x_1)/(B*x_2))^(2));
Y2 = exp(B/(1+ (B*x_2)/(A*x_1))^(2));
P = x_1*Y1*P_1_sat + x_2*Y2*P_2_sat;//[kPa] -
    Azeotrope pressure
y_1 = (x_1*Y1*P_1_sat)/P;

// Since x_1 = y_1,the azeotrope formation will take
    place

printf(" (3).The azeotrope composition is x_1 = y_1
    = %f\n",x_1);
printf("     The azeotrope presssure is %f atm\n",P)
    ;
```

Scilab code Exa 15.22 Tabulation of pressure and composition data

```
clear;
clc;

//Example - 15.22
//Page number - 545
printf("Example - 15.22 and Page number - 545\n\n");

//Given
T = 25;//[C]
// At 50 C
P_1_sat = 7.866;//[kPa]
P_2_sat = 3.140;//[kPa]

// G_E/(R*T) = 1.4938*x_1*x_2/(1.54*x_1 + 0.97*x_2)

// The excess Gibbs free energy expression can be
    written as
```

```
// x_1*x_2/(G_E/(R*T)) = 1.54*x_1/1.4938 + 0.97*x_2
   /1.4938 = x_1/0.97 + x_2/1.54

// Comparing with the van Laar expression
// x_1*x_2/(G_E/(R*T)) = x_1/B + x_2/A,  we get
A = 1.54;
B = 0.97;

// The activity coefficients are thus given by
// log(Y1) = A/(1+ (A*x_1)/(B*x_2))^(2)
// log(Y2) = B/(1+ (B*x_2)/(A*x_1))^(2)

x_1 = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,0.95];

x_2 = zeros(1,10);
Y1 = zeros(1,10);
Y2 = zeros(1,10);
P = zeros(1,10);
y_1 = zeros(1,10);

printf(" x_1       \t\t    Y1    \t\t    Y2    \t\t P
    (kPa)    \t\t  y_1 \n\n");

for i=1:10;
    x_2(1,i) = 1 - x_1(i);
    Y1(1,i) = exp(A/(1+ (A*x_1(i))/(B*x_2(i)))^(2));
    Y2(1,i) = exp(B/(1+ (B*x_2(i))/(A*x_1(i)))^(2));
    P(1,i) = x_1(i)*Y1(i)*P_1_sat + x_2(i)*Y2(i)*
       P_2_sat;
    y_1(1,i) = (x_1(i)*Y1(i)*P_1_sat)/P(i);
    printf(" %f\t\t %f\t\t %f \t\t %f \t\t  %f \n\n"
       ,x_1(i),Y1(i),Y2(i),P(i),y_1(i));

end

// The azeotrope is formed near x_1 = 0.95 as in
   this region a maxima in pressure is obtained.

```

```
// At the azeotrope  point, Y1*P1_sat = Y2*P2_sat
// log(Y1) - log(Y2) = log(P_2_sat/P_2_sat)
// On putting the values and then solving the above
   equation
deff('[y]=f(x_1)','y= A/(1+ (A*x_1)/(B*(1-x_1)))^(2)
   - B/(1+ (B*(1-x_1))/(A*x_1))^(2) - log(P_2_sat/
   P_1_sat)');
x_1_prime = fsolve(0.1,f);

// At x_1
x_2_prime = 1 - x_1_prime;
Y1_prime = exp(A/(1+ (A*x_1_prime)/(B*x_2_prime))
   ^(2));
Y2_prime = exp(B/(1+ (B*x_2_prime)/(A*x_1_prime))
   ^(2));
P_prime = x_1_prime*Y1_prime*P_1_sat + x_2_prime*
   Y2_prime*P_2_sat;//[kPa] - Azeotrope pressure
y_1_prime = (x_1_prime*Y1_prime*P_1_sat)/P_prime;

// Since x_1_prime = y_1_prime,the azeotrope
   formation will take place

printf(" \n\nPart II \n\n");
printf(" The azeotrope composition is x_1 = y_1 = %f
   \n",x_1_prime);
printf(" The azeotrope presssure is %f kPa \n",
   P_prime);
```

Scilab code Exa 15.23 Determination of van Laar parameters

```
clear;
clc;

//Example - 15.23
//Page number - 547
```

```
printf("Example - 15.23 and Page number - 546\n\n");

//Given
T = 58.7;//[C]
P = 1;//[atm]
P = P*101325*10^(-3);//[kPa]

// log(P_sat) = 16.6758 - 3674.49/(t + 226.45) - For
    ethyl alcohol
// log(P_sat) = 13.8216 - 2697.55/(t + 224.37) - For
    hexane

// Let us take hexane as (1) and ethanol as (2)
// At 58.7 C
P_1_sat = exp(13.8216 - 2697.55/(T + 224.37));//[kPa
   ]
P_2_sat = exp(16.6758 - 3674.49/(T + 226.45));//[kPa
   ]

Y1 = P/P_1_sat;
Y2 = P/P_2_sat;

x_2 = 0.332;// Mol % of ethanol (given)
x_1 = 1 - x_2;// Mol % of hehane

// The van Laar parameters are given by
A = ((1 + (x_2*log(Y2))/(x_1*log(Y1)))^(2))*log(Y1);
B = ((1 + (x_1*log(Y1))/(x_2*log(Y2)))^(2))*log(Y2);

printf(" The value of van Laar parameters are, A =
   %f and B = %f \n\n",A,B);

// Now let us calvulate the distribution coefficient
    K
x_1_prime = 0.5;//[given]
x_2_prime = 1 - x_1_prime;

// The activity coefficients are thus given by
// log(Y1) = A/(1+ (A*x_1)/(B*x_2))^(2)
// log(Y2) = B/(1+ (B*x_2)/(A*x_1))^(2)

Y1_prime = exp(A/(1+ (A*x_1_prime)/(B*x_2_prime))
   ^(2));
Y2_prime = exp(B/(1+ (B*x_2_prime)/(A*x_1_prime))
   ^(2));
P_prime = x_1_prime*Y1_prime*P_1_sat + x_2_prime*
   Y2_prime*P_2_sat;

// We have,  K_1 = y_1/x_1 = Y1*P_1_sat/P
K_1 = Y1_prime*P_1_sat/P_prime;

printf(" The distribution coefficient is given by
   K_1 = %f\n",K_1)
```

Chapter 16

Other Phase Equilibria

Scilab code Exa 16.1 Determination of solubility

```
clear;
clc;
funcprot(0);

// Example - 16.1
// Page number - 564
printf("Example - 16.1 and Page number - 564\n\n");

// Given
T = 0 + 273.15;//[K] - Temperature
P = 20*10^(5);//[Pa] - Pressure
R = 8.314;//[J/mol*K] - Universal gas constant

//componenet 1 : methane (1)
//componenet 2 : methanol (2)

H_constant = 1022;//[bar] - Henry's law constant
H_constant = H_constant*10^(5);//[Pa]

// The second virial coefficients are
B_11 = -53.9;//[cm^(3)/mol]
```

```
B_11 = B_11*10^(-6);//[m^(3)/mol]
B_12 = -166;//[cm^(3)/mol]
B_12 = B_12*10^(-6);//[m^(3)/mol]
B_22 = -4068;//[cm^(3)/mol]
B_22 = B_22*10^(-6);//[m^(3)/mol]

den_meth = 0.8102;//[g/cm^(3)] - Density of methanol
    at 0 C
Mol_wt_meth = 32.04;// Molecular weight of methanol
P_2_sat = 0.0401;//[bar] - Vapour pressure of
   methanol at 0 C

//The molar volume of methanol can be calculated as
V_2_liq = (1/(den_meth/Mol_wt_meth))*10^(-6);//[m
   ^(3)/mol]

//The phase equilibrium equation of the components
   at high pressure
//y1*phi_1*P = x_1*H_1
//y2*phi_2*P = x_2*H_2

//Since methane follows Henry's law therefore
   methanol follows the lewis-Rnadall rule
//f_2 is the fugacity of the compressed liquid which
    is calculated using
//f_2 = f_2_sat*exp[V_2_liq*(P - P_sat_2)/(R*T)]
//where f_2_sat can be calculated using virial
   equation
// log(phi_2_sat) = log(f_2_sat/P_2_sat) = (B_22*
   P_2_sat)/(R*T)

f_2_sat = P_2_sat*exp((B_22*P_2_sat*10^(5))/(R*T));
   //[bar]

//Putting the value of 'f_2_sat' in the expression
   of f_2 , we get
f_2 = f_2_sat*exp(V_2_liq*(P - P_2_sat*10^(5))/(R*T)
   );//[bar]
```

```

//Now let us calculate the fugacity coefficients of the species in the vapour mixture
del_12 = 2*B_12 - B_11 - B_22;//[m^(3)/mol]

//log(phi_1) = (P/(R*T))*(B_11 + y2^(2)*del_12)
//log(phi_2) = (P/(R*T))*(B_22 + y1^(2)*del_12)


//The calculation procedure is to assume a value of y1, calculate 'phi_1' and 'phi_2' and calculate 'x_1' and 'x_2' from the phase equilibrium equations and see whether x_1 + x_2 = 1,if not then another value of y1 is assumed.

y2 = 0.1;
error=10;

while(error>0.001)
    y1=1-y2;
    phi_1 = exp((P/(R*T))*((B_11 + y2^(2)*del_12)));
    phi_2 = exp((P/(R*T))*((B_22 + y1^(2)*del_12)));
    x_1 = (y1*phi_1*P)/H_constant;
    x_2 = (y2*phi_2*P)/(f_2*10^(5));
    x = x_1 + x_2;
    error=abs(1-x);
    y2=y2 - 0.000001;
end

printf(" The solubility of methane in methanol is given by x1 = %f\n",x_1);
```

Scilab code Exa 16.2 Determination of solubility

```
clear;
```

```
clc;

// Example - 16.2
// Page number - 566
printf("Example - 16.2 and Page number - 566\n\n");

// Given
x_C2H6_1 = 0.33*10^(-4);// Solubility of ethane in
    water at 25 C and 1 bar

//componenet 1 : ethane (1)
//componenet 2 : water (2)

// Z = 1 - 7.63*10^(3)*P - 7.22*10^(-5)*P^(2)

//The phase equilibrium equation of ethane is
//f_1_V = x_1*H_1
//since vapour is pure gas,  f_1_V = x_1*H_1   or,
    phi_1*P = x_1*H_1,  where 'phi_1' is fugacity
    coefficient of pure ethane
// log(phi) = integral('Z-1)/P) from limit '0' to 'P
    '

P1 = 0;
P2 = 1;
P3 = 35;
intgral = integrate('(1-7.63*10^(-3)*P-7.22*10^(-5)*
   P^(2)-1)/P','P',P1,P2);
phi_1_1 = exp(intgral);// - Fugacity coefficient of
    ethane at 1 bar
f_1_1 = phi_1_1*P2;//[bar] - Fugacity of ethane at 1
     bar

//Similarly
intgral_1 = integrate('(1-7.63*10^(-3)*P
    -7.22*10^(-5)*P^(2)-1)/P','P',P1,P3);
phi_1_35 = exp(intgral_1);// Fugacity coefficient of
     ethane at 35 bar
```

```
f_1_35 = phi_1_35*P3;//[bar] - Fugacity of ethane at
    35 bar

// At ethane pressure of 1 bar ,  x_C2H6_1*H_1 =
    phi_1_1
H_1 = phi_1_1/x_C2H6_1;//[bar] - Henry's constant

// At ethane pressure of 35 bar ,  x_C2H6_35*H_1 =
    phi_1_35
x_C2H6_35 = f_1_35/H_1;// Solubility of ethane at 35
    bar pressure

printf("The solubility of ethane at 35 bar is given
    by   x_C2H6 = %e",x_C2H6_35);
```

Scilab code Exa 16.3 Proving a mathematical relation

```
clear;
clc;

//Example - 16.3
//Page number - 567
printf("Example - 16.3 and Page number - 567\n\n");

//This problem involves proving a relation in which
    no mathematics and no calculations are involved.
//For prove refer to this example 16.3 on page
    number 567 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
    involved.\n\n");
printf(" For prove refer to this example 16.3 on
    page number 567 of the book.")
```

Scilab code Exa 16.4 Determination of composition

```
clear;
clc;

//Example - 16.4
//Page number - 571
printf("Example - 16.4 and Page number - 571\n\n");

//Given
T = 200;//[K]
R = 8.314;//[J/mol*K] - universal gas constant
// G_E = A*x_1*x_2
A = 4000;//[J/mol]
x_1 = 0.6;// Mle fraction of feed composition

// Since A is given to be independent of temperature
UCST = A/(2*R);//[K] - Upper critical solution
    temperature
printf(" The UCST of the system is %f K\n\n",UCST);

// Since the given temperature is less than UCST
    therefore two phase can get formed at the given
    temperature.

// x1_alpha = 1 - x1_beta
// We know that,  x1_alpha*Y_1_alpha = x2_alpha*
    Y_2_alpha
// x1_alpha*exp[(A/(R*T))*(x2_alpha)^(2)] = (1 -
    x1_alpha)*exp[(A/(R*T))*(x1_alpha)^(2)]
// where use has beeen made of the fact that
    x1_alpha = 1 - x1_beta and x2_beta = 1 - x1_beta
    = x1_alpha .Taking logarithm of both side we get
// log(x1_alpha) + (A/(R*T))*(1 - x1_alpha)^(2) =
   log(1 - x1_alpha) + (A/(R*T))*x1_alpha^(2)
// log(x1_alpha/(1-x1_alpha)) = (A/(R*T))*(2*
   x1_alpha - 1)

deff('[y]=f(x1_alpha)','y= log(x1_alpha/(1-x1_alpha)
   ) - (A/(R*T))*(2*x1_alpha - 1)');
x1_alpha = fsolve(0.1,f);
x1_beta = fsolve(0.9,f);
// Because of symmetry 1 - x1_beta = x1_alpha

// It can be seen that the equation,  log(x1/(1-x1))
    = (A/(R*T))*(2*x1 - 1) has two roots.
// The two roots acn be determined by taking
   different values
// Starting with x1 = 0.1, we get x1 = 0.169 as the
   solution and starting with x1 = 0.9,we get x1 =
   0.831 as the solution.
// Thus x1 = 0.169 is the composition of phase alpha
    and x1 = 0.831 is of phase beta
printf(" The composition of two liquid phases in
   equilibrium is given by, x1_alpha = %f and
   x1_beta = %f\n\n",x1_alpha,x1_beta);

// From the equilibrium data it is seen that if the
   feed has composition x1 less than 0.169 or more
   than 0.831 the liquid mixture is of single phase
// whereas if the overall (feed) composition is
   between 0.169 and 0.831 two phases shall be
   formed.
// The amounts of phases can also be calculated. The
    feed composition is given to be z1 = 0.6
z1 = 0.6;
// z1 = x1_alpha*alpha + x1_beta*beta
// beta = 1 - alpha
alpha = (z1-x1_beta)/(x1_alpha-x1_beta);//[mol]
Beta = 1 - alpha;//[mol]
printf(" The relative amount of phases is given by,
   alpha = %f mol and beta = %f mol\n\n\n",alpha,
   Beta);

// the relative amounts of the phases changes with
   the  feed composition

//log(x1/(1-x1)) = (A/(R*T))*(2*x1 - 1)
// If the above equation has two real roots of x1 (
   one for phase alpha and the other for phase beta)
    then two liquid phases get formed
// and if it has no  real roots then a homogeneous
   liquid mixtures is obtained.

printf(" log(x1/(1-x1)) = (A/(R*T))*(2*x1 - 1)\n");
printf(" If the above equation has two real roots of
    x1 (one for phase alpha and the other for phase
   beta) then two liquid phases get formed\n");
printf(" and if it has no  real roots then a
   homogeneous liquid mixture is obtained\n");
```

Scilab code Exa 16.5 Determination of equilibrium composition

```
clear;
clc;

//Example - 16.5
//Page number - 573
printf("Example - 16.5 and Page number - 573\n\n");

//Given
T = 300;//[K]
R = 8.314;//[J/mol*K] - universal gas constant
A = 7000;//[J/mol]

// log(x_1/(1-x_1)) = (A/(R*T))*(2*x_1-1)

```

```
deff('[y]=f(x_1)','y=log(x_1/(1-x_1))-((A/(R*T))*(2*
   x_1-1))');

x1_alpha=fsolve(0.1,f);

x1_beta=1-x1_alpha;

printf("The equilibrium compositin of the two liquid
    phase system is given by\n x1_alpha \t = %f \n
   x1_beta \t = %f",x1_alpha,x1_beta);
```

Scilab code Exa 16.6 Proving a mathematical relation

```
clear;
clc;

//Example - 16.6
//Page number - 577
printf("Example - 16.6 and Page number - 577\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 16.6 on page
   number 577 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 16.6 on
   page number 577 of the book.")
```

Scilab code Exa 16.7 Determination of freezing point depression

```
clear;
```

```
clc;

//Example - 16.7
//Page number - 579
printf("Example - 16.7 and Page number - 579\n\n");

//Given
R = 8.314;//[J/mol*K] - Universal gas constant
M_wt_meth = 32;// Molecular weight of methanol
M_wt_water = 18;// Molecular weight of water
m_meth = 0.01;//[g] - Mass of methanol added per cm
    ^(3) of solution

//Since the concentration of methanol is very small
    therefore we can assume that the density of
    solution = pure water
den_sol = 1;//[g/cm^(3)]

//The mole fraction of solute is given by
//x_2 = (moles of solute in cm^(3) of solution)/(
    moles of solute + moles of water) in 1 cm^(3) of
    solution
x_2 = (m_meth/M_wt_meth)/((m_meth/M_wt_meth)+((1-
    m_meth)/M_wt_water));

//We know that heat of fusion of water is
H_fus = -80;//[cal/g] - Enthalpy change of fusion at
     0 C
H_fus = H_fus*4.186*M_wt_water;//[J/mol]

//Therefore freezing point depression is given by
// T - T_m = (R*(T^(2))*x_2)/H_fus
T_f = 273.15;//[K] - Freezing point of water
delta_T_f = (R*(T_f^(2))*x_2)/H_fus;//[K]

printf("The depression in freezing point is given by
     \n delta_T = %f K",delta_T_f);
```

Scilab code Exa 16.8 Determination of freezing point

```
clear;
clc;

//Example - 16.8
//Page number - 580
printf("Example - 16.8 and Page number - 580\n\n");

//Given
R = 8.314;//[J/mol*K] - universal gas constant
T_f = 273.15;//[K] - Freezing point of water
m_water = 100;//[g] - Mass of water
m_NaCl = 3.5;//[g] - Mass of NaCl
M_wt_water = 18.015;// Molecular weight of water
M_wt_NaCl = 58.5;// Molecular weight of NaCl
mol_water = m_water/M_wt_water;//[mol] - Moles of
    water
mol_NaCl = m_NaCl/M_wt_NaCl;//[mol] - Moles of NaCl

H_fus = -80;//[cal/g] - Enthalpy change of fusion at
     0 C
H_fus = H_fus*4.186*M_wt_water;//[J/mol]

//Mole fraction of the solute (NaCl) is given by
x_2 = mol_NaCl/(mol_NaCl+mol_water);

//But NaCl is compietely ionized and thus each ion
    acts independently to lower the water mole
    fraction.
x_2_act = 2*x_2;// Actual mole fraction

//Now depression in freezing point is given by
// T - T_m = (R*(T^(2))*x_2_act)/H_fus
```

```
delta_T_f = (R*(T_f^(2))*x_2_act)/H_fus;//[C]

//Thus freezing point of seawater = depression in
   freezing point

printf("The freezing point of seawater is  %f C",
   delta_T_f);
```

Scilab code Exa 16.9 Proving a mathematical relation

```
clear;
clc;

//Example - 16.9
//Page number - 580
printf("Example - 16.9 and Page number - 580\n\n");

//This problem involves proving a relation in which
   no mathematics and no calculations are involved.
//For prove refer to this example 16.9 on page
   number 580 of the book.
printf(" This problem involves proving a relation in
    which no mathematics and no calculations are
   involved.\n\n");
printf(" For prove refer to this example 16.9 on
   page number 580 of the book.")
```

Scilab code Exa 16.10 Determination of boiling point elevation

```
clear;
clc;

//Example - 16.10
```

```
//Page number - 583
printf("Example - 16.10 and Page number - 583\n\n");

//Given
R = 8.314;//[J/mol*K] - universal gas constant
T_b = 373.15;//[K] - Boiling point of water
m_water = 100;//[g] - Mass of water
m_C12H22 = 5;//[g] - Mass of glucise (C12H22)
M_wt_water = 18.015;// Molecular weight of water
M_wt_C12H22 = 342.30;// Molecular weight of C12H22
mol_water = m_water/M_wt_water;//[mol] - Moles of
    water
mol_C12H22 = m_C12H22/M_wt_C12H22;//[mol] - Moles of
     C12H22

H_vap = 540;//[cal/g] - Enthalpy change of
    vaporisation
H_vap = H_vap*4.186*M_wt_water;//[J/mol]

//Mole fraction of the solute (C12H22) is given by
x_2 = mol_C12H22/(mol_C12H22+mol_water);

//The boiling point elevation is given by
// T - T_b = (R*T_b^(2)*x_2^(2))/H_vap^(2)

delta_T_b = (R*T_b^(2)*x_2)/(H_vap);

printf("The elevation in boiling point is given by \
    n delta_T = %f C",delta_T_b);
```

Scilab code Exa 16.11 Determination of osmotic pressure

```
clear;
clc;

```

```
//Example - 16.11
//Page number - 584
printf("Example - 16.11 and Page number - 584\n\n");

//Given
R = 8.314;//[J/mol*K] - Universal gas constant
T = 25 + 273.15;//[K] - Surrounding temperature
den_water = 1000;//[kg/m^(3)] - Density of water
m_water = 100;//[g] - Mass of water
m_C12H22 = 5;//[g] - Mass of glucise (C12H22)
M_wt_water = 18.015;// Molecular weight of water
M_wt_C12H22 = 342.30;// Molecular weight of C12H22
mol_water = m_water/M_wt_water;//[mol] - Moles of
    water
mol_C12H22 = m_C12H22/M_wt_C12H22;//[mol] - Moles of
     C12H22

//Mole fraction of the water is given by
x_1 = mol_water/(mol_C12H22+mol_water);

//Molar volume of water can be calculated as
V_l_water = (1/den_water)*M_wt_water*10^(-3);//[m
    ^(3)/mol]

//The osmotic pressure is given by
pi = -(R*T*log(x_1))/V_l_water;//[N/m^(2)]
pi = pi*10^(-5);//[bar]

printf("The osmotic pressure of the mixture is %f
    bar",pi);
```

Scilab code Exa 16.12 Determination of pressure

```
clear;
clc;
```

```
//Example - 16.12
//Page number - 585
printf("Example - 16.12 and Page number - 585\n\n");

//Given
R = 8.314;//[J/mol*K] - universal gas constant
T = 25 + 273.15;//[K] - Surrounding temperature
den_water = 1000;//[kg/m^(3)] - Density of water
m_water = 100;//[g] - Mass of water
m_NaCl = 3.5;//[g] - Mass of NaCl
M_wt_water = 18.015;// Molecular weight of water
M_wt_NaCl = 58.5;// Molecular weight of NaCl
mol_water = m_water/M_wt_water;//[mol] - Moles of
    water
mol_NaCl = m_NaCl/M_wt_NaCl;//[mol] - Moles of NaCl

H_fus = -80;//[cal/g] - Enthalpy change of fusion at
     0 C
H_fus = H_fus*4.186*M_wt_water;//[J/mol]

//Mole fraction of the solute (NaCl) is given by
x_2 = mol_NaCl/(mol_NaCl+mol_water);

//But NaCl is compietely ionized and thus each ion
    acts independently to lower the water mole
    fraction.
x_2_act = 2*x_2;// Actual mole fraction

x_1 = 1 - x_2_act;

//Molar volume of water can be calculated as
V_l_water = (1/den_water)*M_wt_water*10^(-3);//[m
    ^(3)/mol]

//The osmotic pressure is given by
pi = -(R*T*log(x_1))/V_l_water;//[N/m^(2)]
pi = pi*10^(-5);//[bar]
```

```
//The minimum pressure to desalinate sea water is
   nothing but the osmotic pressure

printf("The minimum pressure to desalinate sea water
    is %f bar",pi);
```

Scilab code Exa 16.13 Determination of amount of precipitate

```
clear;
clc;

//Example - 16.13
//Page number - 586
printf("Example - 16.13 and Page number - 586\n\n");

//Given
R = 8.314;//[J/mol*K] - universal gas constant
T = 173.15;//[K] - Surrounding temperature
P = 60;//[bar]
P = P*10^(5);//[Pa]

//componenet 1 : CO2 (1)
//componenet 2 : H2 (2)
P_1_sat = 0.1392;//[bar] - Vapour pressre of pure
   solid CO2
P_1_sat = P_1_sat*10^(5);//[bar]
V_s_1 = 27.6;//[cm^(3)/mol] - Molar volume of solid
   CO2
V_s_1 = V_s_1*10^(-6);//[m^(3)/mol]
n_1 = 0.01;//[mol] - Initial number of moles of CO2
n_2 = 0.99;//[mol] - Initial number of moles of H2

//Let us determine the fugacity of solid CO2 (1) at
   60 C and 173.15 K
// f_1 = f_1_sat*exp(V_s_1*(P-P_1_sat)/(R*T))
```

```

//Since vapour pressure of pure solid CO2 is very
    small, therefore
f_1_sat = P_1_sat;
f_1 = f_1_sat*exp(V_s_1*(P-P_1_sat)/(R*T));

//Since gas phase is ideal therefore
// y1*P = f_1
y1 = f_1/P;

//Number of moles of H2 in vapour phase at
    equilibrium remains the same as initial number of
     moles.
//Number of moles of CO2 in vapour phase at
    equilibrium can be calculated as
//y1 = (n_1_eq/(n_1_eq + n_2)). Therefore
n_1_eq = n_2*y1/(1-y1);

//Therefore moles of CO2 precipitated is
n_ppt = n_1 - n_1_eq;//[mol]

printf("The moles of CO2 precipitated is %f mol",
    n_ppt);
```

Scilab code Exa 16.14 Calculation of pressure

```
clear;
clc;

//Example - 16.14
//Page number - 586
printf("Example - 16.14 and Page number - 586\n\n");

//Given
R = 8.314;//[J/mol*K] - universal gas constant
```

```
T = 350;//[K] - Surrounding temperature

//componenet 1 : organic solid (1)
//componenet 2 : CO2 (2)

P_1_sat = 133.3;//[N/m^(2)] - Vapour pressre of
   organic solid
V_s_1 = 200;//[cm^(3)/mol] - Molar volume of organic
    solid
V_s_1 = V_s_1*10^(-6);//[m^(3)/mol]

///At 350 K, the values of the coefficients
B_11 = -500;//[cm^(3)/mol]
B_22 = -85;//[cm^^(3)/mol]
B_12 = -430;//[cm^(3)/mol]

//From phase equilibrium equation of component 1, we
    get
// y1*P*phi_1 = f_1
// f_1 = f_1_sat*exp(V_s_1*(P-P_1_sat)/(R*T))

//Since vapour pressure of organic solid is very
   small, therefore
f_1_sat = P_1_sat;

// Now let us determine the fugacity coefficient of
   organic solid in the vapour mixture.
// log(phi_1) = (P/(R*T))*(B_11 + y2^(2)*del_12)
del_12 = (2*B_12 - B_11 - B_22)*10^(-6);//[m^(3)/mol
   ]

//It is given that the partial pressure of component
    1 in the vapour mixture is 1333 N?m^(2)
// y1*P = 1333 N/m^(2) or, y1 = 1333/P
// y2 = 1- 1333/P
// log(phi_1) = (P/(R*T))*(B_11 + (1- 1333/P)^(2)*
   del_12)

```

```
//The phase equilibrium equation becomes
// y1*P*phi_1 = f_1_sat*exp(V_s_1*(P-P_1_sat)/(R*T))
//Taking log on both side we have
// log(y1*P) + log(phi_1) = log(f_1_sat) + (V_s_1*(P
   -P_1_sat)/(R*T))
// (V_s_1*(P-P_1_sat)/(R*T)) - log(phi_1) = log
   (1333/133.3) = log(10)

//substituting for log(phi_1) from previous into the
    above equation we get
// (V_s_1*(P-P_1_sat)/(R*T)) - (P/(R*T))*(B_11 + (1-
    1333/P)^(2)*del_12) - log(10) = 0
// On simplification we get,
// 975*P^(2) - 6.7*10^(9)*P + 4.89*10^(8) = 0
// Solving the above qyadratic equation using
   shreedharcharya rule

P3 = (6.7*10^(9) + ((-6.7*10^(9))^(2)
   -4*975*4.98*10^(8))^(1/2))/(2*975);//[Pa]
P4 = (6.7*10^(9) - ((-6.7*10^(9))^(2)
   -4*975*4.98*10^(8))^(1/2))/(2*975);//[Pa]
// The second value is not possible, therefore
   pressure of the system is P3
P3 = P3*10^(-5);//[bar]

printf(" The total pressure of the system is %f bar"
   ,P3);
```

Chapter 17

Chemical Reactions Equilibria

Scilab code Exa 17.1 Proving a mathematical relation

```
clear;
clc;

//Example - 17.1
//Page number - 595
printf("Example - 17.1 and Page number - 595\n\n");

//This problem involves proving a relation in which
   no mathematical components are involved.
//For prove refer to this example 17.1 on page
   number 595 of the book.
printf("This problem involves proving a relation in
   which no mathematical components are involved.\n\
   n");
printf("For prove refer to this example 17.1 on page
    number 595 of the book.")
```

Scilab code Exa 17.2 Determination of number of moles

```
clear;
clc;

//Example - 17.2
//Page number - 598
printf("Example - 17.2 and Page number - 598\n\n");

// Given
P = 1;//[atm] - Reactor pressure
T = 749;//[K] - Reactor temperature
K = 74;// Equlibrium constant

// SO2 + (1/2)*O2 - SO3

Kp = P^(1);
Ky = K/Kp;

//(1)
// Initial number of moles of the components are
n_SO2_1_in = 12;
n_O2_1_in = 9;
n_SO3_1_in = 0;

// Let the reaction coordinate at equilibrium for
   the reaction be X
// At equilibrium, the moles of the components be
// n_SO2_1_eq = 12 - X
// n_O2_1_eq = 9 - 0.5*X
// n_SO3_1_eq = X
// Total moles = 21 - 0.5*X

// The mole fractions of the components at
   equilibrium are
// y_SO3 = X/(21-0.5*X)
// y_SO2 = (12-X)/(21-0.5*X)
// y_O2 = (9-0.5*X)/(21-0.5*X)

// Ky = y_SO3/(y_SO2*y_O2^(2))
// Ky = (X*(21-0.5*X)^(1/2))/((12-X)*(9-0.5*X)^(1/2)
   )
deff('[y]=f(X)','y= Ky-(X*(21-0.5*X)^(1/2))/((12-X)
   *(9-0.5*X)^(1/2))');
X_1 = fsolve(11,f);

y_SO3_1 = X_1/(21-0.5*X_1);
y_SO2_1 = (12-X_1)/(21-0.5*X_1);
y_O2_1 = (9-0.5*X_1)/(21-0.5*X_1);

printf(" (1).The moles of SO3 formed = %f mol\n",X_1
   );
printf("      The mole fractions at equilibrium are
   y_S03 = %f, y_SO2 = %f and y_O2 = %f\n\n",y_SO3_1
   ,y_SO2_1,y_O2_1);

//(2)
// Initial number of moles of the components are
n_SO2_2_in = 24;
n_O2_2_in = 18;
n_SO3_2_in = 0;

// At equilibrium, the moles of the components be
// n_SO2_1_eq = 24 - X
// n_O2_1_eq = 18 - 0.5*X
// n_SO3_1_eq = X
// Total moles = 42 - 0.5*X

// The mole fractions of the components at
   equilibrium are
// y_SO3 = X/(42-0.5*X)
// y_SO2 = (24-X)/(42-0.5*X)
// y_O2 = (18-0.5*X)/(42-0.5*X)

// Ky = y_SO3/(y_SO2*y_O2^(2))
// Ky = (X*(42-0.5*X)^(1/2))/((24-X)*(18-0.5*X)
   ^(1/2))
deff('[y]=f1(X)','y= Ky-(X*(42-0.5*X)^(1/2))/((24-X)
    *(18-0.5*X)^(1/2))');
X_2 = fsolve(22,f1);

y_SO3_2 = X_2/(42-0.5*X_2);
y_SO2_2 = (24-X_2)/(42-0.5*X_2);
y_O2_2 = (18-0.5*X_2)/(42-0.5*X_2);
printf(" (2).The moles of SO3 formed = %f mol\n",X_2
    );
printf("      The mole fractions at equilibrium are
    y_SO3 = %f, y_SO2 = %f and y_O2 = %f\n\n",y_SO3_2
    ,y_SO2_2,y_O2_2);

//(3)
// Initial number of moles of the components are
n_SO2_3_in = 12;
n_O2_3_in = 9;
n_SO3_3_in = 0;
n_N2 = 79;

// At equilibrium, the moles of the components be
// n_SO2_1_eq = 12 - X
// n_O2_1_eq = 9 - 0.5*X
// n_SO3_1_eq = X
// Total moles = 100 - 0.5*X

// The mole fractions of the components at
    equilibrium are
// y_SO3 = X/(100-0.5*X)
// y_SO2 = (12-X)/(100-0.5*X)
// y_O2 = (9-0.5*X)/(100-0.5*X)

// Ky = y_SO3/(y_SO2*y_O2^(2))
// Ky = (X*(100-0.5*X)^(1/2))/((12-X)*(9-0.5*X)
    ^(1/2))
deff('[y]=f2(X)','y= Ky-(X*(100-0.5*X)^(1/2))/((12-X
    )*(9-0.5*X)^(1/2))');
X_3 = fsolve(10,f2);

```

```
y_SO3_3 = X_3/(100-0.5*X_3);
y_SO2_3 = (12-X_3)/(100-0.5*X_3);
y_O2_3 = (9-0.5*X_3)/(100-0.5*X_3);

printf(" (3).The moles of SO3 formed = %f mol\n",X_3
   );
printf("      The mole fractions at equilibrium are
   y_SO3 = %f, y_SO2 = %f and y_O2 = %f\n\n",y_SO3_3
   ,y_SO2_3,y_O2_3);
```

Scilab code Exa 17.3 Determination of equilibrium composition

```
clear;
clc;

//Example - 17.3
//Page number - 599
printf("Example - 17.3 and Page number - 599\n\n");

// Given
T = 600;//[K] -  Reactor temperature
P = 300;//[atm] - Reactor pressure
K = 0.91*10^(-4);// Equilibrium constant

// The fugacity coefficients of the components are
phi_CO = 1.0;
phi_H2 = 1.2;
phi_CH3OH = 0.47;

// CO + 2*H2 - CH3OH

// For gas phase reactions the standard state is
   pure ideal gas and thus fi_0 = 1 atm and thus
// ai_cap = fi_cap/fi_0 = yi*P*phi_i_cap/1
// Thus K = Ky*Kp*K_phi
```

```
Kp = P^(1-3);
K_phi = phi_CH3OH/(phi_CO*phi_H2^(2));
Ky = K/(Kp*K_phi);

// Let the reaction coordinate at equilibrium for
   the reaction be X
// At equilibrium ,the moles of the components be
// n_CO = 1 - X
// n_H2 = 3 - 2*X
// n_CH3OH = X
// Total moles = 4 - 2*X

// The mole fractions of the components at
   equilibrium are
// y_CO = (1-X)/(4-2*X)
// y_H2 = (3-2*X)/(4-2*X)
// y_CH3OH = (X)/(4-2*X)

// Ky = y_CH3OH/(y_CO*y_H2^(2)) = (X/(4-2*X))/(((1-X
   )/(4-2*X))*((3-2*X)/(4-2*X))^(2))
deff('[y]=f(X)','y=Ky-(X/(4-2*X))/(((1-X)/(4-2*X))
   *((3-2*X)/(4-2*X))^(2))');
X = fsolve(0.1,f);

// Therefore at equilibrium
y_CO = (1-X)/(4-2*X);
y_H2 = (3-2*X)/(4-2*X);
y_CH3OH = (X)/(4-2*X);

printf(" The mole fractions at equilibrium are y_CO
   = %f, y_H2 = %f and y_CH3OH = %f",y_CO,y_H2,
   y_CH3OH);
```

Scilab code Exa 17.4 Determination of the value of equilibrium constant

```
clear;
clc;

//Example - 17.4
//Page number - 600
printf("Example - 17.4 and Page number - 600\n\n");

// Given
T = 600;//[K] -  Reactor temperature
P = 4;//[atm] - Reactor pressure
K = 1.175;// Equilibrium constant

// (1/2)*N2 + (3/2)*H_2 - NH3

// Initial number of moles of the components are
n_N2 = 1;
n_H2 = 3;
n_HN3 = 0;

// Let the reaction coordinate at equilibrium for
   the reaction be X.
// At equilibrium ,the moles of the components be
// n_N2 = 1 - 0.5*X
// n_H2 = 3 - 1.5*X
// n_NH3 = X
// Total moles = 4 - X

// We have, K = Ky*Kp
Kp = P^(1-2);//[atm^(-1)]
Ky = K/(Kp);

// Ky = y_NH3/(y_N2^(1/2)*y_H2^(3/2)) = (X/(4-X))
   /(((1-0.5*X)/(4-X))^(1/2)*((3-1.5*X)/(4-X))^(3/2)
   )
// Solving the above equation we get
deff('[y]=f(X)','y=Ky - (X/(4-X))/(((1-0.5*X)/(4-X))
   ^(1/2)*((3-1.5*X)/(4-X))^(3/2))');
X = fsolve(0.1,f);
```

```

y_NH3 = X/(4-X);// Mole fraction of NH3 at
    equilibrium

printf(" The value of Kp = %f and Ky = %f \n",Kp,Ky)
    ;
printf(" The mole fractions of NH3 at equilibrium is
     %f\n\n",y_NH3);

// If reaction carried out at constant temperature
    and volume

// We know that for ideal gas, P*V = n*R*T and thus
    P is directly proportional to n at constant V and
     T.
// Let P = k*n
// Initially P = 4 atm and n = 4 moles, thus K = 1
    and we get p = n, where P is in atm.
// Thus at equilibrium P = 4 - X

// Ky = K/Kp = 1.175*P = 1.175*(4 - X)
// (X/(4-X))/(((1-0.5*X)/(4-X))^(1/2)*((3-1.5*X)/(4-
    X))^(3/2)) = 1.175*(4 - X)
// Solving the above equation we get
deff('[y]=f1(X)','y=(X/(4-X))/(((1-0.5*X)/(4-X))
    ^(1/2)*((3-1.5*X)/(4-X))^(3/2))-1.175*(4-X)');
X_prime = fsolve(1,f1);

// Therefore at equilibrium
P_prime = 4 - X_prime;
y_NH3_prime = X_prime/(4-X_prime);

printf(" If reaction is carried out at constant
    temperature and volume,then\n");
printf(" The equilibrium pressure is %f atm\n",
    P_prime);
printf(" The equilibrium mole fractions of NH3 in
    the reactor is %f\n\n",y_NH3_prime);
```

Scilab code Exa 17.5 Determination of mole fraction

```
clear;
clc;

//Example - 17.5
//Page number - 601
printf("Example - 17.5 and Page number - 601\n\n");

// Given
T = 400;//[K] -  Reactor temperature
P = 1;//[atm] - Reactor pressure
K = 1.52;// Equilibrium constant
y_H2 = 0.4;// Equilibrium mole fraction of hydrogen

// CO(g) + 2*H_2(g) - CH3OH(g)

// K = y_CH3OH/(y_CO*y_H2^(2)*P^(2))
// Let total number of moles at equilibrium be 1
// y_CH3OH = 0.6 - y_CO;
// (0.6 - y_CO)/y_CO = K*P^(2)*y_H2^(2)

y_CO = 0.6/(1 + K*P^(2)*y_H2^(2));
y_CH3OH = 0.6 - y_CO;

printf(" The mole fractions are, y_CO = %f and
   y_CH3OH = %f \n",y_CO,y_CH3OH);
```

Scilab code Exa 17.6 Determination of number of moles

```
clear;
```

```
clc;

//Example - 17.6
//Page number - 602
printf("Example - 17.6 and Page number - 602\n\n");

// Given
T = 749;//[K] - Reactor temperature
P = 1;//[atm] - Reactor pressure
K = 74;

Kp = P^(-1/2);//[atm^(-1/2)]
Ky = K/Kp;

// SO2 + (1/2)*O2 - SO3

// Initial number of moles of the components are
n_SO2_1_in = 10;
n_O2_1_in = 8;
n_SO3_1_in = 0;

// Let the reaction coordinate at equilibrium for
   the reaction be X
// At equilibrium, the moles of the components be
// n_SO2_1_eq = 10 - X
// n_O2_1_eq = 8 - 0.5*X
// SO3_1_eq = X
// Total moles = 18 - 0.5*X

// The mole fractions of the components at
   equilibrium are
// y_SO3 = X/(18-0.5*X)
// y_SO2 = (10-X)/(18-0.5*X)
// y_O2 = (8-0.5*X)/(18-0.5*X)

// Ky = y_SO3/(y_SO2*y_O2^(2))
// Ky = (X*(18-0.5*X)^(1/2))/((10-X)*(8-0.5*X)^(1/2)
   )
```

```
deff('[y]=f(X)','y= Ky-(X*(18-0.5*X)^(1/2))/((10-X)
    *(8-0.5*X)^(1/2))');
X_1 = fsolve(11,f);

n_SO3 = X_1;
n_SO2 = 10 - X_1;
n_O2 = 8 - 0.5*X_1;

printf(" (1).The moles of the components at
    equilibrium are, n_SO3 = %f mol, n_SO2 = %f mol
    and n_O2 = %f mol\n\n",n_SO3,n_SO2,n_O2);

// Now for the reaction
// 2*SO2 + O2 - 2*SO3

// The equilibrium constant for this reaction is KP
    ^(2)
Ky_prime = Ky^(2);

// At equilibrium, the moles of the components be
// n_SO2_1_eq = 10 - 2*X
// n_O2_1_eq = 8 - X
// SO3_1_eq = 2*X
// Total moles = 18 - X

// The mole fractions of the components at
    equilibrium are
// y_SO3 = 2*X/(18-X)
// y_SO2 = (10-2*X)/(18-X)
// y_O2 = (8- X)/(18-X)

// Ky_prime = y_SO3^(2)/(y_SO2^(2)*y_O2)
// Ky_prime = ((2*X)^(2)*(18-X))/((10-2*X)^(2)*(8-X)
    )
deff('[y]=f1(X)','y= Ky_prime-((2*X)^(2)*(18-X))
    /(((10-2*X)^(2))*(8-X))');
X_2 = fsolve(6,f1);

```

```
n_SO3_prime = 2*X_2;
n_SO2_prime = 10 - 2*X_2;
n_O2_prime = 8 - X_2;

printf(" (2).The moles of the components at
   equilibrium are, n_SO3 = %f mol, n_SO2 = %f mol
   and n_O2 = %f mol\n\n",n_SO3_prime,n_SO2_prime,
   n_O2_prime);
printf("     Thus the number of moles remains the
   same irrespective of the stoichoimetry of the
   reaction")
```

Scilab code Exa 17.7 Calculation of mole fraction

```
clear;
clc;

//Example - 17.7
//Page number - 603
printf("Example - 17.7 and Page number - 603\n\n");

// Given
T = 500;//[K]
// For the reaction,  0.5*A2 + 0.5*B2 - AB
delta_G = -4200;//[J/mol]
R = 8.314;//[J/mol*K] - Universal gas constant

//(1)
// A2 + B2 - 2*AB

// We know delta_G_rkn_0 = -R*T*log(K)
delta_G_1 = 2*delta_G;
K_1 = exp(-delta_G_1/(R*T));// Equilibrium constant
   at 500 K for the above reaction
// As can be seen the reaction is not affected by
```

```
    pressure and therefore K = Ky as Kp = 1
Ky = K_1;

// Initial number of moles of the components are
n_A2_1_in = 0.5;
n_B2_1_in = 0.5;
n_AB_1_in = 0;

// Let the reaction coordinate at equilibrium for
    the reaction be X
// At equilibrium , the moles of the components be
// n_A2_1_eq = 0.5 - X
// n_B2_1_eq = 0.5- X
// n_AB_1_eq = 2*X
// Total moles = 1

// Ky = (2*X)^(2)/(0.5-X)^(2)
deff('[y]=f(X)','y= Ky-(2*X)^(2)/(0.5-X)^(2)');
X_1 = fsolve(0.2,f);

// The mole fractions of the components at
    equilibrium are
y_A2_1 = 0.5 - X_1;
y_B2_1 = 0.5- X_1;
y_AB_1 = 2*X_1;

printf(" (1).The mole fractions at equilibrium are
    y_A2 = %f, y_B2 = %f and y_AB = %f\n\n",y_A2_1,
    y_B2_1,y_AB_1);

//(2)
// 0.5*A2 + 0.5*B2 - AB

// We know delta_G_rkn_0 = -R*T*log(K)
delta_G_2 = delta_G;
K_2 = exp(-delta_G_2/(R*T));// Equilibrium constant
    at 500 K for the above reaction

```

```
// As can be seen the reaction is not affected by
   pressure and therefore K = Ky as Kp = 1
Ky_2 = K_2;

// Initial number of moles of the components are
n_A2_2_in = 0.5;
n_B2_2_in = 0.5;
n_AB_2_in = 0;

// Let the reaction coordinate at equilibrium for
   the reaction be X
// At equilibrium , the moles of the components be
// n_A2_2_eq = 0.5 - 0.5*X
// n_B2_2_eq = 0.5- 0.5*X
// n_AB_2_eq = X
// Total moles = 1

// Ky = y_AB/(y_A2^(1/2)*y_B2^(1/2))
// Ky = X/(0.5 - 0.5*X)
X_2 = 0.5*Ky_2/(1+0.5*Ky_2);

// The mole fractions of the components at
   equilibrium are
y_A2_2 = 0.5 - 0.5*X_2;
y_B2_2 = 0.5- 0.5*X_2;
y_AB_2 = X_2;

printf(" (2).The mole fractions at equilibrium are
   y_A2 = %f, y_B2 = %f and y_AB = %f\n\n",y_A2_2,
   y_B2_2,y_AB_2);

//(3)
// 2*AB - A2 + B2

K_3 = 1/K_1;// Equilibrium constant at 500 K for the
    above reaction
// As can be seen the reaction is not affected by
   pressure and therefore K = Ky as Kp = 1
```

```
Ky_3 = K_3;

// Initial number of moles of the components are
n_AB_3_in = 1;
n_A2_3_in = 0;
n_B2_3_in = 0;

// Let the reaction coordinate at equilibrium for
   the reaction be X
// At equilibrium, the moles of the components be
// n_AB_3_eq = 1 - X
// n_A2_3_eq = X/2
// n_B2_3_eq = X/2
// Total moles = 1

// Ky = (X/2)^(2)/(1-X)^(2)
deff('[y]=f1(X)','y= Ky_3-(X/2)^(2)/(1-X)^(2)');
X_3 = fsolve(0.4,f1);

// The mole fractions of the components at
   equilibrium are
y_A2_3 = X_3/2;
y_B2_3 = X_3/2;
y_AB_3 = 1-X_3;

printf(" (3).The mole fractions at equilibrium are
   y_A2 = %f, y_B2 = %f and y_AB = %f\n\n",y_A2_3,
   y_B2_3,y_AB_3);
```

Scilab code Exa 17.8 Calculation of heat exchange

```
clear;
clc;

//Example - 17.8
```

```
//Page number - 606
printf("Example - 17.8 and Page number - 606\n\n");

// Given
// P*P +q*Q - r*R + s*S
// Let Cp_P = p, Cp_Q = Q, Cp_R = R and Cp_S = S

//(1)
// When reactants are heated from 10 to 25 C,
   reaction takes place at 25 C and products are
   raised from 25 C to 1500 K the heat exchange is
   given by
T_1 = 10 + 273.15;//[K]
T_2 = 25 + 273.15;//[K]
T_3 = 1500;//[K]
// Q = integrate('(p*Cp_P + q*Cp_q)*dT','T',T_1,T_2)
    + delta_H_rkn_298 + integrate('(r*Cp_R + s*Cp_S)
   *dT','T'T_2,T_3);
printf(" (1).The expression for the heat exchange
   with the surrounding by the first path is given
   below\n");
printf("     Q = integrate((p*Cp_P + q*Cp_q)*dT,T,
   T_1,T_2) + delta_H_rkn_298 + integrate((r*Cp_R +
   s*Cp_S)*dT,TT_2,T_3);\n\n")

//(2)
// When reactants are heated from 10 C to 1500 K,
   reaction take place at 1500 K the heat exchange
   is given by
// Q = integrate('(p*Cp_P + q*Cp_q)*dT','T',T_1,T_3)
    + delta_H_rkn_1500
// where, delta_H_rkn_1500 = delta_H_rkn_298 +
   integrate('(r*Cp_R + s*Cp_S - p*Cp_P - q*Cp_q)*dT
   ','T'T_2,T_3);
// Therefore
// Q = integrate('(p*Cp_P + q*Cp_q)*dT','T',T_1,T_3)
    + delta_H_rkn_298 + integrate('(r*Cp_R + s*Cp_S
   - p*Cp_P - q*Cp_q)*dT','T'T_2,T_3);
```

```
// Q = integrate('(p*Cp_P + q*Cp_q)*dT','T',T_1,T_2)
   + delta_H_rkn_298 + integrate('(r*Cp_R + s*Cp_S)
  *dT','T'T_2,T_3);

// which is same as in method (1). The total
  enthalpy change between two fixed ponts is a
  function only of state and not the path taken.

printf(" (2).The expression for the heat exchange
  with the surrounding by the second path is given
  below\n");
printf("     Q = integrate((p*Cp_P + q*Cp_q)*dT,T,
  T_1,T_2) + delta_H_rkn_298 + integrate((r*Cp_R +
  s*Cp_S)*dT,TT_2,T_3);\n\n")
```

Scilab code Exa 17.9 Dtermination of heat of reaction

```
clear;
clc;

//Example - 17.9
//Page number - 606
printf("Example - 17.9 and Page number - 606\n\n");

// Given
// SO2 + (1/2)*O2 - SO3
R = 1.987;//[cal/mol-K]

delta_H_SO2_298 = -70.96;//[kcal/mol] - Enthalpy of
  formation of S02 at 298.15 K
delta_H_SO3_298 = -94.45;//[kcal/mol] - Enthalpy of
  formation of S03 at 298.15 K
delta_G_SO2_298 = -71.79;//[kcal/mol] - Gibbs free
  energy change for formation of SO2 at 298.15 K
delta_G_SO3_298 = -88.52;//[kcal/mol] - Gibbs free
```

```
   energy change for formation of SO3 at 298.15 K

// Cp_0 = a + b*T + c*T^(2) + d*T^(3)

a_SO2 = 6.157;
a_SO3 = 3.918;
a_O2 = 6.085;
b_SO2 = 1.384*10^(-2);
b_SO3 = 3.483*10^(-2);
b_O2 = 0.3631*10^(-2);
c_SO2 = -0.9103*10^(-5);
c_SO3 = -2.675*10^(-5);
c_O2 = -0.1709*10^(-5);
d_SO2 = 2.057*10^(-9);
d_SO3 = 7.744*10^(-9);
d_O2 = 0.3133*10^(-9);

//(1)
T_1 = 298.15;//[K]

delta_H_rkn_298 = delta_H_SO3_298 - delta_H_SO2_298;
   //[kcal]
delta_H_rkn_298 = delta_H_rkn_298*10^(3);//[cal]
delta_G_rkn_298 = delta_G_SO3_298 - delta_G_SO2_298;
   //[kcal]
delta_G_rkn_298 = delta_G_rkn_298*10^(3);//[cal]

delta_a = a_SO3 - a_SO2 - (a_O2/2);
delta_b = b_SO3 - b_SO2 - (b_O2/2);
delta_c = c_SO3 - c_SO2 - (c_O2/2);
delta_d = d_SO3 - d_SO2 - (d_O2/2);

// delta_H_rkn_T = delta_H_rkn_298 + integrate('
   delta_a+(delta_b*T)+(delta_c*T^(2))+(delta_d*T
   ^(3))','T',T_1,T);
// On simplification we get
// delta_H_rkn_T = -22630.14 - 5.2815*T +
   0.9587*10^(-2)*T^(2) - 0.5598*10^(-5)*T^(3) +
    1.3826*10^(-9)*T^(4)

printf(" (1).The expression for delta_H_rkn_T as a
    function of T is given by\n");
printf("     delta_H_rkn_T = -22630.14 - 5.2815*T +
    0.9587*10^(-2)*T^(2) - 0.5598*10^(-5)*T^(3) +
    1.3826*10^(-9)*T^(4)\n\n");

//(2)
// R*log(K_T/K_298) = integrate('delta_H_rkn_T/T^(2)
    ',T,T_1,T)
// First let us calculate K_298.
// delta_G_rkn_T = - R*T*log(K)
K_298 = exp(-delta_G_rkn_298/(R*T_1));

// On substituting the values and simplifying we get
     the expression
// log(K) = 3.87 + 11380.10/T - 2.6580*log(T) +
    0.4825*10^(-2)*T - 0.1409*10^(-5)*T^(2) +
    0.2320*10^(-9)*T^(3)

printf(" (2).The expression for log(K) as a function
     of T is given by\n");
printf("     log(K) = 3.87 + 11380.10/T - 2.6580*log
    (T) + 0.4825*10^(-2)*T - 0.1409*10^(-5)*T^(2) +
    0.2320*10^(-9)*T^(3)\n\n");

//(3)
P = 1;//[atm]
T = 880;//[K]
K = exp(3.87 + 11380.10/T - 2.6580*log(T) +
    0.4825*10^(-2)*T - 0.1409*10^(-5)*T^(2) +
    0.2320*10^(-9)*T^(3));
Kp = P^(-1/2);//[atm^(-1/2)]
Ky = K/Kp;

// Let the reaction coordinate at equilibrium for
    the reaction be X
```

```
// At equilibrium , the moles of the components be
// n_SO2_eq = 1 - X
// n_O2_eq = 0.5- 0.5*X
// n_SO3_1_eq = X
// Total moles = 1.5-0.5*X

// Ky = (X*(1.5-0.5*X)^(1/2))/((1-X)*(0.5-0.5*X)
   ^(1/2))
deff('[y]=f(X)','y= Ky - (X*(1.5-0.5*X)^(1/2))/((1-X
   )*(0.5-0.5*X)^(1/2))');
X = fsolve(0.8,f);

// The mole fraction of SO3 at equilibrium is given
   by
y_SO3 = X/(1.5-0.5*X);

printf(" (3).The mole fraction of SO3 at equilibrium
    is given by, y_SO3 = %f\n",y_SO3);
```

Scilab code Exa 17.10 Tabulation of equilibrium constant values

```
clear;
clc;

//Example - 17.10
//Page number - 609
printf("Example - 17.10 and Page number - 606\n\n");

// Given
// (1/2)*N2 + (1/2)*O2 - NO

R = 1.987;//[cal/mol-K]

delta_H_NO_298 = 21.600;//[kcal/mol] - Enthalpy of
   formation of S02 at 298.15 K
```

```
delta_G_NO_298 = 20.719;//[kcal/mol] - Gibbs free
   energy change for formation of SO2 at 298.15 K

// Cp_0 = a + b*T + c*T^(2) + d*T^(3)

a_N2 = 6.157;
a_O2 = 6.085;
a_NO = 6.461;
b_N2 = -0.03753*10^(-2);
b_O2 = 0.3631*10^(-2);
b_NO = 0.2358*10^(-2);
c_N2 = 0.1930*10^(-5);
c_O2 = -0.1709*10^(-5);
c_NO = -0.07705*10^(-5);
d_N2 = -0.6861*10^(-9);
d_O2 = 0.3133*10^(-9);
d_NO = 0.08729*10^(-9);

//(1)
T_1 = 298.15;//[K]

delta_H_rkn_298 = delta_H_NO_298;//[kcal]
delta_H_rkn_298 = delta_H_rkn_298*10^(3);//[cal]
delta_G_rkn_298 = delta_G_NO_298;//[kcal]
delta_G_rkn_298 = delta_G_rkn_298*10^(3);//[cal]

delta_a = a_NO - (a_N2/2) - (a_O2/2);
delta_b = b_NO - (b_N2/2) - (b_O2/2);
delta_c = c_NO - (c_N2/2) - (c_O2/2);
delta_d = d_NO - (d_N2/2) - (d_O2/2);

// delta_H_rkn_T = delta_H_rkn_298 + integrate('
   delta_a+(delta_b*T)+(delta_c*T^(2))+(delta_d*T
   ^(3))','T',T_1,T);
// On simplification we get
// delta_H_rkn_T = 21584.63 - 0.033*T +
   0.0365*10^(-2)*T^(2) - 0.0293*10^(-5)*T^(3) +
   0.0685*10^(-9)*T^(4)
```

```

printf(" The expression for delta_H_rkn_T as a
   function of T is given by\n");
printf(" delta_H_rkn_T = 21584.63 - 0.033*T +
   0.0365*10^(-2)*T^(2) - 0.0293*10^(-5)*T^(3) +
   0.0685*10^(-9)*T^(4)\n\n");

// Now let us calculate K_298 (at 298 K)
// We know delta_G_rkn_298 = -R*T*log(K_298)
K_298 = exp(-delta_G_rkn_298/(R*T_1));// Equilibrium
    constant at 298.15 K

// log(K_2/K_1) = integrate('delta_H_rkn_298/(R*T
   ^(2))','T',T_1,T)
// On substituting the values and simplifying we get
    the expression
// log(K) = 1.5103 - 10862.92/T - 0.0166*log(T) +
   1.84*10^(-4)*T - 7.35*10^(-8)*T^(2) +
   1.15*10^(-11)*T^(3)

printf(" The expression for log(K) as a function of
   T is given by\n");
printf(" log(K) = 1.5103 - 10862.92/T - 0.0166*log(T
   ) + 1.84*10^(-4)*T - 7.35*10^(-8)*T^(2) +
   1.15*10^(-11)*T^(3)\n\n\n\n")

T = [500,1000,1500,2000,2500];
K = zeros(5);

printf(" T (K)  \t\t\t  K  \n\n");

for i=1:5;
    K(i) = exp(1.5103-10862.92/T(i) - 0.0166*log(T(i
       )) + 1.84*10^(-4)*T(i) - 7.35*10^(-8)*T(i)
       ^(2) + 1.15*10^(-11)*T(i)^(3));

    printf(" %f  \t\t  %e  \n\n",T(i),K(i));
end
```

```

printf("\n\n");

// It can be seen that for endothermic reactions
   equilibrium constant increases with temperature.
printf(" It can be seen that for endothermic
   reactions equilibrium constant increases with
   temperature\n");
```

Scilab code Exa 17.11 Determination of mean standard enthalpy of reaction

```
clear;
clc;

//Example - 17.11
//Page number - 611
printf("Example - 17.11 and Page number - 611\n\n");

// Given
// SO2 + (1/2)*O2 - SO3
R = 8.314;//[J/mol-K]

K_800 = 0.0319;// Equilibrium constant at 800 K
K_900 = 0.153;// Equilibrium constant at 900 K
T_1 = 800;//[K]
T_2 = 900;//[K]

// We have the relation
// log(K_2/K_1) = -(delta_H_rkn/R)*(1/T_2 - 1/T_1)
// log(K_900/K_800) = -(delta_H_rkn_850/R)*(1/T_2 -
   1/T_1)
delta_H_rkn_850 = -R*log(K_900/K_800)/(1/T_2 - 1/T_1
   );//[J]
delta_H_rkn_850 = delta_H_rkn_850*10^(-3);//[kJ]
```

```

printf(" The mean standard enthalpy change of
   reaction in the region 800 t0 900 is given by
   delta_H_rkn_850 = %f \n",delta_H_rkn_850);
```

Scilab code Exa 17.12 Derivation of expression for enthalpy of reaction

```
clear;
clc;

//Example - 17.12
//Page number - 611
printf("Example - 17.12 and Page number - 611\n\n");

// Given
// log(K) = -2.61 + 23672.62/T - 7.17*log(T) +
   1.24*10^(-2)*T - 0.148*10^(-5)*T^(2) +
   0.039*10^(-9)*T^(3)

// We know that
// dlog(K)/dT = delta_H_rkn/(R*T^(2))

// Differentiating the above expression of log(K),we
    get
// dlog(K)/dT = -23672.62/T^(2) - 7.17/T +
   1.24*10^(-2) - 2*0.148*10^(-5)*T +
   3*0.039*10^(-9)*T^(2)

// On further simplification we get
// delta_H_rkn = -47037.5 - 14.24*T + 2.46*10^(-2)*T
   ^(2) - 0.59*10^(-5)*T^(3) + 0.23*10^(-9)*T^(4)
printf(" delta_H_rkn = -47037.5 - 14.24*T +
   2.46*10^(-2)*T^(2) - 0.59*10^(-5)*T^(3) +
   0.23*10^(-9)*T^(4)\n");
```

Scilab code Exa 17.13 Determination of equilibrium composition

```
clear;
clc;

//Example - 17.13
//Page number - 611
printf("Example - 17.13 and Page number - 611\n\n");

// Given
T_1 = 298.15;//[K]
T = 2600;//[K]
R = 1.987;//[cal/mol-K] - Universal gas constant

// Cp_0 = a + b*T + c*T^(2) + d*T^(3)
delta_H_CO_298 = -26.416;//[kcal/mol] - Enthalpy of
   formation of CO at 298.15 K
delta_G_CO_298 = -32.808;//[kcal/mol] - Gibbs free
   energy change for formation of CO at 298.15 K
delta_H_CO2_298 = -94.052;//[kcal/mol] - Enthalpy of
    formation of C02 at 298.15 K
delta_G_CO2_298 = -94.260;//[kcal/mol] - Gibbs free
   energy change for formation of CO2 at 298.15 K

// CO + (1/2)*O2 - CO2

a_CO = 6.726;
a_O2 = 6.0685;
a_CO2 = 5.316;
b_CO = 0.04001*10^(-2);
b_O2 = 0.3631*10^(-2);
b_CO2 = 1.4285*10^(-2);
c_CO = 0.1283*10^(-5);
c_O2 = -0.1709*10^(-5);
```

```
c_CO2 = -0.8362*10^(-5);
d_CO = -0.5307*10^(-9);
d_O2 = 0.3133*10^(-9);
d_CO2 = 1.784*10^(-9);


delta_H_rkn_298 = delta_H_CO2_298 - delta_H_CO_298;
    //[kcal]
delta_H_rkn_298 = delta_H_rkn_298*10^(3);//[cal]
delta_G_rkn_298 = delta_G_CO2_298 - delta_G_CO_298;
    //[kcal]
delta_G_rkn_298 = delta_G_rkn_298*10^(3);//[cal]

delta_a = a_CO2 - (a_CO) - (a_O2/2);
delta_b = b_CO2 - (b_CO) - (b_O2/2);
delta_c = c_CO2 - (c_CO) - (c_O2/2);
delta_d = d_CO2 - (d_CO) - (d_O2/2);

// delta_H_rkn_T = delta_H_rkn_298 + integrate('
    delta_a+(delta_b*T)+(delta_c*T^(2))+(delta_d*T
    ^(3))','T',T_1,T);
// On simplification we get
delta_H_rkn_T = -66773.56 - 4.45*T + 0.605*10^(-2)*T
    ^(2) - 0.29*10^(-5)*T^(3) + 0.54*10^(-9)*T^(4);

// log(K/K_298) = integrate('delta_H_rkn_T/(R*T^(2))
    ','T',T_1,T)

// We know that  delta_G_rkn_T = -R*T*log(K)
// At 298.15 K
K_298 = exp(-delta_G_rkn_298/(R*T_1) );

// Therfore on simplification we get
//log(K) = 2.94 + 33605.2/T - 2.24*log(T) +
    0.304*10(-2)*T - 0.073*10^(-5)*T^(2) +
    0.09*10^(-9)*T^(3)
K = exp(2.94 + 33605.2/T - 2.24*log(T) +
    0.304*10^(-2)*T - 0.073*10^(-5)*T^(2) +
    0.09*10^(-9)*T^(3));

printf("  The value of equilibrium constant at 2600
    K is given by, K_298 = %f\n\n",K);


//(a)
P_1 = 1;//[atm]
Kp_1 = P_1^(-1/2);
Ky_1 = K/Kp_1;

// Let the reaction coordinate at equilibrium for
    the reaction be X
// At equilibrium, the moles of the components be
// n_CO_1_eq = 1 - X
// n_02_1_eq = 0.5- 0.5X
// n_CO2_1_eq = X
// Total moles = 1.5 - 0.5*X

// Ky = y_CO2/(y_CO^(1/2)*y_CO)
//ky = (X*(1.5-0.5*X)^(1/2))/((1-X)*(0.5-0.5*X)
    ^(1/2))

deff('[y]=f(X)','y= Ky_1-(X*(1.5-0.5*X)^(1/2))/((1-X
    )*(0.5-0.5*X)^(1/2))');
X_1 = fsolve(0.9,f);

y_CO2_1 = X_1/(1.5-0.5*X_1);
y_CO_1 = (1-X_1)/(1.5-0.5*X_1);
y_O2_1 = (0.5-0.5*X_1)/(1.5-0.5*X_1);

printf(" (a).The equilibrium composition (at 1 atm)
    is given by, y_CO2 = %f, y_CO = %f and y_O2 = %f\
    n\n",y_CO2_1,y_CO_1,y_O2_1);

//(b)
P_2 = 10;//[atm]
Kp_2 = P_2^(-1/2);
```

```
Ky_2 = K/Kp_2;

// Ky = y_CO2/(y_CO^(1/2)*y_CO)
//ky = (X*(1.5-0.5*X)^(1/2))/((1-X)*(0.5-0.5*X)
    ^(1/2))

deff('[y]=f1(X)','y= Ky_2-(X*(1.5-0.5*X)^(1/2))/((1-
   X)*(0.5-0.5*X)^(1/2))');
X_2 = fsolve(0.9,f1);

y_CO2_2 = X_2/(1.5-0.5*X_2);
y_CO_2 = (1-X_2)/(1.5-0.5*X_2);
y_O2_2 = (0.5-0.5*X_2)/(1.5-0.5*X_2);

printf(" (b).The equilibrium composition (at 10 atm)
     is given by, y_CO2 = %f, y_CO = %f and y_O2 = %f
    \n\n",y_CO2_2,y_CO_2,y_O2_2);

//(c)
P_3 = 1;//[atm]
Kp_3 = P_3^(-1/2);
Ky_3 = K/Kp_3;

// Ky = y_CO2/(y_CO^(1/2)*y_CO)
//ky = (X*(1.5-0.5*X)^(1/2))/((1-X)*(0.5-0.5*X)
    ^(1/2))

// At equilibrium, the moles of the components be
// n_CO_3_eq = 1 - X
// n_02_3_eq = 0.5- 0.5X
// n_CO2_3_eq = X
// n_N2_eq = 1;
// Total moles = 2.5 - 0.5*X

deff('[y]=f2(X)','y= Ky_3-(X*(2.5-0.5*X)^(1/2))/((1-
   X)*(0.5-0.5*X)^(1/2))');
X_3 = fsolve(0.9,f2);

```

```
y_CO2_3 = X_3/(2.5-0.5*X_3);
y_CO_3 = (1-X_3)/(2.5-0.5*X_3);
y_O2_3 = (0.5-0.5*X_3)/(2.5-0.5*X_3);
y_N2 = 1/(2.5-0.5*X_3);

printf(" (c).The equilibrium composition (at 1 atm
   and 1 mol N2) is given by, y_CO2 = %f, y_CO = %f
   , y_O2 = %f and y_N2 = %f\n\n",y_CO2_3,y_CO_3,
   y_O2_3,y_N2);
```

Scilab code Exa 17.14 Determination of equilibrium composition

```
clear;
clc;

//Example - 17.14
//Page number - 614
printf("Example - 17.14 and Page number - 614\n\n");

// Given
T = 25 + 298.15;//[K] - Temperature
R = 8.314;//[J/mol-K]
delta_G_298 = -1000;//[J] - Gibbs free energy change
    at 298 K

// G_E/(R*T) = x_1*x_2

// We know that  delta_G_rkn = - R*T*log(K),
   therefore
K = exp(-delta_G_298/(R*T));

//(1)
// Let x_1 is the mole fraction of A and x_2 be the
   mole fraction of B
// If A and B form an ideal mixture then,
Ky = 1;
// and K = Kx = x_2/x_1
// and at equilibrium x_2/x_1 = K
// (1-x_1)/x_1 = K
x_1 = 1/(1+K);

printf(" (1).The equilibrium composition (for ideal
   behaviour) is given by x_1 = %f\n\n",x_1);

//(2)
// The activity coefficients are given by
// log(Y1) = [del(n*G_E/(R*T))/del(n_1)]_T,P,n_2 =
   x_2^(2)
// similarly, log(Y2) = x_1^(2)

// The equilibrium constant for the liquid phase
   reaction is given by
// K = Kx*Ky = (x_2*Y2)/(x_1*Y1) = (x_2*exp(x_1^(2))
   )/(x_1*exp(x_2^(2)))
// Solving the above equation we get

deff('[y]=f2(x_1)','y= K -((1-x_1)*exp(x_1^(2)))/(
   x_1*exp((1-x_1)^(2)))');
x_1_prime = fsolve(0.9,f2);

printf(" (2).The equilibrium composition (for non-
   ideal behaviour) is given by x_1 = %f\n\n",
   x_1_prime);
```

Scilab code Exa 17.15 Determination of the

```
clear;
clc;

//Example - 17.15
```

```
//Page number - 615
printf("Example - 17.15 and Page number - 615\n\n");

// Given
T_1 = 298.15;//[K] - Standard reaction temperature
T_2 = 373;//[K]  - Reaction temperature
P = 1;//[atm]
R = 1.987;//[cal/mol-K] - Universal gas constant

// CH3COOH (l) + C2H5OH (l) - CH3COOC2H5 (l) + H2O (
   l)

delta_H_CH3COOH_298 = -116.2*10^(3);// [cal/mol]
delta_H_C2H5OH_298 = -66.35*10^(3);// [cal/mol]
delta_H_CH3COOC2H5_298 = -110.72*10^(3);// [cal/mol]
delta_H_H2O_298 = -68.3174*10^(3);// [cal/mol]

delta_G_CH3COOH_298 = -93.56*10^(3);// [cal/mol]
delta_G_C2H5OH_298 = -41.76*10^(3);// [cal/mol]
delta_G_CH3COOC2H5_298 = -76.11*10^(3);// [cal/mol]
delta_G_H2O_298 = -56.6899*10^(3);// [cal/mol]

delta_H_rkn_298 = delta_H_CH3COOC2H5_298 +
   delta_H_H2O_298 - delta_H_CH3COOH_298 -
   delta_H_C2H5OH_298;//[cal/mol]
delta_G_rkn_298 = delta_G_CH3COOC2H5_298 +
   delta_G_H2O_298 - delta_G_CH3COOH_298 -
   delta_G_C2H5OH_298;//[cal/mol]

// We know that  delta_G_rkn_T = -R*T*log(K)
// At 298.15 K
K_298 = exp(-delta_G_rkn_298/(R*T_1) );

// We know that dlog(K)/dT = delta_H_rkn/(R*T^(2))
// If delta_H_rkn is assumed constant we get
// log(K_2/K_1) = (-delta_H_rkn/R)*(1/T_2 - 1/T_1)
// log(K_373/K_298) = (-delta_H_rkn_298/R)*(1/T_2 -
   1/T_1)
```

```

K_373 = exp(log(K_298) + (-delta_H_rkn_298/R)*(1/T_2
    - 1/T_1));
// Note that the equilibrium constant value rises
    becauses the reaction is endothermic

printf(" The value of equilibrium constant at 373 K
    is , K_373 = %f\n\n",K_373);

// Let the reaction coordinate at equilibrium for
    the reaction be X
// At equilibrium , the moles of the components be
// n_CH3COOH = 1 - X
// n_C2H5OH = 1 - X
// n_CH3COOC2H5 = X
// n_H20 = X
// Total moles = 2

// Kx = (x_CH3COOH*x_C2H5OH)/(x_CH3COOC2H5*x_H2O)
// Assuming the liquid mixture to be ideal ,that is
    Ky = 1, therefore K_x = K
K_x = K_373;
// X^(2)/(1-X)^(2) = K_x
X = (K_x)^(1/2)/(1+(K_x)^(1/2));

// The mole fraction of ethyl acetate is given by
x_CH3COOC2H5 = X/2;

printf(" The mole fraction of ethyl acetate in the
    equilibrium reaction mixture is given by,
    x_CH3COOC2H5 = %f\n",x_CH3COOC2H5);
```

Scilab code Exa 17.16 Calculation of the value of Gibbs free energy

```
clear;
```

```
clc;

//Example - 17.16
//Page number - 617
printf("Example - 17.16 and Page number - 617\n\n");

// Given
// CaCO3 (s1) - CaO (s2) + CO2 (g)
T_1 = 898 + 273.15;//[K]
T_2 = 700 + 273.15;//[K]
R = 8.314;//[J/mol-K] - Universal gas constant

P_CO2_T_1 = 1;//[atm] - Decomposition pressure of
    CO2 over CaCO3 at 898 C
P_CO2_T_2 = 0.0333;//[atm] - Decomposition pressure
    of  CO2 over CaCO3 at 700 C

// The equilibrium constant of the reaction is given
    by
// K = (a_CO2*a_CaO)/a_CaCO3

// Since the pressure is small therefore carbon
    dioxide can be assumed to behave as ideal gas and
    thus
// a_CO2 = y_CO2*P/1 = P_CO2

// The activity of CaO is (CaO is pure)
// a_CaO = f_CaO/f_0_CaO = exp[V_CaO*(P - P_0)/(R*T)
    ] = 1  (since pressure is low)

// The activity of CaCO3 is (CaCO3 is pure)
// a_CaCO3 = f_CaCO3/f_0_CaCO3 = exp[V_CaCO3*(P -
    P_0)/(R*T)] = 1  (since pressure is low)

//Since pressures are around 1 atm,therefore
    Poynting factors are also near 1, and thus
    activity of CaO and CaCO3 is unity and
    equilibrium constant is given by
```

```
//K = P_CO2 , therefore

// At 898 C
K_T_1 = P_CO2_T_1;
delta_G_T_1 = -R*T_1*log(K_T_1);

// At 700 C
K_T_2 = P_CO2_T_2;
delta_G_T_2 = -R*T_2*log(K_T_2);

printf(" The value of delta_G_rkn at 700 C is %f J\n
   \n",delta_G_T_1);
printf(" The value of delta_G_rkn at 898 C is %f J\n
   \n",delta_G_T_2);
```

Scilab code Exa 17.17 Calculation of nu

```
clear;
clc;

//Example - 17.17
//Page number - 618
printf("Example - 17.17 and Page number - 618\n\n");

// Given
T = 700 + 273.15;//[K]
K = 7.403;// Equilibrium constant for the reaction
   at 700 C

// CH4 - C (s) + 2*H2

// The equilibrium constant of the reaction is given
    by
// K = (a_C*a_H2^(2))/a_CH4

// Since carbon is pure therefore its activity is
   given by
// a_C = f/f_0 = 1 as pressure is 1 atm.
// Since the pressure is low,therefore the gas phase
    can be taken to be ideal,therefore
// K = (y_H2^(2)*P^(2))/(y_CH4*P) = y_H2^(2)/y_CH4
       (as P = 1 atm)
Ky = K;// (Kp = 1 atm)

// Initial moles of the species are
n_CH4 = 1;
n_H2 = 0;
n_C = 0;

// Let the reaction coordinate at equilibrium for
   the reaction be X
// Moles at equilibrium be
// n_CH4_eq = 1 -X
// n_H2_eq = 2*x
// n_C_eq = X

// Moles present in gas phase
// n_CH4_gas = 1 -X
// n_H2_gas = 2*x
// Total moles = 1 + X

// gas phase mole fraction
// y_CH4_gas = (1 -X)/(1+X)
// y_H2_gas = 2*x/(1+X)

// Ky = y_H2_gas^(2)/y_CH4_gaS
X = (K/(4+K))^(1/2);

printf(" The number of moles of carbon black formed
   from one mole of methane  is %f mol\n\n",X);

// Therefore mole fraction of components in the gas
   phase at equilibrium is
```

```
y_CH4 = (1-X)/(1+X);
y_H2 = 2*X/(1+X);

printf(" The mole fraction of components in the gas
    phase at equilibrium is given by y_CH4 = %f and
    y_H2 = %f \n",y_CH4,y_H2);
```

Scilab code Exa 17.18 Determination of value of the equilibrium constant

```
clear;
clc;

//Example - 17.18
//Page number - 619
printf("Example - 17.18 and Page number - 619\n\n");

// Given
T_1 = 298.15;//[K] - Standard reaction temperature
T_2 = 1042;//[K] - Reaction temperature
R = 1.987;//[cal/mol-K] - Universal gas constant

// CaCO3 (s1) - CaO (s2) + CO2 (g)

delta_H_CaCO3_298 = -289.5;//[kcal/mol] - Enthalpy
    of formation of CaCO3 at 298.15 K
delta_H_CaO_298 = -151.7;//[kcal/mol] - Enthalpy of
    formation of CaO at 298.15 K
delta_H_CO2_298 = -94.052;//[kcal/mol] - Enthalpy of
     formation of CO2 at 298.15 K
delta_G_CaCO3_298 = -270.8;//[kcal/mol] - Gibbs free
     energy change for formation of CaCO3 at 298.15 K
delta_G_CaO_298 = -144.3;//[kcal/mol] - Gibbs free
    energy change for formation of CaO at 298.15 K
delta_G_CO2_298 = -94.260;//[kcal/mol] - Gibbs free
    energy change for formation of CO2 at 298.15 K
```

```

// The standard heat capacity data as a function of
    temperature are given below
// Cp_CO2 = 5.316 + 1.4285*10^(2)*T - 0.8362*10^(-5)
    *T^(2) + 1.784*10^(-9)*T^(3)
// Cp_CaO = 12.129 + 0.88*10^(-3)*T - 2.08*10^(5)*T
    ^(-2)
// Cp_CaCO3 = 24.98 + 5.240*10^(-3)*T - 6.199*10^(5)
    *T^(-2)
// Therefore Cp_0 is given by
// Cp_0 = Cp_CO2 + Cp_CaO - Cp_CaCO3
// Cp_0 = -7.535 + 9.925*10^(-3)*T - 0.8362*10^(-5)*
   T^(2) + 1.784*10^(-9)*T^(3) + 4.119*10^(5)*T^(-2)

// The standard enthalpy change of the reaction at
    298.15 K is given by
delta_H_rkn_298 = delta_H_CO2_298 + delta_H_CaO_298
    - delta_H_CaCO3_298;//[kcal]
delta_H_rkn_298 = delta_H_rkn_298*10^(3);//[cal]
delta_G_rkn_298 = delta_G_CO2_298 + delta_G_CaO_298
    - delta_G_CaCO3_298;//[kcal]
delta_G_rkn_298 = delta_G_rkn_298*10^(3);//[cal]

// The standard enthalpy change of the reaction at
    temperature T is given by
// delta_H_rkn_T = delta_H_rkn_298 + integrate
    ('-7.535 + 9.925*10^(-3)*T - 0.8362*10^(-5)*T^(2)
     + 1.784*10^(-9)*T^(3) + 4.119*10^(5)*T^(-2)','T
    ',T_1,T);
// On simplification we get
// delta_H_rkn_T = 47005.3 - 7.535*T +
    4.9625*10^(-3)*T^(2) - 0.2787*10^(-5)*T^(3) +
    0.446*10^(-9)*T^(4) - 4.119*10^(5)*T^(-1)


// log(K_2/K_1) = integrate('delta_H_rkn_T/(R*T^(2))
    ','T',T_1,T)
log_K2_K1 = integrate('(47005.3-7.535*T
    +4.9625*10^(-3)*T^(2) -0.2787*10^(-5)*T^(3)
    +0.446*10^(-9)*T^(4) - 4.119*10^(5)*T^(-1))/T^(2)
    ','T',T_1,T_2);// log(K_2/K_1)

// We know that  delta_G_rkn_T = -R*T*log(K)
// At 298.15 K
K_298 = exp(-delta_G_rkn_298/(R*T_1) );

// Putting the values in the above expression we get
// log(K_1042/K_298) = log_K2_K1/R
K_1042 = K_298*exp(log_K2_K1/R);

printf(" The value of equilibrium constant at 1042 K
     is , K_1042 = %f\n\n",K_1042);

// Since for the given reaction K = P_CO2,where P is
     in atm, therefore,
P_CO2 = K_1042;
// and thus decomposition takes place till the
   partial pressure of CO2 reaches 0.095 atm
// After that the decomposition in the closed vessel
     stops as equilibrium is achieved.

printf(" The equilibrium pressure of CO2 is , P_CO2 =
     %f atm \n",P_CO2);
```

Scilab code Exa 17.19 Determination of the value of equilibrium constant

```
clear;
clc;

//Example - 17.19
//Page number - 620
printf("Example - 17.19 and Page number - 620\n\n");

```

```
// Given
T_1 = 298.15;//[k] - Standard reaction temperature
T_2 = 1200;//[K] - Reaction temperature
R = 1.987;//[cal/mol-K] - Universal gas consatnt

// C (s) + CO2 (g) - 2*CO2 (g)  // Reaction 1
// CO2 + H2 - CO + H2O  // Reacction 2

K_1 = 63;// Equilibrium constant for the first
    reaction
K_2 = 1.4;// Equilibrium constant for the secind
    reaction

delta_G_H2O_298 = -54640;//[cal/mol] - Standard
    Gibbs free energy of formation of water vapour
delta_H_H2O_298 = -57800;//[cal/mol] - Standard
    enthalpy change of formation of water vapour
delta_G_rkn_298 = delta_G_H2O_298;

// The standard heat capacity data of the components
     in cal/mol-K are given below
// Cp_H2 = 6.947 - 0.2*10^(-3)*T + 4.8*10^(-7)*T^(2)
// Cp_O2 = 6.148 + 3.1*10^(-3)*T - 9.2*10^(-7)*T^(2)
// Cp_H2O = 7.256 + 2.3*10^(-3)*T + 2.8*10^(-7)*T
    ^(2)

// Let us consider two more reactions
// C (s) + (1/2)*O2 - CO // Reaction 3
// H2 + (1/2)*O2 - H2O  // Reaction 4

// Considering the above 4 reactions, it can be
    shown that reaction (3) = reaction (1) + reaction
     (4) - reaction (2)
// Thus,  delta_G_rkn_3 = delta_G_rkn_1 +
    delta_G_rkn_4 - delta_G_rkn_2
// or, -R*T*log(K_3) = -R*T*log(K_1) + -R*T*log(K_4)
     - -R*T*log(K_2)
// K_3 = (K_1*K_4/K_2)

// Now we have to determine K_4 at 1200 K.
// The standard enthalpy change of reaction (4) as a
    fuction of temperature is given by
// delta_H_rkn_T = -57020 - 2.765*T + 0.475*10^(-3)*
    T^(2) + 8.67*10^(-8)*T^(3);

// log(K_4_2/K_4_1) = integrate('delta_H_rkn_T/(R*T
    ^(2))','T',T_1,T)
log_K2_K1_4 = integrate('(-57020-2.765*T
    +0.475*10^(-3)*T^(2)+8.67*10^(-8)*T^(3))/T^(2)','
    T',T_1,T_2);

// We know that  delta_G_rkn_T = -R*T*log(K)
// At 298.15 K
K_4_298 = exp(-delta_G_rkn_298/(R*T_1) );

// Putting the values in the above expression we get
// log(K_4_1200/K_4_298) = log_K2_K1_R/R
K_4_1200 = K_4_298*exp(log_K2_K1_4/R);
K_4 = K_4_1200;

// Therefore the equilibrium constant for reaction
    (3) at 1200 K is given by
K_3 = (K_1*K_4)/K_2;

printf(" The equilibrium constant at 1200 K for the
    given reaction is , K = %e\n",K_3);
```

Scilab code Exa 17.20 Calculation of standard equilibrium cell voltage

```
clear;
clc;

//Example - 17.20
```

```
//Page number - 622
printf("Example - 17.20 and Page number - 622\n\n");

// Given
delta_G_H2O_298 = -237.034;//[kJ/mol] - Standard
   Gibbs free energy of formation of H2O (l) at 298
   K
F = 96485;//[C/mol] - Faraday constant

//(1)
// For the reaction
// H2 (g) + (1/2)*O2 (g) - H2O (l)
n = 2;// Number of electrons transferred in the
   reaction

// The standard Gibbs free energy change of the
   reaction is given by
// delta_G_rkn = delta_G_for_H2O(l) - delta_G_for_H2
   (g) - (1/2/)*delta_G_for_O2(g)
// Since delta_G_for_H2  = 0 and delta_G_for_O2 = 0
    (pure components)
delta_G_rkn = delta_G_H2O_298;//[kJ]
delta_G_rkn = delta_G_rkn*10^(3);//[J]

// delta_G_rkn = -n*F*E_0
// Thus standard equilibrium cell voltage is given
   by
E_0 = - delta_G_rkn/(n*F);///[V]

printf(" (1).The standard equilibrium cell voltage
   is %f V\n\n",E_0);

//(2)
// For the reaction
// 2*H2 (g) + O2 (g) - 2*H2O (l)
n_prime = 4;// Number of electrons transferred in
   the reaction

```

```
// The standard Gibbs free energy change of the
   reaction is given by
// delta_G_rkn = 2*delta_G_for_H2O(l) - 2*
   delta_G_for_H2(g) - delta_G_for_O2(g)
// Since delta_G_for_H2  = 0 and delta_G_for_O2 = 0
    (pure components)
delta_G_rkn_prime = 2*delta_G_H2O_298;//[kJ]
delta_G_rkn_prime = delta_G_rkn_prime*10^(3);//[J]

// delta_G_rkn = -n*F*E_0
// Thus standard equilibrium cell voltage is given
   by
E_0_prime = - delta_G_rkn_prime/(n_prime*F);///[V]

printf(" (2).The standard equilibrium cell voltage
   is %f V\n\n",E_0_prime);

// Thus the result obtained is same for both the
   reactions
```

Scilab code Exa 17.21 Calculation of number of chemical reactions

```
clear;
clc;

//Example - 17.21
//Page number - 624
printf("Example - 17.21 and Page number - 624\n\n");

// Given
P = 2;// Number of phases
C = 5;// Number of components

// First we write chemical equations for formation
   of each component from basic elements
```

```
// C + 2*H2 = CH4    // (reaction 1)
// H2 + (1/2)*O2 = H20   // (reaction 2)
// C + (1/2)*O2 = CO   // (reaction 3)
// C + O2 = CO2   // (reaction 4)

// We do not have C in the equilibrium reaction
   mixture, therefore we have to eliminate it.
// Substituting for C from reaction (1) into
   reactions (3) and (4) we get the following set of
    reactions
// H2 + (1/2)*O2 = H20
// CH4 - 2*H2 + (1/2)*O2 = CO
// CH4 - 2*H2 + O2 = CO2

// or,
// H2 + (1/2)*O2 = H2O
// CH4 + (1/2)*O2 = CO + 2*H2
// CH4 + O2 = CO2 + 2*H2

// We do not have O2 in the equilibrium reaction
   mixture,therefore we have to eliminateit
// Substituting for O2 from the first reaction of
   the above set into seecond and third reactions of
    the above set we get the following set of
   reactions.
// CH4 + H2O - H2 = CO + 2*H2
// CH4 + 2*H20 - 2*H2 = CO2 + 2*H2

// Therefore one set of independent reactions is
// CH4 + H20 = CO + 3*H2
// CH4 + 2*H2O = CO2 + 4*H2

// Another set of independent reactions can be
   obtained by substituting for CH4 from the first
   reaction into second and we get
// CH4 + H2O = CO + 3*H2
// CO + 3*H2 - H2O + 2*H2O = CO2 4*H2

// Or,
// CH4 + H2O = CO + 3*H2
// CO + H2O = CO2 + H2
// This is another set of independent reactions.
   Thus whatever be the set of independent reactions
   , the number of independent reactions are two
// If different set of reactions are considered,
   then we get different equilibrium constants,
   different reaction coordinates but the final
   composition will be same
// Thus only 2 independent reactions occur,therefore
r = 2;

// and the number of degree of freedom becomes,
F = r - P + C;

printf(" The number of independent chemical
   reactions are %f \n\n",r);

printf(" The first set of independent reactions are
   given below\n");
printf(" CH4 + H20 = CO + 3*H2\n");
printf(" CH4 + 2*H2O = CO2 + 4*H2\n\n");

printf(" The second set of independent reactions are
    given below\n");
printf(" CH4 + H20 = CO + 3*H2\n");
printf(" CO + H2O = CO2 + H2");
```

Scilab code Exa 17.22 Calculation of number of chemical reactions

```
clear;
clc;

//Example - 17.22
```

```
//Page number - 626
printf("Example - 17.22 and Page number - 626\n\n");

// Given
T = 400;//[K] - Temperature
P = 1;//[atm] - Pressure
R = 1.987;//[cal/mol-K] - Universal gas consatnt

delta_G_n_pentane_400 = 9600;//[cal/mol] - Standard
    Gibbs free energy of formation of n-pentane at
    400 K
delta_G_iso_pentane_400 = 8200;//[cal/mol] -
    Standard Gibbs free energy of formation of iso-
    pentane at 400 K
delta_G_neo_pentane_400 = 9000;//[cal/mol] -
    Standard Gibbs free energy of formation of neo-
    pentane at 400 K

// The three reactions for the formation of these
    isomers can be written as follows
// 5*C + 6*H2 = n-pentane
// 5*C + 6*H2 = iso-pentane
// 5*C + 6*H2 = neo-pentane

// We can eliminate elemental carbon and hydrogen as
     they are not present in equilibrium reaction
    mixture and get the following two sets of
    independent reactions
// n-pentane = iso-pentane
// n-pentane = neo-pentane

// or,
// iso-pentane = n-pentane
// iso-pentane = neo-pentane

// or,
// noe-pentane = iso-pentane
// neo-pentane = n-pentane

// Let us take the following set of independent
   reactions
// iso-pentane = n-pentane   //      (reaction 1)
delta_G_rkn_1 = delta_G_n_pentane_400 -
   delta_G_iso_pentane_400;//[cal]
K_1 = exp(-delta_G_rkn_1/(R*T));
// iso-pentane = neo-pentane //      (reaction 2)
delta_G_rkn_2 = delta_G_neo_pentane_400 -
   delta_G_iso_pentane_400;//[cal]
K_2 = exp(-delta_G_rkn_2/(R*T));

// Let the initial number of moles be
// n_iso_pentane = 1
// n_n_pentane = 0
// n_neo_pentane = 0

// Let the reaction coordinate at equilibrium for
   the two reaction be X_1 and X_2 respectively
// At equilibrium, the moles of the components be
// n_iso_pentane_eq = 1 - X_1 - X_2
// n_n_pentane_eq = X_1
// n_neo_pentane_eq = X_2
// Total moles = 1

// Pressure has no effect on these reactions ( P = 1
    atm) and therefore
Ky_1 = K_1;
Ky_2 = K_2;

// From reaction (1), we get
// Ky_1 = X_1/(1-X_1-X_2)

// From reaction (2), we get
// Ky_2 = X_2/(1-X_1-X_2)

// Putting the value of X_1 from first equation into
    the second we get
```

```
// X_1 = (Ky_1*(1-X_2))/(1+Ky_1)
// Now putting it into the second equation we grt
// Ky_2 = X_2/(1-((Ky_1*(1-X_2))/(1+Ky_1))-X_2)
// Now solving for X_2
deff('[y]=f(X_2)','y= Ky_2 - X_2/(1-((Ky_1*(1-X_2))
   /(1+Ky_1))-X_2)');
X_2 = fsolve(0.8,f);

// Therefore X_1 can be calculated as
X_1 = (Ky_1*(1-X_2))/(1+Ky_1);

// Finally the mole fractions of the components at
   equilibrium
y_n_pentane = X_1;
y_neo_pentane = X_2;
y_iso_pentane = 1 -X_1 - X_2;

printf(" The equilibrium composition is given by,
   y_n_pentane = %f, y_neo_pentane = %f and
   y_iso_pentane = %f\n\n\n",y_n_pentane,
   y_neo_pentane,y_iso_pentane);

// Let us consider another set of independent
   reactions

// n-pentane = iso-pentane   //     (reaction 3)
delta_G_rkn_3 = delta_G_iso_pentane_400 -
   delta_G_n_pentane_400;//[cal]
K_3 = exp(-delta_G_rkn_3/(R*T));
// n-pentane = neo-pentane //     (reaction 4)
delta_G_rkn_4 = delta_G_neo_pentane_400 -
   delta_G_n_pentane_400;//[cal]
K_4 = exp(-delta_G_rkn_4/(R*T));

// Let the initial number of moles be
// n_n_pentane = 1
// n_iso_pentane = 0
// n_neo_pentane = 0

// Let the reaction coordinate at equilibrium for
   the two reaction be X_3 and X_4 respectively
// At equilibrium , the moles of the components be
// n_n_pentane_eq = 1 - X_3 - X_4
// n_iso_pentane_eq = X_4
// n_neo_pentane_eq = X_4
// Total moles = 1

// Pressure has no effect on these reactions (P = 1
   atm) and therefore
Ky_3 = K_3;
Ky_4 = K_4;

// From reaction (3) , we get
// Ky_3 = X_3/(1-X_3-X_4)

// From reaction (4) , we get
// Ky_4 = X_4/(1-X_3-X_4)

// Putting the value of X_3 from first equation into
   the second we get
// X_3 = (Ky_3*(1-X_4))/(1+Ky_3)
// Now putting it into the second equation we grt
// Ky_4 = X_4/(1-((Ky_1*(1-X_4))/(1+Ky_3))-X_4)
// Now solving for X_4
deff('[y]=f1(X_4)','y= Ky_4 - X_4/(1-((Ky_3*(1-X_4))
   /(1+Ky_3))-X_4)');
X_4 = fsolve(0.8,f1);

// Therefore X_3 can be calculated as
X_3 = (Ky_3*(1-X_4))/(1+Ky_3);

// Finally the mole fractions of the components at
   equilibrium
y_n_pentane1 = 1 - X_3 - X_4;
y_neo_pentane1 = X_4;
y_iso_pentane1 = X_3;
```

```

// The final composition does not depend on the set
    of reactions considered.

printf(" For another set of independent reactions
    considered\n");
printf(" The equilibrium composition is given by,
    y_n_pentane = %f, y_neo_pentane = %f and
    y_iso_pentane = %f\n\n\n",y_n_pentane1,
    y_neo_pentane1,y_iso_pentane1);
printf(" Thus the final composition does not depend
    on the set of reactions considered\n\n\n");
printf(" The number of independent reactions taking
    place is two");
```

Scilab code Exa 17.23 Calculation of equilibrium composition

```
clear;
clc;

//Example - 17.23
//Page number - 628
printf("Example - 17.23 and Page number - 628\n\n");

// Given
T = 600;//[K] - Temperature
P = 1;//[atm] - Pressure
R = 1.987;//[cal/mol-K] - Universal gas consatnt

// CH4 + H2O = CO + 3*H2 //     (Reaction 1)
// CO + H2O = CO2 + H2  //      (Reaction 2)

K_1 = 0.574;// Equilibrium constant of first
    reaction
K_2 = 2.21;// Equilibrium constant of second
```

```
    reaction

// Initial number of moles of the components are
// n_CH4 = 1
// n_H2O = 5
// n_CO = 0
// n_H2 = O
// n_CO2 = 0

// Let the reaction coordinate at equilibrium for
    the two reaction be X_1 and X_2 respectively
// At equilibrium , the moles of the components be
// n_CH4_eq = 1 - X_1
// n_H20_eq = 5 - X_1 - X_2
// n_CO_eq = X_1 - X_2
// n_H2_eq = 3*X_1 + X_2
// n_CO2_eq = X_2
// Total moles = 6 + 2*X_1

// Since the pressure is 1 atm, K = Kp, Ky = K
Ky_1 = K_1;
Ky_2 = K_2;

// From reaction (1), we get
// Ky_1 = ((X_1-X_2)*(3*X_1+X_2)^(3))/((1-X_1)*(5-
    X_1-X_2)*(6+2*X_1)^(2))

// From reaction (2), we get
// Ky_2 = (X_2*(3*X_1+X_2))/((X_1-X_2)*(5-X_1-X_2))

// Let us assume a value of X_1
X_1 = 0.1;

fault = 10;
while(fault>0.05)
    deff('[y]=f(X_2)','y= Ky_1-((X_1-X_2)*(3*X_1+X_2
        )^(3))/((1-X_1)*(5-X_1-X_2)*(6+2*X_1)^(2))');
    X_2 = fsolve(0.8,f);
    X_2_prime = X_2;
    deff('[y]=f1(X_1_prime)','y= Ky_2-(X_2_prime*(3*
        X_1_prime+X_2_prime))/((X_1_prime-X_2_prime)
        *(5-X_1_prime-X_2_prime))');
    X_1_prime = fsolve(0.8,f1);
    fault=abs(X_1 - X_1_prime);
    X_1 = X_1 + 0.001;
end

n_CH4 = 1 - X_1;
n_H20 = 5 - X_1 - X_2;
n_CO = X_1 - X_2;
n_H2 = 3*X_1 + X_2;
n_CO2 = X_2;
Total_moles = 6 + 2*X_1;

printf(" The equilibrium composition of the
    resulting mixture is given by\n");
printf(" n_CH4 = % f mol\n n_H2O = %f mol\n n_CO =
    %f mol\n n_H2 = %f mol and\n n_CO2 = %f mol\n\n",
    n_CH4,n_H20,n_CO,n_H2,n_CO2);
printf(" The total number of moles is %f mol\n",
    Total_moles);
```

Scilab code Exa 17.24 Determination of number of moles

```
clear;
clc;

//Example - 17.24
//Page number - 631
printf("Example - 17.24 and Page number - 631\n\n");

// Given
T = 600 + 273.15;//[K] - Reaction temperature
```

```
P = 1;//[atm] - Reaction pressure

// The Gibbs free energy of formation of various
   species at 873.15 K are
delta_G_CH4_RT = -2.82;// delta_G_CH4/(R*T)
delta_G_H2O_RT = -29.73;// delta_G_CH4/(R*T)
delta_G_CO_RT = -27.51;// delta_G_CH4/(R*T)
delta_G_H2_RT = -1.46;// delta_G_CH4/(R*T)
delta_G_CO2_RT = -56.68;// delta_G_CH4/(R*T)

// Initial number of moles of the components are
// n_CH4 = 1
// n_H2O = 5
// n_CO = 0
// n_H2 = O
// n_CO2 = 0

// The del(F)/del(n_i) = 0 equations for CH4 (1),
   H2O (2), CO (3), H2 (4) and CO2 (5) becomes
// delta_G_1_T + R*T*log((n_1*P)/n) + lambda_C + 4*
   lambda_H = 0
// delta_G_2_T + R*T*log((n_2*P)/n) + 2*lambda_C +
   lambda_O = 0
// delta_G_3_T + R*T*log((n_3*P)/n) + lambda_c +
   lambda_O = 0
// delta_G_4_T + R*T*log((n_4*P)/n) + 2*lambda_H = 0
// delta_G_5_T + R*T*log((n_5*P)/n) + lambda_C + 2*
   lambda_O = 0

// Where n is the total number of moles in the
   equilibrium reaction mixture. Dividing the above
   equations by R*T we get
// delta_G_1_T/(R*T) + log((n_1*P)/n) + lambda_C/(R*
   T) + 4*lambda_H/(R*T) = 0
// delta_G_2_T/(R*T) + log((n_2*P)/n) + 2*lambda_C/(
   R*T) + lambda_O/(R*T) = 0
// delta_G_3_T/(R*T) + log((n_3*P)/n) + lambda_c/(R*
   T) + lambda_O/(R*T) = 0
```

```
// delta_G_4_T/(R*T) + log((n_4*P)/n) + 2*lambda_H/(
   R*T) = 0
// delta_G_5_T/(R*T) + log((n_5*P)/n) + lambda_C/(R*
   T) + 2*lambda_O/(R*T) = 0

// Substituting the values of delta_G_i_T/(R*T) in
   the above equations,the full set of equations (
   including elemental balance) becomes
// -2.82 + log(n_1/n) + lambda_C/(R*T) + 4*lambda_H
   /(R*T) = 0
// -29.73 + log(n_2/n) + 2*lambda_H/(R*T) + lambda_O
   /(R*T) = 0
// -27.51 + log(n_3/n) + lambda_C/(R*T) + lambda_O/(
   R*T) = 0
// -1.46 + log(n_4/n) + 2*lambda_H/(R*T) = 0
// -56.68 + log(n_5/n) + lambda_C/(R*T) + 2*lambda_O
   /(R*T) = 0

// The constraint equations are
// n_1 + n_3 + n_5  = 1 // (moles of C in the
   reaction mixture = 1)
// 4*n_1 + 2*n_2 + 2*n_4 = 14  // (moles of H in the
    reaction mixture = 14)
// n_2 + n_3 + 2*n_5 = 5  // (moles of O in the
   raection mixture = 5)

// The total moles are given by
// n = n_1 + n_2 + n_3 + n_4 + n_5

function[f]=solution(x)
f(1) = -2.82 + log(x(1)/x(6)) + x(7) + 4*x(8);
f(2) = -29.73 + log(x(2)/x(6)) + 2*x(8) + x(9);
f(3) = -27.51 + log(x(3)/x(6)) + x(7) + x(9);
f(4) = -1.46 + log(x(4)/x(6)) + 2*x(8);
f(5) = -56.68 + log(x(5)/x(6)) + x(7) + 2*x(9);
f(6) = x(1) + x(3) +x(5) - 1;
f(7) = 4*x(1) + 2*x(2) + 2*x(4) - 14;
f(8) = x(2) + x(3) +2*x(5) - 5;
```

```
f(9) = x(1)+ x(2) + x(3) + x(4) + x(5) - x(6);

funcprot(0);
endfunction
x = [0.01,3.5,0.2,3.0,0.5,5.0,2.0,1.0,25.0];
y = fsolve(x,solution);

printf(" n_1 = %f mol\n",y(1));
printf(" n_2 = %f mol\n",y(2));
printf(" n_3 = %f mol\n",y(3));
printf(" n_4 = %f mol\n",y(4));
printf(" n_5 = %f mol\n",y(5));
printf(" n = %f mol\n",y(6));
printf(" lambda_C/RT = %f\n",y(7));
printf(" lambda_H/RT = %f\n",y(8));
printf(" lambda_O/RT = %f\n\n",y(9));

printf(" The Lagrange multiplier values do not have
   any physical significance\n");
```

Chapter 18

Adiabatic Reaction Temperature

Scilab code Exa 18.1 Calculation of heat transfer

```
clear;
clc;

//Example - 18.1
//Page number - 650
printf("Example - 18.1 and Page number - 648\n\n");

// Given

T_1 = 298.15;//[K] - Standard reaction temperature
T_2 = 500;//[K] - Reaction temperature
P = 1;//[atm] - Pressure

a_CO2 = 5.316;
a_O2 = 6.085;
a_N2 = 6.903;
a_H2O = 7.700;
a_C3H8 = -0.966;
b_CO2 = 1.4285*10^(-2);
```

```
b_O2 = 0.3631*10^(-2);
b_N2 = -0.03753*10^(-2);
b_H2O = 0.04594*10^(-2);
b_C3H8 = 7.279*10^(-2);
c_CO2 = -0.8362*10^(-5);
c_O2 = -0.1709*10^(-5);
c_N2 = 0.1930*10^(-5);
c_H2O = 0.2521*10^(-5);
c_C3H8 = -3.755*10^(-5);
d_CO2 = 1.784*10^(-9);
d_O2 = 0.3133*10^(-9);
d_N2 = -0.6861*10^(-9);
d_H2O = -0.8587*10^(-9);
d_C3H8 = 7.580*10^(-9);

// The standard enthalpy of formation at 298.15 K is
    given by
delta_H_for_CO2 = -94.052;//[kcal/mol]
delta_H_for_C3H8 = -24.820;//[kcal/mol]
delta_H_for_H2O = -57.7979;//[kcal/mol]

// The reaction with stoichiometric amount of air is
// C3H8 + 5(O2 + 3.7N2) - 3CO2 + 4H2O + 18.8N2

// The reaction with 100% excess air is
// C3H8 + 10(O2 + 3.7N2) - 3CO2 + 4H2O + 5O2 + 37.6
   N2

// The standard enthalpy change of reaction at
   298.15 K
delta_H_rkn_298 = 3*delta_H_for_CO2 + 4*
   delta_H_for_H2O - delta_H_for_C3H8;

// For exit stream
sum_ai_ni = 3*a_CO2 + 4*a_H2O + 5*a_O2 + 37.6*a_N2;
sum_bi_ni = 3*b_CO2 + 4*b_H2O + 5*b_O2 + 37.6*b_N2;
sum_ci_ni = 3*c_CO2 + 4*c_H2O + 5*c_O2 + 37.6*c_N2;
sum_di_ni = 3*d_CO2 + 4*d_H2O + 5*d_O2 + 37.6*d_N2;
```

```


// To raise the exit species from 298.15 to 500 K
   the enthalpy change is
delta_H_rkn = integrate('sum_ai_ni+sum_bi_ni*T+
   sum_ci_ni*T^(2)+sum_di_ni*T^(3)','T',T_1,T_2);//[
   cal]
delta_H_rkn = delta_H_rkn*10^(-3);//[kcal]

// Therefore per mole of fuel the heat exchange is
// Q = Heat exchange in step 1 + Heat exchange in
   step 2
Q = delta_H_rkn_298 + delta_H_rkn;

printf("    The heat transfer from the combustion
   chamber per mole of fuel is %f kcal (per mol of
   C3H8)",Q);
```

Scilab code Exa 18.2 Calculation of adiabatic flame temperature

```
clear;
clc;

//Example - 18.2
//Page number - 650
printf("Example - 18.2 and Page number - 650\n\n");

// Given

T_1 = 298.15;//[K] - Standard reaction temperature

a_CO2 = 5.316;
a_H2O = 7.700;
a_O2 = 6.085;
a_C2H6 = 1.648;
```

```
b_CO2 = 1.4285*10^(-2);
b_H2O = 0.04595*10^(-2);
b_O2 = 0.3631*10^(-2);
b_C2H6 = 4.124*10^(-2);
c_CO2 = -0.8362*10^(-5);
c_H2O = 0.2521*10^(-5);
c_O2 = -0.1709*10^(-5);
c_C2H6 = -1.530*10^(-5);
d_CO2 = 1.784*10^(-9);
d_H2O = -0.8587*10^(-9);
d_O2 = 0.3133*10^(-9);
d_C2H6 = 1.740*10^(-9);

// The standard enthalpy of formation at 298.15 K is
    given by
delta_H_for_CO2 = -94.052;//[kcal/mol]
delta_H_for_C2H6 = -20.236;//[kcal/mol]
delta_H_for_H2O = -57.7979;//[kcal/mol]

// The reaction with stoichiometric amount of air is
// C2H6 + (7/2)O2 - 2CO2 + 3H2O

// The reaction with 4 mol of O2 and 10 mol CO2 is
// C2H6 + 4O2 + 10CO2 - 12H2O + 3H2O + 0.5O2
// The product consists of 12 mol of CO2, 3 mol of
   water vapour and 0.5 mol of oxygen
delta_H_rkn_298 = 2*delta_H_for_CO2 + 3*
   delta_H_for_H2O - delta_H_for_C2H6;//[kcal]
delta_H_rkn_298 = delta_H_rkn_298*10^(3);//[cal]

// For exit stream
sum_ai_ni = 12*a_CO2 + 3*a_H2O + 0.5*a_O2;
sum_bi_ni = 12*b_CO2 + 3*b_H2O + 0.5*b_O2;
sum_ci_ni = 12*c_CO2 + 3*c_H2O + 0.5*c_O2;
sum_di_ni = 12*d_CO2 + 3*d_H2O + 0.5*d_O2;

// From energy balance equation we get
// delta_H_rkn_298 + sum_ai_ni*(T_2 - T_1) + (
   sum_bi_ni/2)*(T_2^(2) - T_1^(2)) + (sum_ci_ni/3)
   *(T_2^(3) - T_1^(3)) + (sum_di_ni/4)*(T_2^(4) -
   T_1^(4))
// Solving above equation for T_2
deff('[y]=f(T_2)','y=delta_H_rkn_298 +sum_ai_ni*(T_2
   -T_1)+(sum_bi_ni/2)*(T_2^(2)-T_1^(2))+(sum_ci_ni
   /3)*(T_2^(3)-T_1^(3))+(sum_di_ni/4)*(T_2^(4)-T_1
   ^(4))');
T_2 = fsolve(-1,f);

printf(" The adiabatic flame temperature is %f K",
   T_2);
```

Scilab code Exa 18.3 Calculation of mole fraction and average heat capacity

```
clear;
clc;

//Example - 18.3
//Page number - 651
printf("Example - 18.3 and Page number - 651\n\n");

// Given
T_1 = 298.15;//[K] - Standard reaction temperature

// The reaction with theoritical air is
// CH4 + 2(O2 + 3.76N2) - CO2 + 2H20 + 7.52N2

//(1)
n_product = (1 + 2 + 7.52);// Total number of moles
   of product
// The mole fraction of water vapour is
y_H2O = 2/(n_product);
printf(" (1).The mole fraction of water vapour is %f
    \n\n",y_H2O);

//(2)
delta_H_rkn_298 = -730*10^(3);//[J/mol]
C = 40;//[J/mol-K] - Average molar heat capacity

// From energy balance we have
// delta_H_rkn_298 + n_product*C(T_2 - T_1) = 0
T_2 = - delta_H_rkn_298/(n_product*C) + T_1;//[K]
T_max = T_2 - T_1;

printf(" (2).The maximum temperature rise of the
    exhaust gases is %f K\n",T_max);
```

Scilab code Exa 18.4 Determination of adiabatic flame temperature

```
clear;
clc;

//Example - 18.4
//Page number - 651
printf("Example - 18.4 and Page number - 651\n\n");

// Given
T_1 = 298.15;//[K] - Standard reaction temperature

// The standard enthalpy of formation at 298.15 K is
    given by
delta_H_for_CO2 = -94.052;//[kcal/mol]
delta_H_for_C8H18 = -59.780;//[kcal/mol]
delta_H_for_H2O = -57.7979;//[kcal/mol]

a_CO2 = 5.316;
a_H2O = 7.700;
a_N2 = 6.903;
```

```
b_CO2 = 1.4285*10^(-2);
b_H2O = 0.04595*10^(-2);
b_N2 = -0.03753*10^(-2);
c_CO2 = -0.8362*10^(-5);
c_H2O = 0.2521*10^(-5);
c_N2 = 0.1930*10^(-5);
d_CO2 = 1.784*10^(-9);
d_H2O = -0.8587*10^(-9);
d_N2 = -0.6861*10^(-9);

//(a)
// The reaction with stoichiometric amount of air is
// C3H18 + 12.5(O2 + 3.76N2) - 8CO2 + 9H2O + 47N2

// The standard enthalpy change of reaction at
   298.15 K is
delta_H_rkn_298 = 8*delta_H_for_CO2 + 9*
   delta_H_for_H2O - delta_H_for_C8H18;//[kcal]
delta_H_rkn_298 = delta_H_rkn_298*10^(3);//[cal]

// For exit stream
sum_ai_ni = 8*a_CO2 + 9*a_H2O + 47*a_N2;
sum_bi_ni = 8*b_CO2 + 9*b_H2O + 47*b_N2;
sum_ci_ni = 8*c_CO2 + 9*c_H2O + 47*c_N2;
sum_di_ni = 8*d_CO2 + 9*d_H2O + 47*d_N2;

// From energy balance equation we get
// delta_H_rkn_298 + sum_ai_ni*(T_2 - T_1) + (
   sum_bi_ni/2)*(T_2^(2) - T_1^(2)) + (sum_ci_ni/3)
   *(T_2^(3) - T_1^(3)) + (sum_di_ni/4)*(T_2^(4) -
   T_1^(4))
// Solving above equation for T_2
deff('[y]=f(T_2)','y=delta_H_rkn_298 +sum_ai_ni*(T_2
   -T_1)+(sum_bi_ni/2)*(T_2^(2)-T_1^(2))+(sum_ci_ni
   /3)*(T_2^(3)-T_1^(3))+(sum_di_ni/4)*(T_2^(4)-T_1
   ^(4))');
T_2 = fsolve(-1,f);

```

```
printf(" (1).The adiabatic flame temperature is %f K
   \n\n",T_2);

//(2)
// The mean standard heat capacity of various
   components over the temperature range from 25 to
   3000 C is
Cp_CO2 = 13.91;//[cal/mol-K]
Cp_H2O = 10.16;//[cal/mol-K]
Cp_O2 = 7.88;//[cal/mol-K]
Cp_N2 = 7.45;//[cal/mol-K]

// From energy balance equation we get
// delta_H_rkn_298 + (8*Cp_CO2 + 9*Cp_H2O + 47*Cp_N2
   )*(T_2_prime)
T_2_prime = - delta_H_rkn_298/(8*Cp_CO2 + 9*Cp_H2O +
    47*Cp_N2);//[K]

printf(" (2).The adiabatic flame temperature is %f K
   ",T_2_prime);
```

Scilab code Exa 18.5 Calculation of conversion

```
clear;
clc;

//Example - 18.5
//Page number - 652
printf("Example - 18.5 and Page number - 652\n\n");

// Given
// N2 + 3H2 - 2NH3
T_1 = 700;//[K] - Reaction temperature
Max_adia_rise = 100;///[K] - Maximum adiabatic rise
   in temperature
```

```
T_2 = T_1 + Max_adia_rise;//[K] -

delta_H_rkn_700 = -94.2;//[kJ] - Standard enthalpy
    of reaction at 700 K
delta_H_rkn_700 = delta_H_rkn_700*10^(3);//[J]

// The mean standard heat capacity of various
    components over the temperature range from 700 to
     800 K is
Cp_N2 = 30.0;//[cal/mol-K]
Cp_H2 = 28.9;//[cal/mol-K]
Cp_NH3 = 49.2;//[cal/mol-K]

// The energy balance equation is
// X*delta_H_rkn_700 + integrate('(sum_ni_Cpi_exit)*
    dT','T',T_1,T_2)

//At exit, let moles of NH3 = (1-X),  moles of H2 =
    (3-3X),  moles of NH3 = 2X . Therefore we have,
// delta_H_rkn_700*X + {(1-X)*Cp_N2 + (3-3X)*Cp_H2 +
     (2X)*Cp_NH3}*(T_2 - T_1)
// On simplification we get, 960.3*X = 116.7
X = 116.7/960.3;

printf(" The maximum allowable conversion fraction
    in the reactor is given by, X =  %f \n",X);
```

Scilab code Exa 18.6 Calculation of maximum pressure

```
clear;
clc;

//Example - 18.6
//Page number - 653
printf("Example - 18.6 and Page number - 653\n\n");
```

```

// Given
T_1 = 298.15;//[K] - Standard reaction temperature
V = 2.0*10^(-3);//[m^(3)] - Volume of calorimeter
m = 10;//[g] - Mass of liquid octane
Mol_wt = 114;//[g/mol] - Molecular weight of octane
n = m/Mol_wt;//[mol] - No of moles of octane
R = 8.314;//[J/mol*K] - Universal gas constant

// The standard enthalpy of formation at 298.15 K is
    given by
delta_H_for_CO2 = -94.052;//[kcal/mol]
delta_H_for_C8H18 = -59.780;//[kcal/mol]
delta_H_for_H2O = -57.7979;//[kcal/mol]

// The standard molar heat capacity of various
   components in high temperature range from is
   given by
// Cp_H2O = 6.970 + 0.3464*10^(-2)*T -
   0.04833*10^(-5)*T^(2);
// Cp_O2 = 6.732 + 0.1505*10^(-2)*T -
   0.01791*10^(-5)*T^(2);
// Cp_CO2 = 18.036 - 4.474*10^(-5)*T - 158.08/(T
   ^(1/2));
// Therefore we have
// Sum_ni_Cpi_exit = 249.09 + 0.04*T - 0.547*10^(-5)
   *T^(2) - 1264.64/(T^(1/2))

// The reaction with stoichiometric amount of oxygen
    is
// C8H18 + 12.5O2 - 8CO2 + 9H2O

// The reaction with 50% excess oxygen is
// C8H18 + 18.75O2 - 8CO2 +9H2O + 6.25O2

// The standard enthalpy change of reaction at
   298.15 K is
delta_H_rkn_298 = 8*delta_H_for_CO2 + 9*
```

```
    delta_H_for_H2O - delta_H_for_C8H18;//[kcal]
delta_H_rkn_298 = delta_H_rkn_298*10^(3);//[cal]

// From the energy balance equation we get
// delta_H_rkn_298 + integrate('(sum_ni_Cpi_exit)*dT
    ','T',T_1,T_2)
// delta_H_rkn_298 + 249.09*(T_2 - T_1) + (0.04/2)*(
    T_2^(2) - T_1^(2)) - ((0.547*10^(-5))/3)*(T_2^(3)
    -T_1^(3)) - (1264.64*2)*(T_2^(1/2)-T_1^(1/2))
// Solving above equation for T_2
deff('[y]=f(T_2)','y=delta_H_rkn_298 + 249.09*(T_2 -
     T_1) + (0.04/2)*(T_2^(2)-T_1^(2)) -
    ((0.547*10^(-5))/3)*(T_2^(3)-T_1^(3)) -
    (1264.64*2)*(T_2^(1/2)-T_1^(1/2))');
T_2 = fsolve(1000,f);

// When 1 mol of octane reacts the final number of
    moles in the calorimeter is 23.25
// When n mol of octane reacts the final number of
    moles in the calorimeter is
n_total = n*23.25;//[mol]

// The maximum explosion pressure is calculated when
     no heat is dissipated to the surroundings and
    thus bomb calorimeter attains the adiabatic flame
     temperature
// Thus maximum explosion pressure is given by
P = (n_total*R*T_2)/V;//[N/m^(2)]
P = P*10^(-5);//[bar]

printf(" The maximum explosion pressure inside the
    bomb calorimeter is %f bar",P);
```

Scilab code Exa 18.7 Calculation of number of moles

```
clear;
clc;

//Example - 18.7
//Page number - 656
printf("Example - 18.7 and Page number - 656\n\n");

// Given
T_1 = 400 + 273.15;//[K]
// SO2(g) + 1/2*(O2) (g) - SO3 (g)

a_SO2 = 6.157;
a_SO3 = 3.918;
a_O2 = 6.085;
a_N2 = 6.903;
b_SO2 = 1.384*10^(-2);
b_SO3 = 3.483*10^(-2);
b_O2 = 0.3631*10^(-2);
b_N2 = -0.03753*10^(-2);
c_SO2 = -0.9103*10^(-5);
c_SO3 = -2.675*10^(-5);
c_O2 = -0.01709*10^(-5);
c_N2 = 0.1930;
d_SO2 = 2.057*10^(-9);
d_SO3 = 7.744*10^(-9);
d_O2 = 0.3133*10^(-9);
d_N2 = -0.6861*10^(-9);

// At 400 C, from the given expressions
delta_H_rkn_T_1 = -22630.14 - 5.2815*T_1 +
   0.9587*10^(-2)*T_1^(2) - 0.5598*10^(-5)*T_1^(3) +
    1.3826*10^(-9)*T_1^(4);//[cal]
// This is the standard enthalpy change of reaction
   for 1 mol of SO2 reacted. Since X moles of SO2
   are reactants therefore
// delta_H_rkn_T_X (for X moles of SO2 reacted) =
   delta_H_rkn_T_1*X

```

```
// Let the number of moles at equilibrium be
// n_O2 = 9-0.5*X
// n_SO2 =  12-X
// n_SO3 = X
// n_N2 = 79
// Total moles at equilibrium = 100-0.5X
// Ky = y_SO3/(y_SO2*y_O2^(1/2))
// Ky = (X*(100-0.5*X)^(1/2))/((12-X)*(9-0.5*X)
    ^(1/2))
// We know that K = Ky*Kp. Since P = 1 atm,
    therefore Ky = K

// Now we have to account for the heat required to
    raise 9-0.5*X mol of O2, 12-X mol of SO2, X mol
    of SO3 and 79 mol of N2 from T to ART
// sum_ni_Cp_i = (12-X)*(a + b*T + c*T^(2) + d*T^(3)
    ) + (9-0.5*X)*(a + b*T + c*T^(2) + d*T^(3)) + X*(
    a + b*T + c*T^(2) + d*T^(3)) + 79*(a + b*T + c*T^
       (2) + d*T^(3))

// From energy balance equation we get
// delta_H_rkn_T_1 + integrate('sum_ni_Cp_i','T',T_1
    ,T)
// The above equation on simplification becomes
// (673.99-5.2815*X)*(T-T_1) + (16.91+1.9175*X)
    *(10^(-2)/2)*(T^(2)-T_1^(2)) + (2.79-1.6793*X)
    *(10^(-5)/3)*(T^(3)-T_1^(3)) + (-26.70+5.5304*X)
    *(10^(-9)   /4)*(T^(4)-T_1^(4)) = delta_H_rkn_T_1
    *X

// Let us assume a temperature, say
T = 800;//[K]
fault = 10;

while(fault>0.01)
    K = exp(3.87 + (11389.10/T) - 2.6580*log(T) +
       0.4825*10^(-2)*T - 0.1409*10^(-5)*T^(2) +
       0.2320*10^(-9)*T^(3));
    deff('[y]=f(X)','y= K - (X*(100-0.5*X)^(1/2))
       /((12-X)*(9-0.5*X)^(1/2))');
    X1 = fsolve(0.1,f);// X from equilibrium
       equation
    deff('[y]=f1(X)','y= (673.99-5.2815*X)*(T-T_1)
       +(16.91+1.9175*X)*(10^(-2)/2)*(T^(2)-T_1^(2))
       +(2.79-1.6793*X)*(10^(-5)/3)*(T^(3)-T_1^(3))
       +(-26.70+5.5304*X)*(10^(-9)/4)*(T^(4)-T_1^(4)
       )+delta_H_rkn_T_1*X');
    X2 = fsolve(1,f1);// X from energy balance
       equation
    fault = abs(X1-X2);
    T = T + 0.01;
end

printf(" The moles of SO2 reacted are %f mol\n\n",X1
   );
printf(" The adiabatic reaction temperature is %f K\
   n",T);
```