Scilab Textbook Companion for Introduction To Numerical Methods In Chemical Engineering by P. Ahuja[1]

Created by
Chandra Prakash Sipani
B.TECH Part-2
Chemical Engineering
IIT-BHU
College Teacher
Ahuja, Pradeep
Cross-Checked by
Pradeep Ahuja

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: Introduction To Numerical Methods In Chemical Engineering

Author: P. Ahuja

Publisher: PHI Learning, New Delhi

Edition: 1

Year: 2010

ISBN: 8120340183

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

linear algebraic equations

Scilab code Exa 1.1 TDMA method

```
//ch 1 ex 1.1- solving set of algebraic equations by Tri diagonal Matrix Algorithm Method.
clc
disp("the soln of eg 1.1-->");
for i=2:7, a(i)=1;              //sub diagonal assignment
end
for j=1:7, b(j)=-2;             //main diagonal assignment
end
for k=1:6, c(k)=1;              //super diagonal assignment
end
d(1)=-240                       //given values assignment
for l=2:6, d(l)=-40;
end
d(7)=-60

i=1;
n=7;
```

```
beta1(i)=b(i);                    //initial b is equal to beta since a1=0
gamma1(i)=d(i)/beta1(i);          //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                   //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end

disp("the solution of ex 1.1 by TDMA method is");
for i=1:7, disp(x(i));
end
```

Scilab code Exa 1.2 gauss elimination method

```
//ch 1 ex 1.2
clc
disp("the soln of eg 1.2-->");
a1=10, a2=1, a3=2,                  //1st row
b1=2, b2=10, b3=1,                  //2nd row
c1=1, c2=2, c3=10,                  //3rd row
d1=44, d2=51, d3=61,                //given values

b3=b3-(b1/a1)*a3                    // for making b1=0
b2=b2-(b1/a1)*a2
d2=d2-(b1/a1)*d1
b1=b1-(b1/a1)*a1

c3=c3-(c1/a1)*a3                   // for making c1=0
c2=c2-(c1/a1)*a2
d3=d3-(c1/a1)*d1
```

```
c1=c1-(c1/a1)*a1

c3=c3-(c2/b2)*b3                          // for making c2=0
d3=d3-(c2/b2)*d2
c2=c2-(c2/b2)*b2

x3=d3/c3;                                 // final values of x
x2=(d2-(b3*x3))/b2;
x1=(d1-(x3*a3)-(x2*a2))/a1;
disp(x3,x2,x1,"the solution using gauss elimination
   method is");
```

Scilab code Exa 1.3 gauss elimination method

```
//ch 1 ex 1.3
clc
disp("the soln of eg 1.3-->");
a1=3, a2=1, a3=-2,                        //1st row
b1=-1, b2=4, b3=-3,                       //2nd row
c1=1, c2=-1, c3=4,                        //3rd row
d1=9, d2=-8, d3=1,                        //given values

b3=b3-(b1/a1)*a3                          // for making b1=0
b2=b2-(b1/a1)*a2
d2=d2-(b1/a1)*d1
b1=b1-(b1/a1)*a1

c3=c3-(c1/a1)*a3                          // for making c1=0
c2=c2-(c1/a1)*a2
d3=d3-(c1/a1)*d1
c1=c1-(c1/a1)*a1

c3=c3-(c2/b2)*b3                          // for making c2=0
d3=d3-(c2/b2)*d2
c2=c2-(c2/b2)*b2
```

```

x3=d3/c3;                          // final values of x
x2=(d2-(b3*x3))/b2;
x1=(d1-(x3*a3)-(x2*a2))/a1;
disp(x3,x2,x1,"the solution using gauss elimination
    method is");
```

Scilab code Exa 1.4 gauss elimination method

```
// ch 1 ex 1.4
clc
disp("the solution of eg 1.4-->");
a1=.35, a2=.16, a3=.21, a4=.01                     //1st
    row
b1=.54, b2=.42, b3=.54, b4=.1                     //2nd row
c1=.04, c2=.24, c3=.1, c4=.65                     //3rd row
d1=.07, d2=.18, d3=.15, d4=.24                    //4th row
r1=14, r2=28, r3=17.5, r4=10.5                   //given
    values

b4=b4-(b1/a1)*a4                                 // for
    making b1=0
b3=b3-(b1/a1)*a3
b2=b2-(b1/a1)*a2
r2=r2-(b1/a1)*r1
b1=b1-(b1/a1)*a1

c4=c4-(c1/a1)*a4                                // for
    making c1=0
c3=c3-(c1/a1)*a3
c2=c2-(c1/a1)*a2
r3=r3-(c1/a1)*r1
c1=c1-(c1/a1)*a1

d4=d4-(d1/a1)*a4                                // for
```

```
making d1=0
d3=d3-(d1/a1)*a3
d2=d2-(d1/a1)*a2
r4=r4-(d1/a1)*r1
d1=d1-(d1/a1)*a1

c4=c4-(c2/b2)*b4                                  // for making c2=0
c3=c3-(c2/b2)*b3
r3=r3-(c2/b2)*r2
c2=c2-(c2/b2)*b2

d4=d4-(d2/b2)*b4                                  // for making d2=0
d3=d3-(d2/b2)*b3
r4=r4-(d2/b2)*r2
d2=d2-(d2/b2)*b2

d4=d4-(d3/c3)*c4                                  //for making d3=0
r4=r4-(d3/c3)*r3
d3=d3-(d3/c3)*c3

B2=r4/d4;
D2=(r3-(c4*B2))/c3;
B1=(r2-(D2*b3)-(B2*b4))/b2;
D1=(r1-(B2*a4)-(D2*a3)-(B1*a2))/a1;
disp(B2,D2,B1,D1,"the values of MOLAR FLOW RATES of D1, B1, D2, B2 respectively are");

B=D2+B2;
x1B=(.21*D2 + .01*B2)/B;
x2B=(.54*D2 + .1*B2)/B;
x3B=(.1*D2 + .65*B2)/B;
x4B=(.15*D2 + .24*B2)/B;
disp(x4B, x3B, x2B, x1B,"the composition of stream B is");

```

```
D=D1+B1;
x1D=(.35*D1 + .16*B1)/D;
x2D=(.54*D1 + .42*B1)/D;
x3D=(.04*D1 + .24*B1)/D;
x4D=(.07*D1 + .18*B1)/D;
disp(x4D, x2D, x2D, x1D,"the composition of stream D
    is");
```

Scilab code Exa 1.5 gauss seidel method

```
clc
disp("the soln of eg 1.5--> Gauss Seidel Method");
for i=1:3,xnew(i)=2,e(i)=1
end
x=1e-6
while e(1)>x&e(2)>x&e(3)>x do
    for i=1:3, xold(i)=xnew(i),end
    xnew(1)=(44-xold(2)-2*xold(3))/10
    xnew(2)=(-2*xnew(1)+51-xold(3))/10
    xnew(3)=(-2*xnew(2)-xnew(1)+61)/10
    for i=1:3,e(i)=abs(xnew(i)-xold(i))
    end
end
disp("the values of x1,x2,x3 respectively is");
for i=1:3,disp(xnew(i));
end
```

Scilab code Exa 1.6 gauss seidel method

```
clc
disp("the soln of eg 1.6--> Gauss Seidel Method");
for i=1:3,xnew(i)=2,e(i)=1
end
```

```
x=1e-6
while e(1)>x&e(2)>x&e(3)>x do
    for i=1:3, xold(i)=xnew(i),end
    xnew(1)=(9-xold(2)+2*xold(3))/3
    xnew(2)=(xnew(1)-8+3*xold(3))/4
    xnew(3)=(xnew(2)-xnew(1)+1)/4
    for i=1:3,e(i)=abs(xnew(i)-xold(i))
    end
end
disp("the values of x1,x2,x3 respectively is");
for i=1:3,disp(xnew(i));
end
```

Chapter 2

NONLINEAR ALGEBRAIC EQUATIONS

Scilab code Exa 2.1 algebraic equations

```
//ch 2 ex 2.1 solving using newton's method.
clc
disp("the soln of eqn 2.1--> Newton Method");
x=.5                                //initial value
xnew=0
e=1
while e>10^-4 do x=xnew,function y=Fa(x),
         y=x^3-5*x+1;                          //defining fn
endfunction
der=derivative(Fa,x),                       //differentiating the fn
xnew=x-Fa(x)/der,
e=abs(xnew-x),
end
disp(xnew,"the root of the eqn is");
```

Scilab code Exa 2.2 algebraic equations

```
clc
disp("the solution of ex 2.2 --> Pressure Drop in Pipe");
meu=1.79*10^-5
rough=.0000015      //roughness
dia=.004
e_by_D=rough/dia
rho=1.23
v=50                //velocity of air
l=1
Re=(rho*v*dia)/meu          //Reynold's number
ffnew=0.01
e=1
t1=e_by_D/3.7               //term 1 of eqn.
t2=2.51/Re                  //term 2 of eqn.
disp(Re,"the Reynolds no. is");
funcprot(0)
while e>1e-6 do ff=ffnew,function y=Fh(ff),
        t3=sqrt(ff),
    y=1/t3+2*log(t1+t2/t3)/2.3,     //divide by 2.3 since log is base 'e' instead of 10
endfunction;
fdash=derivative(Fh,ff);                //f'(ff)
ffnew=ff-Fh(ff)/fdash;
e=abs(ff-ffnew)
end
disp(ff,"the fanning friction factor is")
delta_p=(ff*l*v^2*rho)/(2*dia)                  //pressure drop
disp(delta_p,"the pressure drop in pascals is");
```

Scilab code Exa 2.3 algebraic equations

```
clc
disp("the solution of eg 2.3 --> minimum
   fluidization velocity");
P=2*101325      //given data
T=298.15
M=28.97*10^-3
R=8.314
rho=(P*M)/(R*T)
rho_p=1000
dia=1.2*10^-4
ep=.42                //void fraction
sph=.88
meu=1.845*10^-5
t1=1.75*rho*(1-ep)/(sph*dia*ep^3)
                         //these are the terms of the
    function.
t2=150*meu*(1-ep)^2/(sph^2*dia^2*ep^3)
t3=(1-ep)*(rho_p-rho)*9.8
vnew=0.1
e1=1
while e1>1e-6 do v=vnew,function y=Fb(v);
    y=t1*v^2+t2*v-t3,                    //defining
       fn
    endfunction,
vdash=derivative(Fb,v),                 //
   differentiating the fn
vnew=v-Fb(v)/vdash,
e1=abs(vnew-v),
end
disp(v,"the minimum fluidisation velocity in m/s is"
   );
```

Scilab code Exa 2.4 algebraic equations

```
clc
```

```
disp("the soln of eg 2.4--> Terminal Velocity");
dia=2*10^-3
P=101325        //given data
T=298.15
M=28.89*10^-3
R=8.314
rho=(P*M)/(R*T)
rho_oil=900
meu=1.85*10^-5
Re_oil_by_v=rho*dia/meu
vnew=0.1, e=1
while e>1e-6 do v=vnew, Re_oil=Re_oil_by_v*v,
    Cd=24*(1+.15*Re_oil^.687)/Re_oil,
    vnew=sqrt(4*(rho_oil-rho)*9.81*dia/(3*Cd*rho)),
e=abs(vnew-v),
end
disp(v,"the terminal velocity in m/s is");
```

Scilab code Exa 2.5 algebraic equations

```
clc
disp("the soln of eg 2.5--> non linear equations");
xnew=0.1,ynew=0.5, e1=1, e2=1
while e1>1e-6 & e2>1e-6 do x=xnew,y=ynew,
y1=exp(x)+x*y-1,
d_fx=exp(x)+y
d_fy=x
y2=sin(x*y)+x+y-1,
d_gx=y*cos(x*y)+1
d_gy=x*cos(x*y)+1
t1=(y2*d_fy), t2=(y1*d_gy),
D1=d_fx*d_gy-d_fy*d_gx
delta_x=(t1-t2)/D1
t3=(y1*d_gx),t4=(y2*d_fx)
delta_y=(t3-t4)/D1
```

```
xnew=x+delta_x
ynew=y+delta_y
e1=abs(x-xnew)
e2=abs(y-ynew)
end
disp(y,x,"the values of x and y respectively are");
disp("such small value of x can be considered as
   zero.")
```

Scilab code Exa 2.6 algebraic equations

```
clc
disp("the soln of eg 2.6-->");
xnew=0.1,ynew=0.5, e1=1, e2=1
while e1>10^-6 & e2>10^-6 do x=xnew, y=ynew,
y1=3*x^3+4*y^2-145,
d_fx=9*x^2
d_fy=8*y
y2=4*x^2-y^3+28,
d_gx=8*x
d_gy=-3*y^2
D2=d_fx*d_gy-d_gx*d_fy
delta_x=(y2*d_fy-y1*d_gy)/D2
delta_y=(y1*d_gx-y2*d_fx)/D2
xnew=x+delta_x
ynew=y+delta_y
e1=abs(xnew-x)
e2=abs(ynew-y)
end
disp(y,x,"the values of x and y are respectively");
```

Chapter 3

CHEMICAL ENGINEERING THERMODYNAMICS

Scilab code Exa 3.1 thermodynamics

```
clc
disp("the soln of eg 3.1-->Cubic eqn. of state");
disp("all values in m3/mol");
T=373.15, P=101325, Tc=647.1, Pc=220.55*10^5, w
   =.345, R=8.314
Tr=T/Tc, Pr=P/Pc,                    //reduced
   pressure & reduced temperature
b0=.083-.422*Tr^-1.6
b1=.139-.172*Tr^-4.2
B=(b0+w*b1)*R*Tc/Pc
vnew=1
e1=1, vold=R*T/P+B
disp(vold,"the soln by virial gas eqn. of volume in
   m3/mol by Z(T,P) is");
while e1>1e-6 do vold=vnew,function y=Fh(vold),
        y=P*vold/(R*T)-1-B/vold
endfunction;
ydash=derivative(Fh,vold);
vnew=vold-Fh(vold)/ydash;
```

```
e1=abs(vold-vnew)
end
disp(vold,"the soln by virial gas eqn. of volume in m3/mol by Z(T,V) is");
//by peng robinson method
k=.37464+1.54226*w-.26992*w^2
s=1+k*(1-Tr^.5)
lpha=s^2
a=.45724*R^2*Tc^2*lpha/Pc
b=.0778*R*Tc/Pc
vnew=b,e2=1,
vol=b, fe=0, fd=0
disp("the volume of saturated liq. and saturated vapour by peng-robinson method respectively is")
for i=0:2 do
        vol=vnew
        y2=vol^3*P+vol^2*(P*b-R*T)-vol*(3*P*b^2+2*b*R*T-a)+P*b^3+b^2*R*T-a*b
ydash2=3*P*vol^2+(P*b-R*T)*2*vol-(3*P*b^2+2*b*R*T-a)
vnew=vol-y2/ydash2,
e2=abs(vol-vnew)
if i==1 & e2<1e-6 then fd=vnew,vnew=R*T/P,
end
end
disp(vol,fd);
funcprot(0)
//by redlich-kwong method
i=0
a=.42748*R^2*Tc^2.5/Pc
b=.08664*R*Tc/Pc
Vnew=b,e3=1
disp("the vol of saturated liq. and vapour by redlich kwong method respectively are");
for i=0:2 do V=Vnew,function y3=gh(V),
            y3=V^3*P-R*T*V^2-V*(P*b^2+b*R*T-a/sqrt(T))-a*b/sqrt(T)
        endfunction,
        deriv=derivative(gh,V);
```

```
          Vnew=V-gh(V)/deriv;
          e3=abs(Vnew-V)
          if i==1 & e3<1e-6 then de=Vnew,Vnew=R*T/P
                            //for saturated liq.
          end
        end
    disp(V,de);
//vander waals method
i=0
a=27*R^2*Tc^2/(64*Pc)
b=R*Tc/(8*Pc)
vnew=b, v=b,e=1
for i=0:2 do v=vnew,function v3=bh(v),
        v3=v^3*P-v^2*(P*b+R*T)+a*v-a*b,
    endfunction
    deriva=derivative(bh,v),
    vnew=v-bh(v)/deriva
    e4=abs(v-vnew),
    if i==1 & e4<10^-6 then sol=vnew,vnew=R*T/P
    end
end
disp("the vol of saturated liq. and vapour by vander
    waal method respectively are");
disp(vnew,sol);
```

Scilab code Exa 3.2 thermodynamics

```
clc
disp("the soln of eg 3.2--> BPT & DPT Calc.");
z1=.5,P=101.325
                                        //given data
a1=14.3916, b1=2795.32,c1=230.002,
a2=16.262,b2=3799.887,c2=226.346
x1=z1, x2=1-z1
T1sat=b1/(a1-log(P))-c1
```

```
T2sat=b2/(a2-log(P))-c2
Told=273,e=1
Tnew=x1*T1sat+x2*T2sat
while e>1e-6 do Told=Tnew,function y1=fw(Told),
        P1sat=exp(a1-b1/(Told+c1)),
        P2sat=exp(a2-b2/(Told+c2)),
        y1=P-x1*P1sat-x2*P2sat
        endfunction
        ydashd=derivative(fw,Told)
        Tnew=Told-fw(Told)/ydashd
        e=abs(Told-Tnew)
    end
    disp(Tnew,"the bubble point temp. in Celsius is");
                                                     //calc of dew point
y1=z1, y2=1-z1,e=1
Tnew=y1*T1sat+y2*T2sat
while e>1e-6 do Told=Tnew,function y11=fw1(Told),
         P1sat=exp(a1-b1/(Told+c1)),
         P2sat=exp(a2-b2/(Told+c2)),
        y11=1/P-y1/P1sat-y2/P2sat
        endfunction
        ydashd=derivative(fw1,Told)
        Tnew=Told-fw1(Told)/ydashd
        e=abs(Told-Tnew)
    end
    disp(Tnew,"the dew point temp. in Celsius is");
```

Scilab code Exa 3.3 flash calculations using Raoult law

```
clc
disp("the soln of eg 3.3-->Flash Calc. using Raoults Law")
psat(1)=195.75,psat(2)=97.84,psat(3)=50.32                //given data
z(1)=.3,z(2)=.3,z(3)=.4
bpp=z(1)*psat(1)+z(2)*psat(2)+z(3)*psat(3)                //Bubble point pressure
dpp=1/(z(1)/psat(1)+z(2)/psat(2)+z(3)/psat(3))            //dew point pressure
disp(dpp,bpp,"the bubble point pressure and dew point pressure respectively are");
P=100,v=1, vnew=1, e=1,y2=0, der=0                        //given pressure is between BPP & DPP
for i=1:3 k(i)=psat(i)/P
end
while e>1e-6 do v=vnew,for i=1:3,
        t1=(1-v+v*k(i)),y2=y2+z(i)*(k(i)-1)/t1,
        der=der-z(i)*(k(i)-1)^2/t1^2,
        end
    vnew=v-y2/der,e=abs(vnew-v),
    end
    for i=1:3, x(i)=z(i)/(1-v+v*k(i)), y(i)=x(i)*k(i)
    end
    disp(x(1),"mol frctn of acetone in liq. phase is");
    disp(y(1),"mol frctn of acetone in vapour phase is");
    disp(x(2),"mol frctn of acetonitrile in liq. phase is");
    disp(y(2),"mol frctn of acetonitrile in vapour.
```

```
    phase is");
  disp(x(3),"mol frctn of nitromethane in liq.
    phase is");
  disp(y(3),"mol frctn of nitromethane in vapour
    phase is");
```

Scilab code Exa 3.4 BPT and DPT calculation using modified raoult law

```
a12=437.98*4.186, a21=1238*4.186, v1=76.92, v2=18.07
                                                       //
                                                         ca
                                                         of
                                                         BP
clc
disp("the soln of eg 3.4-->");
t=100
x1=.5, R=8.314
a1=16.678,b1=3640.2,c1=219.61
a2=16.2887,b2=3816.44,c2=227.02
x2=1-x1
p1sat=exp(a1-b1/(c1+t))
p2sat=exp(a2-b2/(c2+t))
h12=v2*exp(-a12/(R*(t+273.15)))/v1
h21=v1*exp(-a21/(R*(t+273.15)))/v2
m=h12/(x1+x2*h12)-h21/(x2+x1*h21)
g1=exp(-log(x1+x2*h12)+x2*m)
g2=exp(-log(x2+x1*h21)-x1*m)
p=x1*g1*p1sat+x2*g2*p2sat
disp(p,"boiling point pressure in kPa is");
                                                       //
                                                         ca
```

```
of
BP
p=101.325,x1=.5, e=1
x2=1-x1
t1sat=b1/(a1-log(p))-c1
t2sat=b2/(a2-log(p))-c2
tnew=x1*t1sat+x2*t2sat
while e>10^-4 do told=tnew,
    p1sat=exp(a1-b1/(c1+told)),p2sat=exp(a2-b2/(c2+
       told)),
    p1sat=p/(g1*x1+g2*x2*(p2sat/p1sat))
    tnew=b1/(a1-log(p1sat))-c1,
    e=abs(tnew-told)
end
disp(tnew,"boiling point temperature in Celsius is")
   ;
                                                               //
                                                                 cal
                                                                 of
                                                                 dpp
e1=1, e2=1, e3=1, pold=1
t=100,y1=.5
y2=1-y1
p1sat=exp(a1-b1/(c1+t))
p2sat=exp(a2-b2/(c2+t))
g1=1, g2=1, g11=1, g22=1
pnew=1/(y1/(g1*p1sat)+y2/(g2*p2sat))
while e1>.0001 do pold=pnew, while e2>.0001& e3
   >.0001 do g1=g11,g2=g22,
                              x1=y1*pold/(g1*p1sat)
                              x2=y2*pold/(g2*p2sat)
                              x1=x1/(x1+x2)
                              x2=1-x1
```

```
                              h12=v2*exp(-a12/(R*(t+273.15)))/v1
                              h21=v1*exp(-a21/(R*(t+273.15)))/v2
                              m=h12/(x1+x2*h12)-h21/(x2+x1*h21)
                              g11=exp(-log(x1+x2*h12)+x2*m)
                              g22=exp(-log(x2+x1*h21)-x1*m)
                              e2=abs(g11-g1), e3=abs(g22-g2)
                          end
       pnew=1/(y1/(g1*p1sat)+y2/(g2*p2sat))
       e1=abs(pnew-pold)
    end
   disp(pnew,"dew point pressure in kPa is");
                                                                  //calc dpt
p=101.325,y1=.5, e4=1, e5=1,e6=1
y2=1-y1
t1sat=b1/(a1-log(p))-c1
t2sat=b2/(a2-log(p))-c2
tnew=y1*t1sat+y2*t2sat
g11=1, g22=1
while e4>.0001 do told=tnew,
    p1sat=exp(a1-b1/(c1+told))
    p2sat=exp(a2-b2/(c2+told)), while e5>.0001 & e6>.0001 do g1=g11, g2=g22,
    x1=y1*p/(g1*p1sat)
    x2=y2*p/(g2*p2sat)
    x1=x1/(x1+x2)
    x2=1-x1
    h12=v2*exp(-a12/(R*(t+273.15)))/v1
    h21=v1*exp(-a21/(R*(t+273.15)))/v2
```

```
                                m=h12/(x1+x2*h12)-h21/(x2+x1*h21)
                                g11=exp(-log(x1+x2*h12)+x2*m)
                                g22=exp(-log(x2+x1*h21)-x1*m)
                                e5=abs(g11-g1), e6=abs(g22-g2)
                            end

       p1sat=p*(y1/g1+y2*p1sat/(g2*p2sat))
       tnew=b1/(a1-log(p1sat))-c1
       e4=abs(tnew-told)
    end
   disp(tnew,"dew point temperature  in Celsius is");
```

Scilab code Exa 3.5 flash calculations using modified Raoult law

```
clc
disp("the soln of eg 3.5-->Flash calc. using Modified Raoult law");
a12=292.66*4.18, a21=1445.26*4.18, v1=74.05*10^-6, v2=18.07*10^-6, R=8.314
t=100,z1=.3,
z2=1-z1
a1=14.39155, a2=16.262, b1=2795.82, b2=3799.89, c1=230.002, c2=226.35
e1=1,e2=1,e3=1,e4=1,e5=1,e6=1,vnew=0
                                                    //calc of BPP
x1=z1, x2=z2
p1sat=exp(a1-(b1/(t+c1)))
p2sat=exp(a2-(b2/(t+c2)))
```

```
h12=v2*exp(-a12/(R*(t+273.15)))/v1
h21=v1*exp(-a21/(R*(t+273.15)))/v2
m=(h12/(x1+x2*h12))-(h21/(x2+x1*h21))
g1=exp(-log(x1+x2*h12)+x2*m)
g2=exp(-log(x2+x1*h21)-x1*m)
p=x1*g1*p1sat+x2*g2*p2sat
disp(p,"the bubble point pressure is");
bpp=p, gb1=g1, gb2=g2                                  //g1
   & g2 are activity co-efficients
                                                       //
   calc
   of
   DPP
y1=z1, y2=z2
g1=1, g2=1
pnew=1/(y1/(g1*p1sat)+y2/(g2*p2sat))
g1n=g1, g2n=g2
while e1>.0001 do pold=pnew,while e2>.0001& e3>.0001
    do g1=g1n, g2=g2n,
    x1=y1*pold/(g1*p1sat)
    x2=y2*pold/(g2*p2sat)
    x1=x1/(x1+x2)
    x2=1-x1
    g1n=exp(-log(x1+x2*h12)+x2*m)
    g2n=exp(-log(x2+x1*h21)-x1*m)
    e2=abs(g1n-g1), e3=abs(g2n-g2)
end
pnew=1/(y1/(g1n*p1sat)+y2/(g2n*p2sat))
e1=abs(pnew-pold)
end
disp(pnew,"the dew point pressure is");
dpp=pnew, gd1=g1n, gd2=g2n
p=200
v=(bpp-p)/(bpp-dpp)
g1=((p-dpp)*(gb1-gd1))/(bpp-dpp)+gd1
```

```
g2=((p-dpp)*(gb2-gd2))/(bpp-dpp)+gd2

//calc of distribution co-efficients
while e4>.0001 & e5>.0001 do g1n=g1,g2n=g2,
    k1=g1n*p1sat/p
    k2=g2n*p2sat/p
    while e6>.00001 do v=vnew,
        function f=fv(v),
        y1=(k1*z1)/(1-v+v*k1)
        y2=(k2*z2)/(1-v+v*k2)
        x1=y1/k1
        x2=y2/k2
        f=y1-x1+y2-x2
        endfunction
        derv=derivative(fv,v)
        vnew=v-fv(v)/derv
        e6=abs(vnew-v)
    end
    h12=v2*exp(-a12/(R*(t+273.15)))/v1
    h21=v1*exp(-a21/(R*(t+273.15)))/v2
    m=(h12/(x1+x2*h12))-(h21/(x2+x1*h21))
    g1=exp(-log(x1+x2*h12)+x2*m)
    g2=exp(-log(x2+x1*h21)-x1*m)
    e4=abs(g1-g1n), e5=abs(g2-g2n)
end
disp(v,"the no. of moles in vapour phase is");
disp(y1,x1,"x1 and y1 respectively are");
```

Scilab code Exa 3.6 vapour pressure calculation using cubic equation of state

```
clc
disp("the soln of eg 3.6-->Vapour Pressure using
   Cubic Eqn. of State")
t=373.15, tc=647.1, pc=220.55*10^5, w=.345,R=8.314
```

```
               //given
f1=1, e1=1, e2=1, vnew=1, pnew=1              //assumed values
k=.37464+1.54226*w-.26992*w*2
s=(1+k*(1-(t/tc)^.5))^2,
a=.45724*R*R*tc*tc*s/pc
b=.0778*R*tc/pc
//calc of vol. of liq.
while f1>10^-4 do p=pnew,vnew=b,
t1=(p*b-R*t)                              //co-efficients of vol. in the eqn.
t2=3*p*b^2+2*b*R*t-a
t3=p*b^3+b^2*R*t-a*b
    while e1>1e-6 & vnew>0 do vold=vnew,
          y1=vold^3*p+vold^2*t1-vold*t2+t3,
          der=3*vold^2*p+2*vold*t1-t2
          vnew=vold-y1/der
          e1=abs(vnew-vold)
end
vliq=vold,
//fugacity of liq.
zliq=p*vliq/(R*t)
c=(a/(2*1.414*b*R*t))*(log((vliq+(1+sqrt(2))*b)/(vliq+(1-sqrt(2))*b))),
t5=zliq-p*b/(R*t)
fl2=p*exp(zliq-1-log(t5)-c),
vvnew=R*t/p,                          //assumed value close to the ideal value
//calc of vol. of vapour
while e2>1e-6 do vvold=vvnew,
            x2=vvold^3*p+vvold^2*t1-vvold*t2+t3,
          der1=3*vvold^2*p+2*vvold*t1-t2,
    vvnew=vvold-x2/der1
e2=abs(vvnew-vvold)
end
//fugacity of vapour
vvap=vvold,zvap=p*vvap/(R*t),
d=(a/(2*sqrt(2)*b*R*t))*(log((zvap+(1+sqrt(2))*b*p/(
```

```
R*t))/(zvap+(1-sqrt(2))*p*b/(R*t))))
t6=zvap-p*b/(R*t)
fv2=p*exp(zvap-1-log(t6)-d)
pnew=p*fl2/fv2                                //updating the value of P
f1=abs((fl2-fv2)/fv2)
end
disp(p,"the vapour pressure of water in Pa is");
```

Scilab code Exa 3.7 pressure x y diagram using gamma phi approach

```
clc
disp("the soln of eg 3.7-->P-x-y calc. using Gamma-Phi approach");
et=1,er=1,sold=0, snew=0                      //assumed values
R=8.314,t=100, x1=.958, a12=107.38*4.18, a21=469.55*4.18,
tc1=512.6,tc2=647.1,pc1=80.97*10^5,pc2=220.55*10^5,w1=.564,w2=.345,zc1=.224,zc2=.229,v1=40.73*10^-6, v2=18.07*10^-6                  //given data
x2=1-x1,
a1=16.5938, a2=16.262, b1=3644.3, b2=3799.89, c1=239.76, c2=226.35
p1sat=exp(a1-b1/(c1+t))*1000                  //Saturation Pressure
p2sat=exp(a2-b2/(c2+t))*1000
t=t+273.15                                    //Temp in Kelvin req.
h12=(v2/v1)*exp(-a12/(R*t))
h21=(v1/v2)*exp(-a21/(R*t))
z=h12/(x1+x2*h12)-h21/(x2+x1*h21)
g1=exp(-log(x1+x2*h12)+x2*z)                  //Activity Co-efficients
```

```
g2=exp(-log(x2+x1*h21)-x1*z)
tr1=t/tc1                                   //Reduced Temp.
b0=.083-.422*tr1^-1.6
b1=.139-.172*tr1^-4.2
b11=(R*tc1/pc1)*(b0+b1*w1)
tr2=t/tc2
b0=.083-.422*tr2^-1.6
b1=.139-.172*tr2^-4.2
b22=(R*tc2/pc2)*(b0+b1*w2)
w12=(w1+w2)^.5
tc12=(tc1*tc2)^.5
zc12=(zc1+zc2)^.5
vc1=zc1*R*tc1/pc1, vc2=zc2*R*tc2/pc2
vc12=((vc1^.33+vc2^.33)/2)^3
pc12=zc12*R*tc12/vc12
tr12=t/tc12
b0=.083-.422*tr12^-1.6
b1=.139-.172*tr12^-4.2
b12=(R*tc12/pc12)*(b0+b1*w12)
d12=2*b12-b11-b22
p=x1*g1*p1sat+x2*p2sat*g2
y1=x1*g1*p1sat/p, y2=x2*g2*p2sat/p
pnew=p,
//calc of Pressure
while et>1e-6 do p=pnew,
    f1=p1sat*(exp(b11*p1sat/(R*t)))*(exp((v1*(p-p1sat)/(R*t))))
f2=p2sat*(exp(b22*p2sat/(R*t)))*(exp((v2*(p-p2sat)/(R*t))))
while er>1e-6 do sold=snew,
    fc1=exp((p/(R*t))*(b11+y2^2*d12))
 fc2=exp((p/(R*t))*(b22+y1^2*d12))
    k1=g1*f1/(fc1*p)
k2=g2*f2/(fc2*p)
snew=x1*k1+x2*k2
y1=x1*k1/snew
y2=x2*k2/snew
```

```
er=abs(snew-sold)
end
pnew=(x1*g1*f1/fc1)+(x2*g2*f2/fc2)
y1=x1*g1*f1/(fc1*pnew)
y2=x2*g2*f2/(fc2*pnew)
et=abs(pnew-p)
end
disp(p,y1,"the amt. of methanol in vapour phase  and
    system pressure in Pa respectively are")
```

Scilab code Exa 3.9 chemical reaction engineering 2 simultaneous reactions

```
clc
disp("the soln of eg 3.9-->Chemical Reaction
   Equilibrium-2 Simultaneous Reactions")
//let x1 and x2 be the reaction co-ordinate for 1st
   and 2nd reactions
x1new=.9, x2new=.6,r1=1,r2=1                    //assumed
    values
Kp=1                //since P=1 atm
K1=.574, K2=2.21                //given
Kye1=K1, Kye2=K2                //at eqm.
while r1>1e-6 & r2>1e-6, x1=x1new,x2=x2new,
m_CH4=1-x1,m_H2O=5-x1-x2,m_CO=x1-x2,m_H2=3*x1+x2,
   m_CO2=x2       //moles of reactants and products
   at eqm.
total=m_CO2+m_H2+m_CO+m_H2O+m_CH4
Ky1=m_CO*m_H2^3/(m_CH4*m_H2O*total^2)
Ky2=m_CO2*m_H2/(m_CO*m_H2O)
    f1=Ky1-.574                              //1st
       function in x1 and x2
    f2=Ky2-2.21                              //2nd
       function in x1 and x2
    d3=((3*x1+x2)^2*(12*x1-8*x2))/((1-x1)*(5-x1-x2)
```

```
*(6+2*x1)^2)
    d4=(3*x1+x2)^3*(x1-x2)*(8*x1^2+6*x1*x2-24*x1+2*x2-16)
    d5=((1-x1)^2)*((5-x1-x2)^2)*((6+2*x1)^3)
    df1_dx1=d3-(d4/d5)                              //df1/dx1- partial derivative of f1 wrt to x1
    d6=3*(x1-x2)*((3*x1+x2)^2)-(3*x1+x2)^3
    d7=(1-x1)*(5-x1-x2)*((6+x1*2)^2)
    d8=((x1-x2)*(3*x1+x2)^3)/((1-x1)*((5-x1-x2)^2)*(6+2*x1)^2)
    df1_dx2=(d6/d7)+d8                              //df1/dx2- partial derivative of f1 wrt to x2
    d9=(x1-x2)*(5-x1-x2)
    df2_dx1=3*x2/d9-(x2*(3*x1+x2)*(5-2*x1))/(d9^2)          //df1/dx2- partial derivative of f1 wrt to x2
    d10=(3*x1+2*x2)/d9
    d11=x2*(3*x1+x2)*(2*x2-5)/(d9^2)
    df2_dx2=d10-d11                                  //df1/dx2- partial derivative of f1 wrt to x2
    dm=df1_dx1*df2_dx2-df1_dx2*df2_dx1
    delta_x1=(f2*df1_dx2-f1*df2_dx2)/dm
    delta_x2=(f1*df2_dx1-f2*df1_dx1)/dm
x1new=x1+delta_x1                                //updating the values of x1 & x2
x2new=x2+delta_x2
r1=abs(x1-x1new), r2=abs(x2new-x2)
end
disp(x2,x1,"the value of X1 and X2 respectively is");
m_CH4=1-x1,m_H2O=5-x1-x2,m_CO=x1-x2,m_H2=3*x1+x2,m_CO2=x2       //moles of reactants and products at eqm.
total=m_CO2+m_H2+m_CO+m_H2O+m_CH4
disp(m_CO2,m_H2,m_CO,m_H2O,m_CH4,"the moles at eqm of CH4,H2O,CO,H2,CO2 are")
disp(total,"total number of moles at eqm. is")
```

Scilab code Exa 3.10 adiabatic flame temperature

```
clc
disp("the soln of eg 3.10-->Adiabatic Flame
    Temperature");
u1=1,u2=3.5,u3=2,u4=3                          //moles
    given 1-C2H6, 2-O2, 3-CO2, 4-H2O
a1=1.648,a2=6.085,a3=5.316,a4=7.7
b1=4.124e-2,b2=.3631e-2,b3=1.4285e-2,b4=.04594e-2
c1=-1.53e-5,c2=-.1709e-5,c3=-.8362e-5,c4=.2521e-5
d1=1.74e-9,d2=.3133e-9,d3=1.784e-9,d4=-.8587e-9
n1=1,n2=4,n3=10,n4=0,t0=298.15,t1=25,e1=1
t1=t1+273.15
//calc. of sum of co-efficients of heat capacity of
    the rxn.
sa=n1*a1+n2*a2+n3*a3+n4*a4
sb=n1*b1+n2*b2+n3*b3+n4*b4
sc=n1*c1+n2*c2+n3*c3+n4*c4
sd=n1*d1+n2*d2+n3*d3+n4*d4
da=u4*a4+u3*a3-u2*a2-u1*a1
db=u4*b4+u3*b3-u2*b2-u1*b1
dc=u4*c4+u3*c3-u2*c2-u1*c1
dd=u4*d4+u3*d3-u2*d2-u1*d1
h0=(u4*(-57.798)+u3*(-94.05)-u2*0-u1*(-20.236))*1000
                        //enthalpy of the rxn.
tnew=1000
while e1>1e-6 do t=tnew,
    function f=ft(t),
    f=sa*(t-t1)+(sb/2)*(t^2-t1^2)+(sc/3)*(t^3-t1^3)
        +(sd/4)*(t^4-t1^4)+h0+da*(t-t0)+(db/2)*(t^2-
        t0^2)+(dc/3)*(t^3-t0^3)+(dd/4)*(t^4-t0^4)
endfunction
dr=derivative(ft,t),
tnew=t-ft(t)/dr,
```

```
e1=abs((tnew-t)/tnew)
end
disp(tnew,"the adiabatic flame temp in K is");
```

Chapter 4

INITIAL VALUE PROBLEMS

Scilab code Exa 4.1 solution of ordinary differential equation

```
clc
disp("the solution of e.g. 4.1 —>Integration of
   Ordinary Differential Equation")
// in this problem dy/dx=x+y
x_0=0                   //initial values given
y_0=0

function ydash=fs(x,y),
    ydash=x+y,
endfunction

for x_0=0:0.1:0.2,
    h=0.1                                        //step
       increment of 0.1
    f_0=fs(x_0,y_0)
    k1=h*f_0
    k2=h*fs(x_0+h/2,y_0+k1/2)
    k3=h*fs(x_0+h/2,y_0+k2/2)
    k4=h*fs(x_0+h,y_0+k3)
    y_0=y_0+(k1+2*k2+2*k3+k4)/6
end
```

```
y_0=y_0-(k1+2*k2+2*k3+k4)/6                          //to
    get value at x=0.2
disp(y_0,"the value of y at x=.2 using Runge Kutta
    method is");
ae=exp(x_0)-x_0-1
    //analytical eqn given
disp(ae,"the value of y at x=0.2 from analytical eqn
    is");
```

Scilab code Exa 4.2 solution of ordinary differential equation

```
clc
disp("the solution of e.g. 4.2 -->Ordinary
    Differential Eqn.-Runge Kutta method")
// in this problem dy/dx=-y/(1+x)
x_0=0                    //initial values given
y_0=2
function ydash=fr(x,y),
    ydash=-y/(1+x),
endfunction
for x_0=0:0.01:2.5,
    h=0.01                                        //step
        increment of 0.01
    f_0=fr(x_0,y_0)
    k1=h*f_0
    k2=h*fr(x_0+h/2,y_0+k1/2)
    k3=h*fr(x_0+h/2,y_0+k2/2)
    k4=h*fr(x_0+h,y_0+k3)
    y_0=y_0+(k1+2*k2+2*k3+k4)/6
end
y_0=y_0-(k1+2*k2+2*k3+k4)/6                          //
    final value at x=2.5
disp(y_0,"the value of y at x=2.5 using Runge Kutta
    method is");
```

Scilab code Exa 4.3 double pipe heat exchanger

```
clc
disp("the solution of e.g. 4.3 -->Double Pipe Heat Exchanger");
rho=1000, v=1, dia=2.4*10^-2, Cp=4184                   //given data
mdot=rho*v*%pi*dia^2/4
t1=mdot*Cp
U=200
Ts=250
z=0                  //initial values given
// dT/dz=U*pi*dia*(Ts-T)/(mdot*Cp)
function Tgrad=fr(z,T),
    Tgrad=U*%pi*dia*(Ts-T)/(mdot*Cp),
endfunction
T=20
for z=0:0.01:10,
    h=0.01                                      //step increment of 0.01
    k1=h*fr(z,T)
    k2=h*fr(z+h/2,T+k1/2)
    k3=h*fr(z+h/2,T+k2/2)
    k4=h*fr(z+h,T+k3)
    T=T+(k1+2*k2+2*k3+k4)/6
    if z==5 then T=T-(k1+2*k2+2*k3+k4)/6,
        disp(T,"the value of T in deg Celsius at z=5 m using Runge Kutta method is");
    end
end
T=T-(k1+2*k2+2*k3+k4)/6                         //final value at z=10 m
disp(T,"the value of T in deg Celsius at z=10 m using Runge Kutta method is");
```

Scilab code Exa 4.4 stirred tank with coil heater

```
clc
disp("the solution of e.g. 4.4 -->Stirred Tank with Coil Heater")
vol=.5*.5*2                                //given data
rho=1000
m=vol*rho
vol_rate=.001
mdot=vol_rate*rho
out_A=1
U=200
Cp=4184
T1=20, Ts=250, T_exit=28    //temp in Celsius
t1=(mdot*Cp*T1+U*out_A*Ts)/(m*Cp)    //terms of the function
t2=(mdot*Cp+U*out_A)/(m*Cp)
//dt/dt=(mdot*Cp(T1-T)+Q_dot)/m*Cp
function tgrad=fv(tm,T),
    tgrad=t1-t2*T
endfunction
T=20                                             //initial value
funcprot(0)
for tm=0:1:5000,
    h=1                                      //step increment of 1 sec
    k1=h*fv(tm,T)
    k2=h*fv(tm+h/2,T+k1/2)
    k3=h*fv(tm+h/2,T+k2/2)
    k4=h*fv(tm+h,T+k3)
    e1=abs(T-T_exit)
    if e1<1e-3 then disp(tm,"the time at which exit
```

```
temp. in sec. is 28 C is")
    end
    T=T+(k1+2*k2+2*k3+k4)/6
    end
T=T-(k1+2*k2+2*k3+k4)/6                    //final steady state temp.
disp(T,"the value of steady Temp in Celsius is");
```

Scilab code Exa 4.5 stirred tank with coil heater

```
clc
disp("the solution of e.g. 4.5 -->Stirred Vessel with Coil Heater");
m=760                                          //given data
mdot=12/60
U_into_A=11.5/60
Cp=2.3
T1=25, Ts=150
t1=(mdot*Cp*T1+U_into_A*Ts)/(m*Cp)
t2=(mdot*Cp+U_into_A)/(m*Cp)
//using energy balance we know accumulation=input-output
//T is the temp. of fluid in stirred tank
function tgrade=fg(t,T);
     tgrade=(t1-t2*T),
endfunction
T=25
for t=0:1:3000,
    h=1                                        //step increment of 1 sec
    k1=h*fg(t,T)
    k2=h*fg(t+h/2,T+k1/2)
    k3=h*fg(t+h/2,T+k2/2)
    k4=h*fg(t+h,T+k3)
```

```
    T=T+(k1+2*k2+2*k3+k4)/6
    end
T=T-(k1+2*k2+2*k3+k4)/6                    //to get
   value at x=0.2
disp(T,"the value of T in deg C after 50 mins is");
T_steady=(mdot*Cp*T1+U_into_A*Ts)/(mdot*Cp+U_into_A)
disp(T_steady,"the steady state temp in deg C is");
```

Scilab code Exa 4.6 pneumatic conveying

```
clc
disp("the soln of eg 4.6-->Pneumatic Conveying")
dia=3*10^-4                                //given data
v_sprfcl=12
rho_p=900
meu=1.8*10^-5
P=101325
T=298.15
R=8.314
M=28.84*10^-3
rho_air=P*M/(R*T)
proj_A=%pi*(dia^2)/4
volm=%pi*(dia^3)/6
t1=rho_air*proj_A/(volm*rho_p)
                                   //terms of the function
t2=((rho_air/rho_p)-1)*9.81*2
y=0
for z=.01:.01:10,
    h=.01
    vn1=sqrt(y)
    Re=rho_air*(12-vn1)*dia/meu
    Cd=24*(1+.15*Re^.687)/Re
    function dy_by_dz=fy(z,y),
    dy_by_dz=t1*Cd*(12-sqrt(y))^2+t2,
endfunction
```

```
    kk1=h*fy(z,y)
    kk2=h*fy(z+h/2,y+kk1/2)
    kk3=h*fy(z+h/2,y+kk2/2)
    kk4=h*fy(z+h,y+kk3)
    y=y+(kk1+2*kk2+2*kk3+kk4)/6
    end
v=sqrt(y)                    //final value of velocity
disp(v,"the value of v at the end of the pneumatic conveyor is");
```

Scilab code Exa 4.7 simultaneous ordinary differential equations

```
clc
disp("the soln of eg 4.7-->Simultaneous O.D.E.")
function dx_dt=fw(t,x,y);
         dx_dt=x+2*y,
endfunction
function dy_dt=fq(t,x,y);
         dy_dt=3*x+2*y
endfunction
y=4,x=6                           //initial values
//solving by Runge-Kutta method
for t=0:.1:.2,
    h=.1                                    //step increment of 0.1
    k1=h*fw(t,x,y)
    l1=h*fq(t,x,y)
    k2=h*fw(t+h/2,x+k1/2,y+l1/2)
    l2=h*fq(t+h/2,x+k1/2,y+l1/2)
    k3=h*fw(t+h/2,x+k2/2,y+l2/2)
    l3=h*fq(t+h/2,x+k2/2,y+l2/2)
    k4=h*fw(t+h,x+k3,y+l3)
    l4=h*fq(t+h,x+k3,y+l3)
    x=x+(k1+2*k2+2*k3+k4)/6
```

```
    y=y+(l1+2*l2+2*l3+l4)/6
    end
x=x-(k1+2*k2+2*k3+k4)/6
y=y-(l1+2*l2+2*l3+l4)/6
disp(y,x,"the values of x and y repectively are");
t_an=.2
x_an=4*exp(4*t)+2*exp(-t)
y_an=6*exp(4*t)-2*exp(-t)
disp(y_an,x_an,"the analytical values of x and y are respectively");
```

Scilab code Exa 4.8 simultaneous ordinary differential equations

```
clc
disp("the soln of eg 4.8-->Simultaneous O.D.E.")
function dy_dx=fw(x,y,z);
         dy_dx=z,
endfunction
function dz_dx=fq(x,y,z);
         dz_dx=-y
endfunction
y=2,z=1                              //initial values
for x=0:.1:3,
    h=.1                                         //step increment of 0.1
    k1=h*fw(x,y,z)
    l1=h*fq(x,y,z)
    k2=h*fw(x+h/2,y+k1/2,z+l1/2)
    l2=h*fq(x+h/2,y+k1/2,z+l1/2)
    k3=h*fw(x+h/2,y+k2/2,z+l2/2)
    l3=h*fq(x+h/2,y+k2/2,z+l2/2)
    k4=h*fw(x+h,y+k3,z+l3)
    l4=h*fq(x+h,y+k3,z+l3)
    y=y+(k1+2*k2+2*k3+k4)/6
    z=z+(l1+2*l2+2*l3+l4)/6
```

```
    end
y=y-(k1+2*k2+2*k3+k4)/6
z=z-(l1+2*l2+2*l3+l4)/6
disp(z,y,"the values of y and z respectively are");
// for the given analytical eqns the values of A and alpha can be determined using initial values of y and z
alpha=atan(2)
A=2/sin(alpha)
y_an=A*sin(alpha+x)
z_an=A*cos(alpha+x)
disp(z_an,y_an,"the analytical values of y and z are");
```

Scilab code Exa 4.9 simultaneous ordinary differential equations

```
clc
disp("the soln of eg 4.9-->Simultaneous O.D.E.")
function dy_dx=fw(x,y,z);                    //let us have dy/dx=z, therefore d2y/dx2=dz/dx
         dy_dx=z,
endfunction
function dz_dx=fq(x,y,z);
         dz_dx=-y*x,
endfunction
y=2,z=1
for x=0:.1:3,
    h=.1                                     //step increment of 0.1
    k1=h*fw(x,y,z)
    l1=h*fq(x,y,z)
    k2=h*fw(x+h/2,y+k1/2,z+l1/2)
    l2=h*fq(x+h/2,y+k1/2,z+l1/2)
    k3=h*fw(x+h/2,y+k2/2,z+l2/2)
    l3=h*fq(x+h/2,y+k2/2,z+l2/2)
```

```
    k4=h*fw(x+h,y+k3,z+l3)
    l4=h*fq(x+h,y+k3,z+l3)
    y=y+(k1+2*k2+2*k3+k4)/6
    z=z+(l1+2*l2+2*l3+l4)/6
    end
y=y-(k1+2*k2+2*k3+k4)/6
z=z-(l1+2*l2+2*l3+l4)/6
disp(z,y,"the values of y and z repectively are");
```

Scilab code Exa 4.10 series of stirred tank with coil heater

```
clc
disp("the solution of eg 4.10 -->Series of Stirred
   Tanks with Coil Heaters")

Cp=2000,A=1,U=200,m=1000,mdot=2,Ts=250
                   //given data
T0=20, T1=0, T2=0, T3=0

//from energy balances for the tanks we have
   accumulation=inlet-outlet
T1_steady=(mdot*Cp*(T0)+U*A*(Ts))/(mdot*Cp+U*A)
disp(T1_steady,"the steady state temperature of tank
    1 is");
T2_steady=(mdot*Cp*(T1_steady)+U*A*(Ts))/(mdot*Cp+U*
   A)
disp(T2_steady,"the steady state temperature of tank
    2 is");
T3_steady=(mdot*Cp*(T2_steady)+U*A*(Ts))/(mdot*Cp+U*
   A)
disp(T3_steady,"the steady state temperature of tank
    3 is");
final_T3=.99*T3_steady
function dT1_by_dt=f1(t,T1,T2,T3),
    dT1_by_dt=(mdot*Cp*(T0-T1)+U*A*(Ts-T1))/(m*Cp),
```

```
endfunction
function dT2_by_dt=f2(t,T1,T2,T3),
    dT2_by_dt=(mdot*Cp*(T1-T2)+U*A*(Ts-T2))/(m*Cp),
endfunction
function dT3_by_dt=f3(t,T1,T2,T3),
    dT3_by_dt=(mdot*Cp*(T2-T3)+U*A*(Ts-T3))/(m*Cp),
endfunction
T1=20,T2=20,T3=20
//solving by Newton's Method
for t=0:1:10000,
    h=1                                    //step
       increment of 1
    k1=h*f1(t,T1,T2,T3)
    l1=h*f2(t,T1,T2,T3)
    m1=h*f3(t,T1,T2,T3)
    k2=h*f1(t+h/2,T1+k1/2,T2+l1/2,T3+m1/2)
    l2=h*f2(t+h/2,T1+k1/2,T2+l1/2,T3+m1/2)
    m2=h*f3(t+h/2,T1+k1/2,T2+l1/2,T3+m1/2)
    k3=h*f1(t+h/2,T1+k2/2,T2+l2/2,T3+m2/2)
    l3=h*f2(t+h/2,T1+k2/2,T2+l2/2,T3+m2/2)
    m3=h*f3(t+h/2,T1+k2/2,T2+l2/2,T3+m2/2)
    k4=h*f1(t+h,T1+k3,T2+l3,T3+m3)
    l4=h*f2(t+h,T1+k3,T2+l3,T3+m3)
    m4=h*f3(t+h,T1+k3,T2+l3,T3+m3)
    T1=T1+(k1+2*k2+2*k3+k4)/6
    T2=T2+(l1+2*l2+2*l3+l4)/6
    e1=abs(T3-final_T3)
    if e1<1e-3 then disp(t,"the approx. time when
       Temperature in 3rd tank is 99% of steady
       value is"); break
    end
    T3=T3+(m1+2*m2+2*m3+m4)/6
end
```

Scilab code Exa 4.11 batch and stirred tank reactor

```
clc
//batch reactors
disp("the solution of e.g. 4.11 -->Batch and Stirred Tank Reactors")
//rxn given A--> B
rate_const_k=1
function dCa_by_dt=fs1(t,Ca),
    dCa_by_dt=-rate_const_k*Ca,
endfunction
Ca=1
for t=0.1:0.1:3,
    h=0.1                                    //step increment of 0.1
    k1=h*fs1(t,Ca)
    k2=h*fs1(t+h/2,Ca+k1/2)
    k3=h*fs1(t+h/2,Ca+k2/2)
    k4=h*fs1(t+h,Ca+k3)
    Ca=Ca+(k1+2*k2+2*k3+k4)/6
end
disp(Ca,"the value of conc. at t=3 using Runge Kutta method is");
Ca_anl=exp(-t)                    //analytical solution
disp(Ca_anl,"the analytical soln. is")
```

Scilab code Exa 4.12 batch and stirred tank reactor

```
clc
//rxn A-->B
//input=FCa0, output=FCa
//applying mass balance of component A we get d(V*Ca)/dt=F*Ca0-F*Ca-k*Ca*V
disp("the solution of e.g. 4.12 -->Batch and Stirred Tank Reactors")
rate_const_k=1
```

```
Ca0=1,F=1, V=10

function  dVCa_by_dt=fr(t,Ca1),
    dVCa_by_dt=F*Ca0-F*Ca1-rate_const_k*Ca1*V,
endfunction
Ca1=1
for t=0.1:0.1:10,
    h=0.1                                    //step increment of 0.1
    k1=h*fr(t,Ca1)
    k2=h*fr(t+h/2,Ca1+k1/2)
    k3=h*fr(t+h/2,Ca1+k2/2)
    k4=h*fr(t+h,Ca1+k3)
    Ca1=Ca1+(k1+2*k2+2*k3+k4)/6
end                        //final value
disp(Ca1,"the value of Ca at t=10 s using Runge Kutta method is");
Ca_steady=F*Ca0/(F+rate_const_k*V)
disp(Ca_steady,"the steady state value of conc. is");
```

Scilab code Exa 4.13 batch and stirred tank reactor

```
clc
//given rxn A-->B-->C
k1=1, k2=1                        //given data
disp("the solution of eg 4.13 -->Batch Reactors")
function dA_by_dt=f1a(t,A,B,C),                    //functions defined
    dA_by_dt=-A,
endfunction
function dB_by_dt=f2a(t,A,B,C),
    dB_by_dt=A-B,
endfunction
function dC_by_dt=f3a(t,A,B,C),
```

```
    dC_by_dt=B,
endfunction
A=1,B=0,C=0                         //initial values
for t=0.1:.1:10,
    h=.1                                    //step increment of 0.1
    k1=h*f1a(t,A,B,C)
    l1=h*f2a(t,A,B,C)
    m1=h*f3a(t,A,B,C)
    k2=h*f1a(t+h/2,A+k1/2,B+l1/2,C+m1/2)
    l2=h*f2a(t+h/2,A+k1/2,B+l1/2,C+m1/2)
    m2=h*f3a(t+h/2,A+k1/2,B+l1/2,C+m1/2)
    k3=h*f1a(t+h/2,A+k2/2,B+l2/2,C+m2/2)
    l3=h*f2a(t+h/2,A+k2/2,B+l2/2,C+m2/2)
    m3=h*f3a(t+h/2,A+k2/2,B+l2/2,C+m2/2)
    k4=h*f1a(t+h,A+k3,B+l3,C+m3)
    l4=h*f2a(t+h,A+k3,B+l3,C+m3)
    m4=h*f3a(t+h,A+k3,B+l3,C+m3)
    A=A+(k1+2*k2+2*k3+k4)/6
    B=B+(l1+2*l2+2*l3+l4)/6
    C=C+(m1+2*m2+2*m3+m4)/6
    if t==.5 |t==1|t==2|t==5 then disp(C,B,A,"secs is",t,"the conc. of A,B,C after");
    end
end
disp(C,B,A,"the conc. of A,B,C after 10 secs respectively is");
```

Scilab code Exa 4.14 batch and stirred tank reactor

```
clc
//given rxn A+B—k1—>C
//             B+C—k2—>D
k1=1, k2=1                          //given rate constants
```

```
disp("the solution of eg 4.14 -->Batch Reactors")
function dA_by_dt=f1a(t,A,B,C,D),
    dA_by_dt=-A*B,
endfunction
function dB_by_dt=f2a(t,A,B,C,D),
    dB_by_dt=-A*B-B*C,
endfunction
function dC_by_dt=f3a(t,A,B,C,D),
    dC_by_dt=A*B-B*C,
endfunction
function dD_by_dt=f4a(t,A,B,C,D),
    dD_by_dt=B*C,
endfunction
A=1,B=1,C=0,D=0                                  //initial values
for t=.1:.1:10,
    h=.1                                         //step increment of 0.1
    k1=h*f1a(t,A,B,C,D)
    l1=h*f2a(t,A,B,C,D)
    m1=h*f3a(t,A,B,C,D)
    n1=h*f4a(t,A,B,C,D)
    k2=h*f1a(t+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
    l2=h*f2a(t+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
    m2=h*f3a(t+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
    n2=h*f4a(t+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
    k3=h*f1a(t+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
    l3=h*f2a(t+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
    m3=h*f3a(t+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
    n3=h*f4a(t+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
    k4=h*f1a(t+h,A+k3,B+l3,C+m3,D+n3)
    l4=h*f2a(t+h,A+k3,B+l3,C+m3,D+n3)
    m4=h*f3a(t+h,A+k3,B+l3,C+m3,D+n3)
    n4=h*f4a(t+h,A+k3,B+l3,C+m3,D+n3)
    A=A+(k1+2*k2+2*k3+k4)/6
    B=B+(l1+2*l2+2*l3+l4)/6
    C=C+(m1+2*m2+2*m3+m4)/6
    D=D+(n1+2*n2+2*n3+n4)/6
```

```
    if t==.5 |t==1|t==2|t==5 then disp(D,C,B,A,"secs
        is",t,"the conc. of A,B,C,D after");
    end
end
    disp(D,C,B,A,"the conc. of A,B,C,D after 10 secs
        respectively is");
```

Scilab code Exa 4.15 plug flow reactor

```
clc
disp("the solution of eg 4.15 -->Plug Flow Reactor")
rc_k1=1                          //given rate constant
u=1                    //mean axial velocity
function dCa_by_dx=fm(x,Ca),
    dCa_by_dx=-Ca,
endfunction
Ca=1
for x=.1:.1:10,
    h=0.1                                       //step
       increment of 0.1
    k1=h*fm(x,Ca)
    k2=h*fm(x+h/2,Ca+k1/2)
    k3=h*fm(x+h/2,Ca+k2/2)
    k4=h*fm(x+h,Ca+k3)
    Ca=Ca+(k1+2*k2+2*k3+k4)/6
    if x==.5|x==1|x==2|x==5 then disp(Ca,"length is"
       ,x,"the value of conc. at");
    end
end
disp(Ca,"the value of Ca at x=10 using Runge Kutta
   method in plug flow reactor is");
```

Scilab code Exa 4.16 plug flow reactor

```
clc
//given rxn A—>B—>C
rc_k1=1, rc_k2=1                                    //given rate constants
u=1                                          //mean axial velocity
disp("the solution of eg 4.16 —>Plug Flow Reactor")
function dA_by_dx=f1e(x,A,B,C),
    dA_by_dx=-A,
endfunction
function dB_by_dx=f2e(x,A,B,C),
    dB_by_dx=A-B,
endfunction
function dC_by_dx=f3e(x,A,B,C),
    dC_by_dx=B,
endfunction
A=1,B=0,C=0
for x=.1:.1:10,
    h=.1                                         //step increment of 0.1
    k1=h*f1e(x,A,B,C)
    l1=h*f2e(x,A,B,C)
    m1=h*f3e(x,A,B,C)
    k2=h*f1e(x+h/2,A+k1/2,B+l1/2,C+m1/2)
    l2=h*f2e(x+h/2,A+k1/2,B+l1/2,C+m1/2)
    m2=h*f3e(x+h/2,A+k1/2,B+l1/2,C+m1/2)
    k3=h*f1e(x+h/2,A+k2/2,B+l2/2,C+m2/2)
    l3=h*f2e(x+h/2,A+k2/2,B+l2/2,C+m2/2)
    m3=h*f3e(x+h/2,A+k2/2,B+l2/2,C+m2/2)
    k4=h*f1e(x+h,A+k3,B+l3,C+m3)
    l4=h*f2e(x+h,A+k3,B+l3,C+m3)
    m4=h*f3e(x+h,A+k3,B+l3,C+m3)
    A=A+(k1+2*k2+2*k3+k4)/6
    B=B+(l1+2*l2+2*l3+l4)/6
    C=C+(m1+2*m2+2*m3+m4)/6
    if x==.5 |x==1|x==2|x==5 then disp(C,B,A,"mtr is ",x,"the conc. of A,B,C at a distance of");
    end
```

```
end
    disp(C,B,A,"the conc. of A,B,C at a distance of
       10 mtr is");
```

Scilab code Exa 4.17 plug flow reactor

```
clc
//given rxn A+B—k1—>C
//            B+C—k2—>D
rc_k1=1,rc_k2=1                                //rate
   constants
disp("the solution of eg 4.17 —>Plug Flow Reactor")
function dA_by_dx=f1a(x,A,B,C,D),
    dA_by_dx=-A*B,
endfunction
function dB_by_dx=f2a(x,A,B,C,D),
    dB_by_dx=-A*B-B*C,
endfunction
function dC_by_dx=f3a(x,A,B,C,D),
    dC_by_dx=A*B-B*C,
endfunction
function dD_by_dx=f4a(x,A,B,C,D),
    dD_by_dx=B*C,
endfunction
A=1,B=1,C=0,D=0
for x=.1:.1:10,
    h=.1                                       //step
       increment of 0.1
    k1=h*f1a(x,A,B,C,D)
    l1=h*f2a(x,A,B,C,D)
    m1=h*f3a(x,A,B,C,D)
    n1=h*f4a(x,A,B,C,D)
    k2=h*f1a(x+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
    l2=h*f2a(x+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
    m2=h*f3a(x+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
```

```
    n2=h*f4a(x+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
    k3=h*f1a(x+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
    l3=h*f2a(x+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
    m3=h*f3a(x+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
    n3=h*f4a(x+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
    k4=h*f1a(x+h,A+k3,B+l3,C+m3,D+n3)
    l4=h*f2a(x+h,A+k3,B+l3,C+m3,D+n3)
    m4=h*f3a(x+h,A+k3,B+l3,C+m3,D+n3)
    n4=h*f4a(x+h,A+k3,B+l3,C+m3,D+n3)
    A=A+(k1+2*k2+2*k3+k4)/6
    B=B+(l1+2*l2+2*l3+l4)/6
    C=C+(m1+2*m2+2*m3+m4)/6
    D=D+(n1+2*n2+2*n3+n4)/6
    if x==.5 |x==1|x==2|x==5 then disp(D,C,B,A,"secs is",x,"the conc. of A,B,C,D after");
    end
end
    disp(D,C,B,A,"the conc. of A,B,C,D after 10 secs respectively is");
```

Scilab code Exa 4.18 non isothermal plug flow reactor

```
clc
disp("the solution of eg 4.18-->Non- Isothermal Plug Flow Reactor")
T=294.15
//rxn A-->B
R=8.314, rho=980.9, MW=200, U=1900, Cp=15.7, H_rxn=92900, T1=388.71, mdot=1.26, dia=2.54*10^-2, E=108847          //given data
b=E/(R*T1),k1=5.64*10^13*exp(-b),  A=%pi*dia^2/4, na0=mdot*1000/MW, Ts=388.71
k=k1*exp(b*(1-T1/T))

//dX_by_dV=ra/na0
```

```
//dX_by_dV=k*(1-X)/F
//from energX balance
//mdot*Cp*dT_by_dz+ra*A*H_RXN-q=0
//q=U*%pi*dia*(Ts-T)
//-mdot*Cp*dT_by_dV+4*U/dia*(Ts-T)-ra*H_rxn=0
F=mdot/rho
t1=A*k1/F

s1=mdot*Cp/A
s2=4*U/dia
s3=H_rxn*t1

function dX_by_dz=fg1(z,X,T),
         dX_by_dz=t1*(1-X)*exp(b*(1-T1/T))
endfunction
function dT_by_dz=fd1(z,X,T),
         ra=na0/A*(t1*(1-X)*exp(b*(1-T1/T)))
         dT_by_dz=(ra*H_rxn-s2*(Ts-T))/-s1

endfunction

X=0,T=294.15
for z=0:.1:350,
    h=.1                                                    //szep incremenz of 0.1
    k1=h*fg1(z,X,T)
    l1=h*fd1(z,X,T)
    k2=h*fg1(z+h/2,X+k1/2,T+l1/2)
    l2=h*fd1(z+h/2,X+k1/2,T+l1/2)
    k3=h*fg1(z+h/2,X+k2/2,T+l2/2)
    l3=h*fd1(z+h/2,X+k2/2,T+l2/2)
    k4=h*fg1(z+h,X+k3,T+l3)
    l4=h*fd1(z+h,X+k3,T+l3)
    X=X+(k1+2*k2+2*k3+k4)/6
    T=T+(l1+2*l2+2*l3+l4)/6
    //condition for height calc. for 90% conversion
    if X>.9 &X<.9005 then disp(z,"the height of the tower required for 90% conversion in mtrs is"
```

```
        ); break
    end
    end
```

Chapter 5

BOUNDARY VALUE PROBLEMS

Scilab code Exa 5.1 discretization in 1 D space

```
clc
//finite difference method
disp("the solution of eg 5.1-->Discretization in 1-D space");
//given d2y_by_dx2-2=0 hence it is dirichlet's problem

x_1=0, y_1=0                //initial boundary conditions
x_3=1, y_3=0

delta_x=.5                 //since we have to find y_2 at x=.5 so x_2=.5
//in the central difference method substituting i=2 we have
//(y_3-2*y_2+y_1)/(delta_x)^2=2
//since y_1 & y_3=0 as per B.C.
y_2=(y_3+y_1-2*delta_x^2)/2
disp(y_2,"the value of y at x=.5 from finite
```

```
difference method is");
x=.5
y_exact=x^2-x
disp(y_exact,"the value of y from exact soln at x=.5 is");
```

Scilab code Exa 5.2 discretization in 1 D space

```
clc
disp("the solution of eg 5.2 -->Discretization in 1-D space");
//boundary conditions are: x=0 at y=0; dy/dx=1 at x=1
disp("to solve this problem we will take delta x=.5 since we have to find the value at x=.5");
delta_x=.5
y_1=0
//using central difference eqn
dy_by_dx=1                    //at x=1, i=3
//y_4=dy/dx*2*delta_x+y_2                sincefrom B.C. at node 3

//y_2=delta_x^2+y_3-delta_x              on substituting the value of y_4
y_3=-(2*delta_x^2+2*(delta_x^2-delta_x)-y_1) //on substituting the value of y_2
y_2=delta_x^2+y_3-delta_x
disp(y_2,"the value of y at x=.5 is");
```

Scilab code Exa 5.3 discretization in 1 D space

```
clc
```

```
disp("the solution of eg 5.3 —>Discretization in 1—
   D space");
//given the source term f(x)=4x^2−2x−4
//given eqn d2y/dx2−2y=f(x)
y_1=0
y_4=-1
delta_x=1/3              //since given 3 parts and
   length=1
for i=0:3,j=0:delta_x:1;
end
//given to divide the line in 3 parts
//at node 2
//y_3 −2*y_2
function d=fx3(x),
d=(4*x^2-2*x-4)
endfunction
f2=fx3(j(2))
f3=fx3(j(3))
y_3=((f2)*delta_x^2+(2+2*delta_x^2)*((f3)*delta_x^2-
   y_4)-y_1)/(1-(2+2*delta_x^2)^2)
y_2=(f3+2*y_3)*delta_x^2+2*y_3-y_4
disp(y_3,y_2,"is respectively",j(3),j(2),"the value
   of y at x=");
```

Scilab code Exa 5.4 discretization in 1 D space

```
//f(x)=4x2−2x−4
clc
disp("the solution of ex 5.4 by TDMA method is");
y_1=0
dy_by_dx=-3              //at x=1
delta_x=1/3              //since given 3 parts and
   length=1
for i=0:3,j=0:delta_x:1;
end
```

```
//given to divide the line in 3 parts
function d=fx3(x),
d=(4*x^2-2*x-4)
endfunction
f2=fx3(j(2))
f3=fx3(j(3))
f4=fx3(j(4))
disp("the exact analytical soln are");
for i=1:3,y(i)=-2*j(i+1)^2+j(i+1),disp(y(i));
end
//using B.C.-2 at node 4 we get
//(y_5-y_3)/(2*delta_x)=-3
//eqn at node 2
//-20*y_2+9*y_3=f2
//at node 3
//9*y_2 -20*y_3+9*y_4=f3
//at node 4
//18*y_3 -20*y_4=16
disp(f4,f3,f2,"the value of f(x) at node 2 3 and 4 are");
//now solving the equations using TDMA method

a(2)=9,a(3)=18                //sub diagonal assignment

for j=1:3, b(j)=-20;             //main diagonal assignment
end
for k=1:2, c(k)=9;              //super diagonal assignment
end
d(1)=f2                         //given values assignment
d(2)=f3
d(3)=16

i=1;
n=3;
beta1(i)=b(i);                  //initial b is equal
```

```
to beta since a1=0
gamma1(i)=d(i)/beta1(i);        //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                 //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end
disp("the values of y2, y3, y4 by TDMA method are");
for i=1:3, disp(x(i));
end
```

Scilab code Exa 5.5 discretization in 1 D space

```
clc
disp("the soln of eqn 5.5-->");
delta_x=.1
y_11=1
dy_by_dx_1=0            //dy/dx at x=0
// given eqn d2y/dx2=y
disp("the analytical soln are");
for i=1:10, y_an(i)=cosh((i-1)/10)/cosh(1), disp(y_an(i));
end
//using central difference method we have
//y(i-1) - (2+delat_x^2)y(i) + y(i+1)=0
//therefore the eqns can be solved using TDMA method
for i=2:10, a(i)=1              //sub diagonal assignment
end
for j=1:10, b(j)=-2.01;             //main diagonal assignment
```

```
end
c(1)=2
for k=2:9, c(k)=1;                //super diagonal assignment
end
for l=1:9, d(l)=0;
    end                          //given values assignment
d(10)=-1
i=1;
n=10;
beta1(i)=b(i);                    //initial b is equal to beta since a1=0
gamma1(i)=d(i)/beta1(i);          //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                   //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end
disp("the values of y from y1 to y10 by TDMA method are");
for i=1:10, disp(x(i));
end
```

Scilab code Exa 5.6 1 D steady state heat conduction

```
clc
disp("the soln of eqn 5.6-->1-D Steady Heat Conduction");
//in the given problem
T_1=100, T_10=200
```

```
delta_x=(T_10-T_1)/9                    //since 9 divisions are to be made
//using central difference method we get
//for node 2--> 2*T_2-T_3=100
for i=2:8, a(i)=-1                 //sub diagonal assignment
end
for j=1:8, b(j)=2;                //main diagonal assignment
end
for k=1:7, c(k)=-1;                //super diagonal assignment
end
d(1)=100, d(8)=200
for l=2:7, d(l)=0;
end                          //given values assignment
i=1;
n=8;
beta1(i)=b(i);                    //initial b is equal to beta since a1=0
gamma1(i)=d(i)/beta1(i);          //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                   //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end
disp("the values of T from T2 to T9 by TDMA method are");
for i=1:8, disp(x(i));
end
```

Scilab code Exa 5.7 1 D steady state heat conduction

```
clc
disp("the soln of eqn 5.7-->1-D Steady Heat Conduction");
dia=.02
l=.05
T_0=320
delta_x=l/4
k=50
h=100
T_surr=20
//B.C--> d(theta)/dx+h(theta)/k=0 at x=0.05
//let m=sqrt(hP/kA)
P=%pi*dia
A=%pi*dia^2/4
m=sqrt(h*P/(k*A));
//using central difference method we get
//-theta(i-1)+(2+(m*delta_x^2)*theta(i)+theta(i+1))=0
theta_0=T_0-T_surr
//using B.C. at node 4 we get--> theta(5)=theta(3)-2h*theta(4)*delta_x/k
//now the eqns can be solved using TDMA method
for i=2:3, a(i)=-1                //sub diagonal assignment
end
a(4)=-2
for j=1:3, b(j)=2.0625;           //main diagonal assignment
end
b(4)=2.1125
for k=1:3, c(k)=-1;               //super diagonal assignment
end
for l=2:4, d(l)=0;
    end                           //given values assignment
```

```
d(1)=300
i=1;
n=4;
beta1(i)=b(i);                    //initial b is equal
    to beta since a1=0
gamma1(i)=d(i)/beta1(i);          //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                   //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
    j);
end
disp("the values of T from T1 to T4 in Celsius by
    TDMA method are");
for i=1:4, disp(x(i)-T_surr);
end
```

Scilab code Exa 5.8 chemical reaction and diffusion in pore

```
clc
disp("the soln of eg 5.8-->Chemical Reaction and
    Diffusion in Pore");
lnght=.001
k_const=.001
D=10^-9
delta_x=l/100
C=1                    //in mol/m3
//B.C. are C=1 at x=0
//          dC/dx=0 at x=10^-3     since at the end
    point conc. is const.
//using central difference method we get the
    following eqns which can be solved using TDMA
```

```
method
for i=2:99, a(i)=1                    //sub diagonal assignment
end
a(100)=2                         //since C99=C100 using B.C.
for j=1:100, b(j)=-2.0001,               //main diagonal assignment
end
for k=1:99, c(k)=1;                  //super diagonal assignment
end
d(1)=-1
for l=2:100, d(l)=0;
end                            //given values assignment
i=1;
n=100;
beta1(i)=b(i);                       //initial b is equal to beta since a1=0
gamma1(i)=d(i)/beta1(i);             //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                      //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end
disp(x(50),"the values of conc. at x=.5mm or at the 50th node is");
```

Chapter 6

CONVECTION DIFFUSION PROBLEMS

Scilab code Exa 6.1 upwind scheme

```
//given convective diffusive eqn—> −u*(dc/dx)+D*(
    d2C/dx2)=0
C_ini=0                 //at x=0
C_end=1                 //at x=1
clc
disp("the soln of eg 6.1");

//using central difference method for both diffusion
     and convective term
//−u*(C(i+1)−C(i−1))/(2*delta_x) + D*(C(i+1)+C(i−1)
    −2*C(i))/delta_x^2 = 0
delta_x=1/50
//on solving the given eqns and by using the given
    boundary eqns we have
Pe=50                                //given
Pe_local=50*delta_x                  //u/D=50 as l=1
alpha=Pe_local-2                     //co−eff of C(i
    +1)
Beta=Pe_local+2                      //co−eff of C(i
```

```
-1)
//multipling with -2*delta_x^2/D we get
//-(Pe_local+2)*C(i-1) + 4*C(i) + (Pe_local-2)*C(i+1)=0
//solving eqns using TDMA method
for i=2:49, a(i)=-Beta              //sub diagonal assignment
end
for j=1:49, b(j)=4,             //main diagonal assignment
end
for k=1:48, c(k)=alpha;             //super diagonal assignment
end
d(1)=Beta*C_ini, d(49)=-alpha*C_end
for l=2:48, d(l)=0;
end                         //given values assignment
i=1;
n=49;
beta1(i)=b(i);                  //initial b is equal to beta since a1=0
gamma1(i)=d(i)/beta1(i);        //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                 //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end
disp("the values of conc. using CDS method for x=.84 to 1 are")
for i=42:49,disp(x(i));
end
//part (ii) using CDS and UDS method
//multipling with -delta_x^2/D we get
//-(Pe_local+1)*C(i-1) + (Pe_local+2)*C(i)-C(i+1)=0
```

```
BEta=Pe_local+2
Gamma=Pe_local+1
for i=2:49, a(i)=-Gamma              //sub diagonal assignment
end
for j=1:49, b(j)=BEta,             //main diagonal assignment
end
for k=1:48, c(k)=-1;             //super diagonal assignment
end
d(1)=Gamma*C_ini, d(49)=C_end
for l=2:48, d(l)=0;
end                        //given values assignment
i=1;
n=49;
beta1(i)=b(i);                  //initial b is equal to beta since a1=0
gamma1(i)=d(i)/beta1(i);        //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                 //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end
disp("the values of conc. using CDS/UDS method for x=.84 to 1 are")
for i=42:49,disp(x(i));
end
disp("the analytical soln is");
for i=42:49;
     C_an(i)=C_ini+((exp(Pe*.02*i)-1)*(C_end-C_ini)/(exp(Pe)-1));
    disp(C_an(i));
    end
```

Scilab code Exa 6.2 upwind scheme

```
//given convective diffusive eqn—> −u∗(dc/dx)+D∗(
    d2C/dx2)=0
C_ini=0             //at x=0
C_end=1             //at x=1
clc
disp("the soln of eg 6.2−−>");

//using central difference method for both diffusion
     and convective term
//−u∗(C(i+1)−C(i−1))/(2∗delta_x) + D∗(C(i+1)+C(i−1)
    −2∗C(i))/delta_x^2 = 0
delta_x=1/50
//on solving the given eqns and by using the given
    boundary eqns we have
Pe=500                              //given
Pe_local=500*delta_x                //u/D=50 as l=1
alpha=Pe_local-2                   //co−eff of C(i
    +1)
Beta=Pe_local+2                    //co−eff of C(i
    −1)
//multipling with −2∗delta_x^2/D we get
//−(Pe_local+2)∗C(i−1) + 4∗C(i) + (Pe_local−2)∗C(i
    +1)=0
//solving eqns using TDMA method
for i=2:49, a(i)=-Beta              //sub diagonal
    assignment
end
for j=1:49, b(j)=4,            //main diagonal
    assignment
end
for k=1:48, c(k)=alpha;             //super diagonal
    assignment
```

```
end
d(1)=Beta*C_ini, d(49)=-alpha*C_end
for l=2:48, d(l)=0;
end                      //given values assignment
i=1;
n=49;
beta1(i)=b(i);                 //initial b is equal to beta since a1=0
gamma1(i)=d(i)/beta1(i);       //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end
disp("the values of conc. using CDS method for x=.84 to 1 are")
for i=42:49,disp(x(i));
end
//part (ii) using CDS and UDS method
//multipling with -delta_x^2/D we get
//-(Pe_local+1)*C(i-1) + (Pe_local+2)*C(i)-C(i+1)=0
BEta=Pe_local+2
Gamma=Pe_local+1
for i=2:49, a(i)=-Gamma             //sub diagonal assignment
end
for j=1:49, b(j)=BEta,            //main diagonal assignment
end
for k=1:48, c(k)=-1;             //super diagonal assignment
end
d(1)=Gamma*C_ini, d(49)=C_end
for l=2:48, d(l)=0;
```

```
end                         //given values assignment
i=1;
n=49;
beta1(i)=b(i);                    //initial b is equal to beta since a1=0
gamma1(i)=d(i)/beta1(i);          //since c7=0
m=i+1;
for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
end
x(n)=gamma1(n);                   //since c7=0
n1=n-i;
for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(j);
end
disp("the values of conc. using CDS/UDS method for x=.84 to 1 are")
for i=42:49,disp(x(i));
end
disp("the analytical soln is");
for i=42:49;
     C_an(i)=C_ini+((exp(Pe*.02*i)-1)*(C_end-C_ini)/(exp(Pe)-1));
    disp(C_an(i));
    end
```

Chapter 7

TUBULAR REACTOR WITH AXIAL DISPERSION

Scilab code Exa 7.1 boundary value problem in chemical reaction engineering

```
clc
disp("soln of eg 7.1-->First Order Reaction-50 parts")
e1=1,e2=1                                    //assumed values
u=1,D=10^-4,k=1,C_a_in=1,delta_x=10/50                    //given data
cf_ca1_n1=-2*D/delta_x^2-3*u/delta_x-k-2*u^2/D               //co-efficient of C_A1 at node 1
cf_ca2_n1=2*D/delta_x^2+u/delta_x
cf_da1_n1=-(2*u^2/D+2*u/delta_x)*C_a_in                    //right hand side co-efficient
cf_ca1_n2=D/delta_x^2+u/delta_x
cf_ca2_n2=-2*D/delta_x^2-u/delta_x-k
cf_ca3_n2=D/delta_x^2
cf_da1_n2=0
```

```
cf_ca2_n3=cf_ca1_n2
cf_ca3_n3=cf_ca2_n2
cf_ca4_n3=cf_ca3_n2
cf_da1_n3=0
cf_ca50_n51=2*D/delta_x^2+u/delta_x //co-efficient of C-A50 at node 51
cf_ca51_n51=-2*D/delta_x^2-u/delta_x-k
cf_da51_n51=0
for i=2:50, a(i)=cf_ca1_n2,
end
a(51)=cf_ca2_n1,c(1)=cf_ca2_n1
for i=2:50,c(i)=cf_ca3_n2,
end
d(1)=cf_da1_n1
for i=2:51,d(i)=cf_da1_n2
end
for i=1:51,x(i)=0,
end
b(1)=cf_ca1_n1,
for i=2:51,b(i)=cf_ca2_n2,end
while e1>1e-6 & e2>1e-6 do for i=1:51,x1(i)=x(i),end,
i=1, n=51, Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
e1=abs(x(42)-x1(42)), e2=abs(x(18)-x1(18))
end
for i=1:51,disp(x(i))
end
```

Scilab code Exa 7.2 boundary value problem in chemical reaction engineering second order

```
clc
disp("soln of eg 7.2-->Second Order Reaction -20
   parts")
e1=1,e2=1
u=1,D=10^-4,k=1,C_a_in=1,delta_x=10/20
cff_ca2_n1=2*D/delta_x^2+u/delta_x
                                    //co-efficient of C-A2 at
   node 1
cff_da1_n1=-(2*u^2/D+2*u/delta_x)*C_a_in
                          //right hand side co-efficient
cff_ca1_n2=D/delta_x^2+u/delta_x
cff_ca3_n2=D/delta_x^2
                                                //co-efficient
    of C-A3 at node 2
cff_da1_n2=0
cff_ca2_n3=cf_ca1_n2
cff_ca4_n3=cf_ca3_n2
cff_da1_n3=0
cff_ca20_n21=2*D/delta_x^2+u/delta_x
                                    //co-efficient of C-A20 at
    node 21
cff_da21_n21=0
for i=2:20, a(i)=cff_ca1_n2,
end
a(21)=cff_ca2_n1,c(1)=cff_ca2_n1
for i=2:20,c(i)=cff_ca3_n2,
end
d(1)=cff_da1_n1
for i=2:21,d(i)=cff_da1_n2
end
for i=1:21,x(i)=0,
```

```
end
while e1>1e-6 & e2>1e-6 do for i=1:21,x1(i)=x(i),end,
   cff_ca1_n1=-2*D/delta_x^2-3*u/delta_x-x1(1)-2*u^2/D                 //main diagonal elements dependence on conc.
   b(1)=cff_ca1_n1,
   for i=2:21,b(i)=-2*D/delta_x^2-u/delta_x-x(i),end

//solving by TDMA method
   i=1, n=21, Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
e1=abs(x(1)-x1(1)), e2=abs(x(21)-x1(21))
end
for i=1:21,disp(x(i))
end
```

Chapter 8

CHEMICAL REACTION AND DIFFUSION IN SPHERICAL CATALYST PELLET

Scilab code Exa 8.1 first order reaction

```
clc
disp("the soln of eg 8.1-->");
for i=1:100, x(i)=0
end
iter=0, e1=1, f=1
while e1>1e-6 & f>1e-6 do iter=iter+1,for i=1:100,
   x1(i)=x(i),
    end, for i=2:100, a(i)=1-(1/(i-1))
    end, b(1)=-6.01, for i=2:100, b(i)=-2.01
    end, c(1)=6, for i=2:99,c(i)=1+(1/(i-1))
    end, for i=1:99,d(i)=0, end, d(100)=-100/99,
    i=1, n=100, Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
```

```
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
       (j)
    end
    e1=abs(x(1)-x1(1)),
    f=abs(x(100)-x1(100)),
end
disp(iter,"the solution by TDMA of node 77 to 99 by
   1st order rxn. is");
for i=78:100,
    disp(x(i));
end
```

Scilab code Exa 8.2 second order reaction

```
clc
disp("the soln of eg 8.2-->");
for i=1:100, x(i)=0
end
iter=0, e1=1, f=1
k=.1,D=10^-9,r=.01,delta_r=r/10,t1=k*delta_r^2/D
while e1>1e-6 & f>1e-6 do iter=iter+1,for i=1:100,
   x1(i)=x(i),
    end, for i=2:100, a(i)=1-(1/(i-1))
    end, b(1)=-6-.t1*x1(1), for i=2:100, b(i)=-2-t1*
       x1(i)
    end, c(1)=6, for i=2:99,c(i)=1+(1/(i-1))
    end, for i=1:99,d(i)=0, end, d(100)=-100/99,
    i=1, n=100, Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
```

```
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
    e1=abs(x(1)-x1(1)),
    f=abs(x(100)-x1(100)),
end
disp("the solution by TDMA of node 77 to 99 by 2nd order rxn. is");
for i=77:100,
    disp(x(i));
end
```

Scilab code Exa 8.3 non isothermal condition

```
clc
disp("the soln of eg 8.3-->");
for i=1:100, x(i)=0                        //initial values
end
e2=1, f1=1,iter=0                          //assumed values
k=.1*10^-2, D=10^-9,r=.01,delta_r=r/100,t1=k*delta_r^2/D        //given data
//now solving the eqns for all the nodes and then simplifying we get the following relations
while e2>1e-6 & f1>1e-6 do iter=iter+1,for i=1:100, x1(i)=x(i),
    end, for i=2:100, a(i)=1-(1/(i-1))
    end, b(1)=-6-t1*exp((1-x1(1))/(2-x1(1))), for i=2:100, b(i)=-2-t1*exp((1-x(i))/(2-x(i)))
    end, c(1)=6, for i=2:99,c(i)=1+(1/(i-1))
    end, for i=1:99,d(i)=0, end, d(100)=-100/99,
    i=1, n=100, Beta(i)=b(i),
```

```
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
    e2=abs(x(1)-x1(1)),
    f1=abs(x(100)-x1(100)),
end
disp("the solution by TDMA of node 77 to 100 by 1st order rxn. is");
for i=76:100,
    disp(x(i));
end
```

Scilab code Exa 8.4 non isothermal condition

```
clc
disp("the soln of eg 8.4-->");
for i=1:100, x(i)=0
end
iter=0, e1=1, f1=1
while e1>1e-6 & f1>1e-6 do iter=iter+1,for i=1:100, x1(i)=x(i),
    end, for i=2:100, a(i)=1-(1/(i-1))
    end, b(1)=-6-.01*exp((10-10*x1(1))/(11-10*x1(1))), for i=2:100, b(i)=-2-.01*exp((10-10*x1(i))/(11-10*x1(i)))
    end, c(1)=6, for i=2:99,c(i)=1+(1/(i-1))
    end, for i=1:99,d(i)=0, end, d(100)=-100/99,
    i=1, n=100, Beta(i)=b(i),
```

```
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
    e1=abs(x(1)-x1(1)),
    f1=abs(x(100)-x1(100)),
end
disp("the solution by TDMA of node 77 to 99 by 1st order rxn. is");
for i=76:100,
    disp(x(i));
end
```

Chapter 9

ONE DIMENSIONAL TRANSIENT HEAT CONDUCTION

Scilab code Exa 9.1 transient conduction in rectangular slab

```
clc
disp("the soln of eg 9.1-->Transient heat conduction
    in a Rectangular slab")
disp("the values of Temp measured from centre at the
    gap of .2 cm are")
alpha=10^-5, delta_t=.1, delta_x=10^-3             //
   given data
m=alpha*delta_t/(delta_x^2)
for i=2:4, a(i)=m                                  //
   sub-diagonal
end
b(1)=(-2*m-1)/2
for i=2:4,b(i)=-2*m-1
   //diagonal
end
for i=1:3, c(i)=m                                  //
   super-diagonal
```

```
end
for i=1:4, x(i)=20                              //initial temperature
end
for t=0.1:.1:3.1,for i=1:4, y(i)=x(i), end              //TDMA method
    d(1)=-.5*y(1),
    d(2)=-y(2)
    d(3)=-y(3)
    d(4)=-y(4)-300
    i=1, n=4
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
for i=1:4,disp(x(i));                                  //solution of temperature
end
disp("--------------------------------")
end
disp("------------------END------------------");
```

Scilab code Exa 9.2 transient conduction in rectangular slab

```
clc
disp("the soln of eg 9.2-->Transient Conduction in Rectangular Slab");
```

```
delta_t=1, delta_x=.05, alpha=10^-5
t1=alpha*delta_t/delta_x^2
for i=2:9, a(i)=-t1
end
for i=1:9,b(i)=1+2*t1
end
for i=1:8, c(i)=-t1
end
t=1,tf=3000
for i=1:9, x(i)=300
end
e1=425,
disp("time when centre temp is 425 K in secs. is")
for t=1:1:tf,for i=1:9, y(i)=x(i),end
    d(1)=y(1)+1.7,
    d(9)=y(9)+2.4,
    for i=2:8,d(i)=y(i)
    end
    i=1, n=9
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
    if abs(x(5)-e1)>0 & abs(x(5)-e1)<.1 then disp(t); break;
    end
end
disp("the values of temp. at req. time is");
for i=1:9,disp(x(i)); end
```

Scilab code Exa 9.3 transient conduction in cylinder

```
clc
disp("the soln of eg 9.3-->Transient Conduction in cylinder");
delta_t=.1, delta_r=.001, alpha=10^-5 //given data
t1=alpha*delta_t/delta_r^2
a(2)=.5*t1,a(3)=.75*t1,a(4)=.833*t1 //sub-diagonal
b(1)=-4*t1-1
for i=2:4,b(i)=-2*t1-1 //main diagonal
end
c(1)=4*t1,c(2)=1.5*t1,c(3)=1.25*t1 //super-diagonal
for i=1:4, x(i)=20
end
disp("T1,T2,T2 & T4 at time interval of .1 sec is")
for t=.1:.1:2.1,for i=1:4, y(i)=x(i),end //TDMA Method
    d(4)=-y(4)-7*t1*300/6,
    for i=1:3,d(i)=-y(i)
    end
    i=1, n=4
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
```

```
    (j)
    end
disp(x(4),x(3),x(2),x(1));
disp("------------------------");
end
```

Scilab code Exa 9.4 transient conduction in sphere

```
clc
disp("the soln of eg 9.4-->Transient conduction in
   Sphere");
delta_t=.1, delta_r=.001, alpha=10^-5
t1=alpha*delta_t/delta_r^2
a(2)=0,a(3)=.5,a(4)=.667
b(1)=-7*t1
for i=2:4,b(i)=-3
end
c(1)=6,c(2)=2,c(3)=1.5
for i=1:4, x(i)=20
end
disp("T1,T2,T2 & T4 at time interval of .1 sec is")
for t=.1:.1:1.4,for i=1:4, y(i)=x(i),end
    d(4)=-y(4)-400,
    for i=1:3,d(i)=-y(i)
    end
    i=1, n=4
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
```

```
         (j)
    end
disp(x(4),x(3),x(2),x(1));
disp("--------------------");
end
```

Scilab code Exa 9.5 transient diffusion in sphere

```
clc
disp("the soln of eg 9.5-->");
R=.326, D=3*10^-7
delta_t=1, delta_r=.0326, conc_ini=10/(1.33*%pi*R^3)
t1=D*delta_t/delta_r^2
disp(conc_ini,"the initial conc. of drug is");
for i=2:10, a(i)=-(1-1/(i-1))
end
b(1)=591.4
for i=2:10,b(i)=3544.5
end
c(1)=-1,for i=2:9, c(i)=-(1+1/(i-1))
end
for i=1:10, x(i)=conc_ini
end
disp("Conc. at centre at t=3hr, 12 hr, 24 hr,48 hr is");
for t=1:delta_t:172800,for i=1:10, y(i)=x(i),end
    d(1)=y(1)*590.4,
    for i=2:10,d(i)=3542.5*y(i)
    end
    i=1, n=10
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
```

```
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
    if t==10800| t==43200| t==86400|t==172800 then disp(x(6));
    end
end
```

Chapter 10

TWO DIMENSIONAL STEADY AND TRANSIENT HEAT CONDUCTION

Scilab code Exa 10.1 discretization in 2 D space gauss seidel method

```
clc
disp("the soln of eg 10.1-->2-D steady heat
   conduction-Gauss Seidel method ");
for i=1:9,tnew(i)=101,e(i)=1
   //assumed values
end
t=1e-6
//since all the nodes are interior nodes so
   discretized eqn used is eqn 10.10
while e(1)>t&e(2)>t&e(3)>t&e(4)>t&e(5)>t &e(6)>t& e
   (7)>t& e(8)>t & e(9)>t do
    for i=1:9, told(i)=tnew(i),end
    tnew(1)=(told(2)+40+told(4))/4
                                   //on solving eqns for
       various nodes we get,
    tnew(2)=(tnew(1)+told(3)+told(5)+20)/4
    tnew(3)=(tnew(2)+told(6)+420)/4
```

```
    tnew(4)=(told(5)+tnew(1)+told(7)+20)/4
    tnew(5)=(tnew(2)+told(8)+told(6)+tnew(4))/4
    tnew(6)=(tnew(3)+tnew(5)+told(9)+400)/4
    tnew(7)=(tnew(4)+told(8)+40)/4
    tnew(8)=(tnew(5)+tnew(7)+told(9)+20)/4
    tnew(9)=(tnew(6)+420+tnew(8))/4
    for i=1:9,e(i)=abs(tnew(i)-told(i))
    end
end
disp("the values of T from 1st element to last is");
for i=1:9,disp(tnew(i));
end
```

Scilab code Exa 10.2 discretization in 2 D space gauss seidel method

```
clc
disp("the soln of eg 10.2-->");
for i=1:5,tnew(i)=101,e(i)=1
end
t=1e-6
//since there is no source term so we get the
   following eqns.
while e(1)>t&e(2)>t&e(3)>t&e(4)>t&e(5)>t do
    for i=1:5, told(i)=tnew(i),end
    tnew(1)=(told(2)*2+300)/4
    tnew(2)=(tnew(1)+told(3)+300)/4
    tnew(3)=(tnew(2)+told(4)+200)/4
    tnew(4)=(told(5)+tnew(3)+300)/4
    tnew(5)=(2*tnew(4)+300)/4
    for i=1:5,e(i)=abs(tnew(1)-told(i))
    end
end
disp("the values of T from 1st element to last is");
for i=1:5,disp(tnew(i));
end
```

Scilab code Exa 10.3 discretization in 2 D space

```
clc
disp("the soln of eg 10.3-->Red-Black Gauss-Seidel Method");
for j=1:5,tn(j,1)=400,end                              //defining conditions
for j=2:4,tn(1,j)=20,tn(5,j)=20,tn(j,5)=20, end
for i=1:9,e(i)=1
end
for i=2:4,j=2:4,tn(i,j)=60
end
t1=1e-6
while e(1)>t1&e(2)>t1&e(3)>t1&e(4)>t1&e(5)>t1 &e(6)>t1& e(7)>t1& e(8)>t1 & e(9)>t1 do for i=2:4,j=2:4,t(i,j)=tn(i,j),end
//using red-black gauss-seidel method
for i=2:4,j=2:4,tn(i,j)=(tn(i+1,j)+tn(i-1,j)+tn(i,j+1)+tn(i,j-1))/4, end
//now getting the absolute difference of the 9 variables
e(1)=abs(t(2,2)-tn(2,2)),e(2)=abs(t(2,3)-tn(2,3)),e(3)=abs(t(2,4)-tn(2,4)),e(4)=abs(t(3,2)-tn(3,2)),e(5)=abs(t(3,3)-tn(3,3)),e(6)=abs(t(3,4)-tn(3,4)),e(7)=abs(t(4,2)-tn(4,2)),e(8)=abs(t(4,3)-tn(4,3)),e(9)=abs(t(4,4)-tn(4,4)),
end
//now defining positions of the various nodes according to red black combination
R1=t(2,4),R2=t(4,4),R3=t(3,3),R4=t(2,2),R5=t(2,4),B1=t(4,3),B2=t(3,2),B3=t(3,4),B4=t(2,3)
disp(R5,R4,R3,R2,R1,"the value of RED points respectively is");
disp(B4,B3,B2,B1,"the value of BLUE points
```

```
respectively is");
```

Scilab code Exa 10.8 discretization in 2 D space

```
clc
disp("the soln of eg 10.8-->Gauss Seidel Method");
for i=1:9,tnew(i)=101,e(i)=1                    //
   assumed values
end
t=1e-6
while e(1)>t&e(2)>t&e(3)>t&e(4)>t&e(5)>t &e(6)>t& e
   (7)>t& e(8)>t & e(9)>t do
    for i=1:9, told(i)=tnew(i),end
    //using eqn 10.10 for the interior nodes and
       convective boundary conditions for corner
       nodes
    tnew(1)=(told(2)+50+.5*told(4)+100/3)*3/7
    tnew(2)=(tnew(1)+told(3)+told(5)+100)/4
    tnew(3)=(tnew(2)+told(6)+600)/4
    tnew(4)=(told(5)+.5*tnew(1)+.5*told(7)+100/3)
       *3/7
    tnew(5)=(tnew(2)+told(8)+told(6)+tnew(4))/4
    tnew(6)=(tnew(3)+tnew(5)+told(9)+500)/4
    tnew(7)=(.5*tnew(4)+.5*told(8)+100/3)*3/4
    tnew(8)=(tnew(5)+.5*tnew(7)+.5*told(9)+100/3)
       *3/7
    tnew(9)=(tnew(6)+100/3+.5*tnew(8)+250)*3/7
    for i=1:9,e(i)=abs(tnew(i)-told(i))
    end
end
disp("the values of T from 1st element to last is");
for i=1:9,disp(tnew(i));
end
```

Scilab code Exa 10.9 discretization in 2 D space

```
clc
disp("the soln of eg 10.9-->Steady state heat conduction in L-shaped body");
for i=1:9,tnew(i)=101,e(i)=1                //assumed values
end
t=1e-6
while e(1)>t&e(2)>t&e(3)>t&e(4)>t&e(5)>t &e(6)>t& e(7)>t& e(8)>t & e(9)>t do
    for i=1:9, told(i)=tnew(i),end
    //using the basic discretization eqn. for all the nodes since the boundary conditions vary for each point
    tnew(1)=(told(2)+1.25+told(4))/2.05
    tnew(2)=(.5*tnew(1)+.5*told(3)+told(5)+1.25)/2.05
    tnew(3)=(.5*tnew(2)+.5*told(6)+1.25)/1.05
    tnew(4)=(told(5)+.5*tnew(1)+45)/2
    tnew(5)=(tnew(2)+told(8)+90+tnew(4))/4
    tnew(6)=(.5*tnew(3)+tnew(5)+.5*told(7)+91.25)/3.05
    tnew(7)=(.5*tnew(6)+.5*told(8)+91.25)/2.05
    tnew(8)=(91.25+.5*tnew(7)+.5*told(9))/2.05
    tnew(9)=(47.125+.5*tnew(8))/1.025
    for i=1:9,e(i)=abs(tnew(i)-told(i))
    end
end
disp("the values of T from 1st element to last is");
for i=1:9,disp(told(i));
end
```

Scilab code Exa 10.10 ADI method

```
clc
disp("the soln of eg 10.10-->Alternating Direction
   Implicit Method");
nmax=25
a(2)=1, a(3)=1,                              //
   defining variables
b(1)=-4,b(2)=-4,b(3)=-4
c(1)=1,c(2)=1
t(1,2)=20,t(1,3)=20,t(1,4)=20,t(2,1)=20,t(3,1)=20,t
   (4,1)=20,t(5,2)=20,t(5,3)=20,t(5,4)=20,t(2,5)
   =400,t(3,5)=400,t(4,5)=400
tstar(1,2)=20,tstar(1,3)=20,tstar(1,4)=20,tstar(2,1)
   =20,tstar(3,1)=20,tstar(4,1)=20,tstar(5,2)=20,
   tstar(5,3)=20,tstar(5,4)=20,tstar(2,5)=400,tstar
   (3,5)=400,tstar(4,5)=400
for i=2:4, for j=2:4 t(i,j)=20
      end
end
//solving by TDMA Method
for nn=1:nmax, for jj=2:4,d(1)=-t(1,jj)-t(2,jj+1)-
   tstar(2,jj-1),
         d(2)=-t(3,jj+1)-tstar(3,jj-1),
         d(3)=-t(5,jj)-t(4,jj+1)-tstar(4,jj-1)
         i=1,n=3
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
```

```
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
    tstar(2,jj)=x(1)
    tstar(3,jj)=x(2)
    tstar(4,jj)=x(3)
end
for ii=2:4,d(1)=-t(ii,1)-tstar(ii+1,2)-tstar(ii-1,2),
    d(2)=-tstar(ii+1,3)-tstar(ii-1,3)
    d(3)=-t(ii,5)-tstar(ii+1,4)-tstar(ii-1,4)
    i=1,n=3
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end
    t(ii,2)=x(1)
t(ii,3)=x(2)
t(ii,4)=x(3)
end
end
disp("the soln by ADI method is");
disp(t(2,4),t(2,3),t(2,2));
disp("---------------");
disp(t(3,4),t(3,3),t(3,2));
disp("---------------");
disp(t(4,4),t(4,3),t(4,2));
```

Scilab code Exa 10.11 ADI method for transient heat conduction

```
clc
disp("the soln of eg 10.11-->");
for k=2:10, a(k)=-1, b(k)=2.4, c(k)=-1
end
alpha=1,delta_t=.05,delta_x=.1
m=alpha*delta_t/delta_x^2
b(1)=2.4
c(1)=-2
for k=1:11, t(11,k)=400, tstar(11,k)=400, t(k,11)=400,tstar(k,11)=400
end
for i=1:10, for j=1:10, t(i,j)=0, tstar(i,j)=0
    end
end
for tm=.05:.05:.5, for jj=1:10,if jj==1 then for ii=1:10,d(ii)=2*t(ii,2)-1.6*t(ii,1),end,d(10)=d(10)+400,else for ii=1:10, d(ii)=t(ii,jj+1)+t(ii,jj-1)-1.6*t(ii,jj),end,d(10)=d(10)+400,end,
    i=1, n=10
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta(j)
    end, for count=1:10, tstar(count,jj)=x(count),end
    end
    for ii=1:10,
        if ii==1 then for jj=1:10,d(jj)=2*tstar(ii+1,1)-1.6*tstar(ii,1),end, d(10)=d(10)+400,else for jj=1:10,d(jj)=tstar(ii-1,jj
```

```
        )+tstar(ii+1,jj)-1.6*tstar(ii,jj), end, d
        (10)=d(10)+400, end
    i=1, n=10
    Beta(i)=b(i),
    Gamma(i)=d(i)/Beta(i)
    i1=i+1,
    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
    end
    x(n)=Gamma(n)
    n1=n-i,
    for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
       (j)
    end,
    for count=1:10, t(ii,count)=x(count), end
  end
end
disp("the soln by ADI is");
for i=1:10,j=1:10,disp(t(i,j));
end
disp("the table is actually interchanged row &
   column.. horizontally are the elements of the
   column");
```