

Scilab Textbook Companion for
Introduction To Numerical Methods In
Chemical Engineering
by P. Ahuja¹

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

May 21, 2016

¹Funded 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	4
1 linear algebraic equations	5
2 NONLINEAR ALGEBRAIC EQUATIONS	12
3 CHEMICAL ENGINEERING THERMODYNAMICS	17
4 INITIAL VALUE PROBLEMS	35
5 BOUNDARY VALUE PROBLEMS	56
6 CONVECTION DIFFUSION PROBLEMS	66
7 TUBULAR REACTOR WITH AXIAL DISPERSION	72
8 CHEMICAL REACTION AND DIFFUSION IN SPHERICAL CATALYST PELLET	76
9 ONE DIMENSIONAL TRANSIENT HEAT CONDUCTION	81
10 TWO DIMENSIONAL STEADY AND TRANSIENT HEAT CONDUCTION	88

List of Scilab Codes

Exa 1.1	TDMA method	5
Exa 1.2	gauss elimination method	6
Exa 1.3	gauss elimination method	7
Exa 1.4	gauss elimination method	8
Exa 1.5	gauss seidel method	10
Exa 1.6	gauss seidel method	10
Exa 2.1	algebraic equations	12
Exa 2.2	algebraic equations	13
Exa 2.3	algebraic equations	13
Exa 2.4	algebraic equations	14
Exa 2.5	algebraic equations	15
Exa 2.6	algebraic equations	16
Exa 3.1	thermodynamics	17
Exa 3.2	thermodynamics	19
Exa 3.3	flash calculations using Raoult law	21
Exa 3.4	BPT and DPT calculation using modified raoult law	22
Exa 3.5	flash calculations using modified Raoult law	25
Exa 3.6	vapour pressure calculation using cubic equation of state	27
Exa 3.7	pressure x y diagram using gamma phi approach	29
Exa 3.9	chemical reaction engineering 2 simultaneous reactions	31
Exa 3.10	adiabatic flame temperature	33
Exa 4.1	solution of ordinary differential equation	35
Exa 4.2	solution of ordinary differential equation	36
Exa 4.3	double pipe heat exchanger	37
Exa 4.4	stirred tank with coil heater	38
Exa 4.5	stirred tank with coil heater	39
Exa 4.6	pneumatic conveying	40
Exa 4.7	simultaneous ordinary differential equations	41

Exa 4.8	simultaneous ordinary differential equations	42
Exa 4.9	simultaneous ordinary differential equations	43
Exa 4.10	series of stirred tank with coil heater	44
Exa 4.11	batch and stirred tank reactor	45
Exa 4.12	batch and stirred tank reactor	46
Exa 4.13	batch and stirred tank reactor	47
Exa 4.14	batch and stirred tank reactor	48
Exa 4.15	plug flow reactor	50
Exa 4.16	plug flow reactor	50
Exa 4.17	plug flow reactor	52
Exa 4.18	non isothermal plug flow reactor	53
Exa 5.1	discretization in 1 D space	56
Exa 5.2	discretization in 1 D space	57
Exa 5.3	discretization in 1 D space	57
Exa 5.4	discretization in 1 D space	58
Exa 5.5	discretization in 1 D space	60
Exa 5.6	1 D steady state heat conduction	61
Exa 5.7	1 D steady state heat conduction	62
Exa 5.8	chemical reaction and diffusion in pore	64
Exa 6.1	upwind scheme	66
Exa 6.2	upwind scheme	69
Exa 7.1	boundary value problem in chemical reaction engineering	72
Exa 7.2	boundary value problem in chemical reaction engineering second order	74
Exa 8.1	first order reaction	76
Exa 8.2	second order reaction	77
Exa 8.3	non isothermal condition	78
Exa 8.4	non isothermal condition	79
Exa 9.1	transient conduction in rectangular slab	81
Exa 9.2	transient conduction in rectangular slab	82
Exa 9.3	transient conduction in cylinder	84
Exa 9.4	transient conduction in sphere	85
Exa 9.5	transient diffusion in sphere	86
Exa 10.1	discretization in 2 D space gauss seidel method	88
Exa 10.2	discretization in 2 D space gauss seidel method	89
Exa 10.3	discretization in 2 D space	90
Exa 10.8	discretization in 2 D space	91
Exa 10.9	discretization in 2 D space	92

Exa 10.10	ADI method	93
Exa 10.11	ADI method for transient heat conduction	94

Chapter 1

linear algebraic equations

Scilab code Exa 1.1 TDMA method

```
1 //ch 1 ex 1.1- solving set of algebraic equations by
   Tri diagonal Matrix Algorithm Method.
2 clc
3 disp("the soln of eg 1.1-->");
4 for i=2:7, a(i)=1;           //sub diagonal
   assignment
5 end
6 for j=1:7, b(j)=-2;        //main diagonal
   assignment
7 end
8 for k=1:6, c(k)=1;         //super diagonal
   assignment
9 end
10 d(1)=-240                  //given values
   assignment
11 for l=2:6, d(l)=-40;
12 end
13 d(7)=-60
14
15 i=1;
16 n=7;
```



```

17 beta1(i)=b(i);           //initial b is equal
    to beta since a1=0
18 gamma1(i)=d(i)/beta1(i); //since c7=0
19 m=i+1;
20 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
21 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
22 end
23 x(n)=gamma1(n);         //since c7=0
24 n1=n-i;
25 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
    j);
26 end
27
28 disp("the solution of ex 1.1 by TDMA method is");
29 for i=1:7, disp(x(i));
30 end

```

Scilab code Exa 1.2 gauss elimination method

```

1 //ch 1 ex 1.2
2 clc
3 disp("the soln of eg 1.2-->");
4 a1=10, a2=1, a3=2,           //1st row
5 b1=2, b2=10, b3=1,         //2nd row
6 c1=1, c2=2, c3=10,        //3rd row
7 d1=44, d2=51, d3=61,      //given values
8
9 b3=b3-(b1/a1)*a3           // for making b1=0
10 b2=b2-(b1/a1)*a2
11 d2=d2-(b1/a1)*d1
12 b1=b1-(b1/a1)*a1
13
14 c3=c3-(c1/a1)*a3          // for making c1=0
15 c2=c2-(c1/a1)*a2
16 d3=d3-(c1/a1)*d1

```

```

17 c1=c1-(c1/a1)*a1
18
19 c3=c3-(c2/b2)*b3           // for making c2=0
20 d3=d3-(c2/b2)*d2
21 c2=c2-(c2/b2)*b2
22
23 x3=d3/c3;                 // final values of x
24 x2=(d2-(b3*x3))/b2;
25 x1=(d1-(x3*a3)-(x2*a2))/a1;
26 disp(x3,x2,x1,"the solution using gauss elimination
    method is");

```

Scilab code Exa 1.3 gauss elimination method

```

1 //ch 1 ex 1.3
2 clc
3 disp("the soln of eg 1.3-->");
4 a1=3, a2=1, a3=-2,         //1st row
5 b1=-1, b2=4, b3=-3,       //2nd row
6 c1=1, c2=-1, c3=4,        //3rd row
7 d1=9, d2=-8, d3=1,        //given values
8
9 b3=b3-(b1/a1)*a3           // for making b1=0
10 b2=b2-(b1/a1)*a2
11 d2=d2-(b1/a1)*d1
12 b1=b1-(b1/a1)*a1
13
14 c3=c3-(c1/a1)*a3         // for making c1=0
15 c2=c2-(c1/a1)*a2
16 d3=d3-(c1/a1)*d1
17 c1=c1-(c1/a1)*a1
18
19 c3=c3-(c2/b2)*b3         // for making c2=0
20 d3=d3-(c2/b2)*d2
21 c2=c2-(c2/b2)*b2

```

```

22
23 x3=d3/c3; // final values of x
24 x2=(d2-(b3*x3))/b2;
25 x1=(d1-(x3*a3)-(x2*a2))/a1;
26 disp(x3,x2,x1,"the solution using gauss elimination
method is");

```

Scilab code Exa 1.4 gauss elimination method

```

1 // ch 1 ex 1.4
2 clc
3 disp("the solution of eg 1.4-->");
4 a1=.35, a2=.16, a3=.21, a4=.01 //1st
   row
5 b1=.54, b2=.42, b3=.54, b4=.1 //2nd row
6 c1=.04, c2=.24, c3=.1, c4=.65 //3rd row
7 d1=.07, d2=.18, d3=.15, d4=.24 //4th row
8 r1=14, r2=28, r3=17.5, r4=10.5 //given
   values
9
10 b4=b4-(b1/a1)*a4 // for
   making b1=0
11 b3=b3-(b1/a1)*a3
12 b2=b2-(b1/a1)*a2
13 r2=r2-(b1/a1)*r1
14 b1=b1-(b1/a1)*a1
15
16 c4=c4-(c1/a1)*a4 // for
   making c1=0
17 c3=c3-(c1/a1)*a3
18 c2=c2-(c1/a1)*a2
19 r3=r3-(c1/a1)*r1
20 c1=c1-(c1/a1)*a1
21
22 d4=d4-(d1/a1)*a4 // for

```

```

    making d1=0
23 d3=d3-(d1/a1)*a3
24 d2=d2-(d1/a1)*a2
25 r4=r4-(d1/a1)*r1
26 d1=d1-(d1/a1)*a1
27
28 c4=c4-(c2/b2)*b4 // for
    making c2=0
29 c3=c3-(c2/b2)*b3
30 r3=r3-(c2/b2)*r2
31 c2=c2-(c2/b2)*b2
32
33 d4=d4-(d2/b2)*b4 // for making
    d2=0
34 d3=d3-(d2/b2)*b3
35 r4=r4-(d2/b2)*r2
36 d2=d2-(d2/b2)*b2
37
38 d4=d4-(d3/c3)*c4 //for making
    d3=0
39 r4=r4-(d3/c3)*r3
40 d3=d3-(d3/c3)*c3
41
42 B2=r4/d4;
43 D2=(r3-(c4*B2))/c3;
44 B1=(r2-(D2*b3)-(B2*b4))/b2;
45 D1=(r1-(B2*a4)-(D2*a3)-(B1*a2))/a1;
46 disp(B2,D2,B1,D1,"the values of MOLAR FLOW RATES of
    D1, B1, D2, B2 respectively are");
47
48 B=D2+B2;
49 x1B=(.21*D2 + .01*B2)/B;
50 x2B=(.54*D2 + .1*B2)/B;
51 x3B=(.1*D2 + .65*B2)/B;
52 x4B=(.15*D2 + .24*B2)/B;
53 disp(x4B, x3B, x2B, x1B,"the composition of stream B
    is");
54

```

```

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

```

Scilab code Exa 1.5 gauss seidel method

```

1  clc
2  disp("the soln of eg 1.5--> Gauss Seidel Method");
3  for i=1:3,xnew(i)=2,e(i)=1
4  end
5  x=1e-6
6  while e(1)>x&e(2)>x&e(3)>x do
7      for i=1:3, xold(i)=xnew(i),end
8      xnew(1)=(44-xold(2)-2*xold(3))/10
9      xnew(2)=(-2*xnew(1)+51-xold(3))/10
10     xnew(3)=(-2*xnew(2)-xnew(1)+61)/10
11     for i=1:3,e(i)=abs(xnew(i)-xold(i))
12     end
13 end
14 disp("the values of x1,x2,x3 respectively is");
15 for i=1:3,disp(xnew(i));
16 end

```

Scilab code Exa 1.6 gauss seidel method

```

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

```

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

Chapter 2

NONLINEAR ALGEBRAIC EQUATIONS

Scilab code Exa 2.1 algebraic equations

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

Scilab code Exa 2.2 algebraic equations

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

Scilab code Exa 2.3 algebraic equations


```

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

```

Scilab code Exa 2.4 algebraic equations

```

1  clc

```

```

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

```

Scilab code Exa 2.5 algebraic equations

```

1 clc
2 disp("the soln of eg 2.5--> non linear equations");
3 xnew=0.1,ynew=0.5, e1=1, e2=1
4 while e1>1e-6 & e2>1e-6 do x=xnew,y=ynew,
5 y1=exp(x)+x*y-1,
6 d_fx=exp(x)+y
7 d_fy=x
8 y2=sin(x*y)+x+y-1,
9 d_gx=y*cos(x*y)+1
10 d_gy=x*cos(x*y)+1
11 t1=(y2*d_fy), t2=(y1*d_gy),
12 D1=d_fx*d_gy-d_fy*d_gx
13 delta_x=(t1-t2)/D1
14 t3=(y1*d_gx),t4=(y2*d_fx)
15 delta_y=(t3-t4)/D1

```

```

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

```

Scilab code Exa 2.6 algebraic equations

```

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

```

Chapter 3

CHEMICAL ENGINEERING THERMODYNAMICS

Scilab code Exa 3.1 thermodynamics

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

```

17 e1=abs(vold-vnew)
18 end
19 disp(vold,"the soln by virial gas eqn. of volume in
      m3/mol by Z(T,V) is");
20 //by peng robinson method
21 k=.37464+1.54226*w-.26992*w^2
22 s=1+k*(1-Tr^.5)
23 lpha=s^2
24 a=.45724*R^2*Tc^2*lpha/Pc
25 b=.0778*R*Tc/Pc
26 vnew=b,e2=1,
27 vol=b, fe=0, fd=0
28 disp("the volume of saturated liq. and saturated
      vapour by peng-robinson method respectively is")
29 for i=0:2 do
30     vol=vnew
31     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
32 ydash2=3*P*vol^2+(P*b-R*T)*2*vol-(3*P*b^2+2*b*R*T-a)
33 vnew=vol-y2/ydash2,
34 e2=abs(vol-vnew)
35 if i==1 & e2<1e-6 then fd=vnew,vnew=R*T/P,
36 end
37 end
38 disp(vol,fd);
39 funcprot(0)
40 //by redlich-kwong method
41 i=0
42 a=.42748*R^2*Tc^2.5/Pc
43 b=.08664*R*Tc/Pc
44 Vnew=b,e3=1
45 disp("the vol of saturated liq. and vapour by
      redlich kwong method respectively are");
46 for i=0:2 do V=Vnew,function y3=gh(V),
47     y3=V^3*P-R*T*V^2-V*(P*b^2+b*R*T-a/sqrt(T
      ))-a*b/sqrt(T)
48     endfunction,
49     deriv=derivative(gh,V);

```

```

50         Vnew=V-gh(V)/deriv;
51         e3=abs(Vnew-V)
52         if i==1 & e3<1e-6 then de=Vnew,Vnew=R*T/P
                    //for saturated liq.
53         end
54         end
55         disp(V,de);
56 //vander waals method
57 i=0
58 a=27*R^2*Tc^2/(64*Pc)
59 b=R*Tc/(8*Pc)
60 vnew=b, v=b,e=1
61 for i=0:2 do v=vnew,function v3=bh(v),
62             v3=v^3*P-v^2*(P*b+R*T)+a*v-a*b,
63             endfunction
64             deriva=derivative(bh,v),
65             vnew=v-bh(v)/deriva
66             e4=abs(v-vnew),
67             if i==1 & e4<10^-6 then sol=vnew,vnew=R*T/P
68             end
69 end
70 disp("the vol of saturated liq. and vapour by vander
        waal method respectively are");
71 disp(vnew,sol);

```

Scilab code Exa 3.2 thermodynamics

```

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

```

```

8 T2sat=b2/(a2-log(P))-c2
9 Told=273,e=1
10 Tnew=x1*T1sat+x2*T2sat
11 while e>1e-6 do Told=Tnew,function y1=fw(Told),
12     P1sat=exp(a1-b1/(Told+c1)),
13     P2sat=exp(a2-b2/(Told+c2)),
14     y1=P-x1*P1sat-x2*P2sat
15     endfunction
16     ydashd=derivative(fw,Told)
17     Tnew=Told-fw(Told)/ydashd
18     e=abs(Told-Tnew)
19 end
20 disp(Tnew,"the bubble point temp. in Celsius is"
21     );
                                                                    //
                                                                    calc
                                                                    of
                                                                    dew
                                                                    point

22 y1=z1, y2=1-z1,e=1
23 Tnew=y1*T1sat+y2*T2sat
24 while e>1e-6 do Told=Tnew,function y11=fw1(Told),
25     P1sat=exp(a1-b1/(Told+c1)),
26     P2sat=exp(a2-b2/(Told+c2)),
27     y11=1/P-y1/P1sat-y2/P2sat
28     endfunction
29     ydashd=derivative(fw1,Told)
30     Tnew=Told-fw1(Told)/ydashd
31     e=abs(Told-Tnew)
32 end
33 disp(Tnew,"the dew point temp. in Celsius is");

```

Scilab code Exa 3.3 flash calculations using Raoult law

```
1 clc
2 disp("the soln of eg 3.3-->Flash Calc. using Raoults
      Law")
3 psat(1)=195.75,psat(2)=97.84,psat(3)=50.32
      //given data
4 z(1)=.3,z(2)=.3,z(3)=.4
5 bpp=z(1)*psat(1)+z(2)*psat(2)+z(3)*psat(3)
      //Bubble point pressure
6 dpp=1/(z(1)/psat(1)+z(2)/psat(2)+z(3)/psat(3))
      //dew point pressure
7 disp(dpp,bpp,"the bubble point pressure and dew
      point pressure respectively are");
8 P=100,v=1, vnew=1, e=1,y2=0, der=0 //
      given pressure is between BPP & DPP
9 for i=1:3 k(i)=psat(i)/P
10 end
11 while e>1e-6 do v=vnew,for i=1:3,
12     t1=(1-v+v*k(i)),y2=y2+z(i)*(k(i)-1)/t1,
13     der=der-z(i)*(k(i)-1)^2/t1^2,
14     end
15     vnew=v-y2/der,e=abs(vnew-v),
16     end
17     for i=1:3, x(i)=z(i)/(1-v+v*k(i)), y(i)=x(i)*k(i)
18     )
19     end
20     disp(x(1),"mol frctn of acetone in liq. phase is
21     ");
22     disp(y(1),"mol frctn of acetone in vapour phase
23     is");
24     disp(x(2),"mol frctn of acetonitrile in liq.
25     phase is");
26     disp(y(2),"mol frctn of acetonitrile in vapour.
```



```

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

```

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

```

1  a12=437.98*4.186, a21=1238*4.186, v1=76.92, v2=18.07
2
3  clc
4  disp("the soln of eg 3.4-->");
5  t=100
6  x1=.5, R=8.314
7  a1=16.678, b1=3640.2, c1=219.61
8  a2=16.2887, b2=3816.44, c2=227.02
9  x2=1-x1
10 p1sat=exp(a1-b1/(c1+t))
11 p2sat=exp(a2-b2/(c2+t))
12 h12=v2*exp(-a12/(R*(t+273.15)))/v1
13 h21=v1*exp(-a21/(R*(t+273.15)))/v2
14 m=h12/(x1+x2*h12)-h21/(x2+x1*h21)
15 g1=exp(-log(x1+x2*h12)+x2*m)
16 g2=exp(-log(x2+x1*h21)-x1*m)
17 p=x1*g1*p1sat+x2*g2*p2sat
18 disp(p,"boiling point pressure in kPa is");
19

```

//
ca
of
BP

//
ca

```

20 p=101.325,x1=.5, e=1
21 x2=1-x1
22 t1sat=b1/(a1-log(p))-c1
23 t2sat=b2/(a2-log(p))-c2
24 tnew=x1*t1sat+x2*t2sat
25 while e>10^-4 do told=tnew,
26     p1sat=exp(a1-b1/(c1+told)),p2sat=exp(a2-b2/(c2+
        told)),
27     p1sat=p/(g1*x1+g2*x2*(p2sat/p1sat))
28     tnew=b1/(a1-log(p1sat))-c1,
29     e=abs(tnew-told)
30 end
31 disp(tnew,"boiling point temperature in Celsius is")
    ;
32

33 e1=1, e2=1, e3=1, pold=1
34 t=100,y1=.5
35 y2=1-y1
36 p1sat=exp(a1-b1/(c1+t))
37 p2sat=exp(a2-b2/(c2+t))
38 g1=1, g2=1, g11=1, g22=1
39 pnew=1/(y1/(g1*p1sat)+y2/(g2*p2sat))
40 while e1>.0001 do pold=pnew, while e2>.0001& e3
        >.0001 do g1=g11,g2=g22,
41     x1=y1*pold/(g1*p1sat)
42     x2=y2*pold/(g2*p2sat)
43     x1=x1/(x1+x2)
44     x2=1-x1

```

```

45         h12=v2*exp(-a12/(R*(t
46             +273.15)))/v1
47         h21=v1*exp(-a21/(R*(t
48             +273.15)))/v2
49         m=h12/(x1+x2*h12)-h21
50             /(x2+x1*h21)
51         g11=exp(-log(x1+x2*h12
52             )+x2*m)
53         g22=exp(-log(x2+x1*h21
54             )-x1*m)
55         e2=abs(g11-g1), e3=abs
56             (g22-g2)
57
58         end
59         pnew=1/(y1/(g1*p1sat)+y2/(g2*p2sat))
60         e1=abs(pnew-pold)
61     end
62     disp(pnew,"dew point pressure in kPa is");
63

```

```

//
calc
dpt

```

```

64 p=101.325,y1=.5, e4=1, e5=1,e6=1
65 y2=1-y1
66 t1sat=b1/(a1-log(p))-c1
67 t2sat=b2/(a2-log(p))-c2
68 tnew=y1*t1sat+y2*t2sat
69 g11=1, g22=1
70 while e4>.0001 do told=tnew,
71     p1sat=exp(a1-b1/(c1+told))
72     p2sat=exp(a2-b2/(c2+told)), while e5>.0001 & e6
73     >.0001 do g1=g11, g2=g22,
74     x1=y1*p/(g1*p1sat)
75     x2=y2*p/(g2*p2sat)
76     x1=x1/(x1+x2)
77     x2=1-x1
78     h12=v2*exp(-a12/(R*(t+273.15)))/v1
79     h21=v1*exp(-a21/(R*(t+273.15)))/v2

```

```

72         m=h12/(x1+x2*h12)-h21
           /(x2+x1*h21)
73         g11=exp(-log(x1+x2*h12
           )+x2*m)
74         g22=exp(-log(x2+x1*h21
           )-x1*m)
75         e5=abs(g11-g1), e6=abs
           (g22-g2)
76         end
77
78         p1sat=p*(y1/g1+y2*p1sat/(g2*p2sat))
79         tnew=b1/(a1-log(p1sat))-c1
80         e4=abs(tnew-told)
81     end
82     disp(tnew,"dew point temperature in Celsius is")
       ;

```

Scilab code Exa 3.5 flash calculations using modified Raoult law

```

1  clc
2  disp("the soln of eg 3.5-->Flash calc. using
       Modified Raoult law");
3  a12=292.66*4.18, a21=1445.26*4.18, v1=74.05*10^-6,
       v2=18.07*10^-6, R=8.314
4  t=100, z1=.3,
5  z2=1-z1
6  a1=14.39155, a2=16.262, b1=2795.82, b2=3799.89, c1
       =230.002, c2=226.35
7  e1=1, e2=1, e3=1, e4=1, e5=1, e6=1, vnew=0
8
                                           // calc
                                           of
                                           BPP
9  x1=z1, x2=z2
10 p1sat=exp(a1-(b1/(t+c1)))
11 p2sat=exp(a2-(b2/(t+c2)))

```

```

12 h12=v2*exp(-a12/(R*(t+273.15)))/v1
13 h21=v1*exp(-a21/(R*(t+273.15)))/v2
14 m=(h12/(x1+x2*h12))-(h21/(x2+x1*h21))
15 g1=exp(-log(x1+x2*h12)+x2*m)
16 g2=exp(-log(x2+x1*h21)-x1*m)
17 p=x1*g1*p1sat+x2*g2*p2sat
18 disp(p,"the bubble point pressure is");
19 bpp=p, gb1=g1, gb2=g2 //g1
    & g2 are activity co-efficients //
20 // calc
    of
    DPP

21 y1=z1, y2=z2
22 g1=1, g2=1
23 pnew=1/(y1/(g1*p1sat)+y2/(g2*p2sat))
24 g1n=g1, g2n=g2
25 while e1>.0001 do pold=pnew,while e2>.0001& e3>.0001
    do g1=g1n, g2=g2n,
26     x1=y1*pold/(g1*p1sat)
27     x2=y2*pold/(g2*p2sat)
28     x1=x1/(x1+x2)
29     x2=1-x1
30     g1n=exp(-log(x1+x2*h12)+x2*m)
31     g2n=exp(-log(x2+x1*h21)-x1*m)
32     e2=abs(g1n-g1), e3=abs(g2n-g2)
33 end
34 pnew=1/(y1/(g1n*p1sat)+y2/(g2n*p2sat))
35 e1=abs(pnew-pold)
36 end
37 disp(pnew,"the dew point pressure is");
38 dpp=pnew, gd1=g1n, gd2=g2n
39 p=200
40 v=(bpp-p)/(bpp-dpp)
41 g1=((p-dpp)*(gb1-gd1))/(bpp-dpp)+gd1

```

```

42 g2=((p-dpp)*(gb2-gd2))/(bpp-dpp)+gd2
43
44 //calc of distribution co-efficients
45 while e4>.0001 & e5>.0001 do g1n=g1,g2n=g2,
46     k1=g1n*p1sat/p
47     k2=g2n*p2sat/p
48     while e6>.00001 do v=vnew,
49         function f=fv(v),
50             y1=(k1*z1)/(1-v+v*k1)
51             y2=(k2*z2)/(1-v+v*k2)
52             x1=y1/k1
53             x2=y2/k2
54             f=y1-x1+y2-x2
55         endfunction
56         derv=derivative(fv,v)
57         vnew=v-fv(v)/derv
58         e6=abs(vnew-v)
59     end
60     h12=v2*exp(-a12/(R*(t+273.15)))/v1
61     h21=v1*exp(-a21/(R*(t+273.15)))/v2
62     m=(h12/(x1+x2*h12))-(h21/(x2+x1*h21))
63     g1=exp(-log(x1+x2*h12)+x2*m)
64     g2=exp(-log(x2+x1*h21)-x1*m)
65     e4=abs(g1-g1n), e5=abs(g2-g2n)
66 end
67 disp(v,"the no. of moles in vapour phase is");
68 disp(y1,x1,"x1 and y1 respectively are");

```

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

```

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

```

```

//given
4 f1=1, e1=1, e2=1, vnew=1, pnew=1 //
  assumed values
5 k=.37464+1.54226*w-.26992*w*2
6 s=(1+k*(1-(t/tc)^.5))^2,
7 a=.45724*R*R*tc*tc*s/pc
8 b=.0778*R*tc/pc
9 //calc of vol. of liq.
10 while f1>10^-4 do p=pnew,vnew=b,
11 t1=(p*b-R*t) //co-efficients
  of vol. in the eqn.
12 t2=3*p*b^2+2*b*R*t-a
13 t3=p*b^3+b^2*R*t-a*b
14   while e1>1e-6 & vnew>0 do vold=vnew,
15     y1=vold^3*p+vold^2*t1-vold*t2+t3,
16     der=3*vold^2*p+2*vold*t1-t2
17     vnew=vold-y1/der
18     e1=abs(vnew-vold)
19 end
20 vliq=vold,
21 //fugacity of liq.
22 zliq=p*vliq/(R*t)
23 c=(a/(2*1.414*b*R*t))*(log((vliq+(1+sqrt(2))*b)/(
  vliq+(1-sqrt(2))*b))),
24 t5=zliq-p*b/(R*t)
25 f12=p*exp(zliq-1-log(t5)-c),
26 vvnew=R*t/p, //assumed value close
  to the ideal value
27 //calc of vol. of vapour
28 while e2>1e-6 do vvold=vvnew,
29   x2=vvold^3*p+vvold^2*t1-vvold*t2+t3,
30   der1=3*vvold^2*p+2*vvold*t1-t2,
31   vvnew=vvold-x2/der1
32 e2=abs(vvnew-vvold)
33 end
34 //fugacity of vapour
35 vvap=vvold,zvap=p*vvap/(R*t),
36 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)))
37 t6=zvap-p*b/(R*t)
38 fv2=p*exp(zvap-1-log(t6)-d)
39 pnew=p*f12/fv2 //updating the
    value of P
40 f1=abs((f12-fv2)/fv2)
41 end
42 disp(p,"the vapour pressure of water in Pa is");

```

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

```

1 clc
2 disp("the soln of eg 3.7-->P-x-y calc. using Gamma-
    Phi approach");
3 et=1,er=1,sold=0, snew=0 //assumed
    values
4 R=8.314,t=100, x1=.958, a12=107.38*4.18, a21
    =469.55*4.18,
5 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
6 x2=1-x1,
7 a1=16.5938, a2=16.262, b1=3644.3, b2=3799.89, c1
    =239.76, c2=226.35
8 p1sat=exp(a1-b1/(c1+t))*1000 //Saturation Pressure
9 p2sat=exp(a2-b2/(c2+t))*1000
10 t=t+273.15 //Temp
    in Kelvin req.
11 h12=(v2/v1)*exp(-a12/(R*t))
12 h21=(v1/v2)*exp(-a21/(R*t))
13 z=h12/(x1+x2*h12)-h21/(x2+x1*h21)
14 g1=exp(-log(x1+x2*h12)+x2*z) //Activity Co-efficients

```



```

15 g2=exp(-log(x2+x1*h21)-x1*z)
16 tr1=t/tc1 //Reduced
    Temp.
17 b0=.083-.422*tr1^-1.6
18 b1=.139-.172*tr1^-4.2
19 b11=(R*tc1/pc1)*(b0+b1*w1)
20 tr2=t/tc2
21 b0=.083-.422*tr2^-1.6
22 b1=.139-.172*tr2^-4.2
23 b22=(R*tc2/pc2)*(b0+b1*w2)
24 w12=(w1+w2)^.5
25 tc12=(tc1*tc2)^.5
26 zc12=(zc1+zc2)^.5
27 vc1=zc1*R*tc1/pc1, vc2=zc2*R*tc2/pc2
28 vc12=((vc1^.33+vc2^.33)/2)^3
29 pc12=zc12*R*tc12/vc12
30 tr12=t/tc12
31 b0=.083-.422*tr12^-1.6
32 b1=.139-.172*tr12^-4.2
33 b12=(R*tc12/pc12)*(b0+b1*w12)
34 d12=2*b12-b11-b22
35 p=x1*g1*p1sat+x2*p2sat*g2
36 y1=x1*g1*p1sat/p, y2=x2*g2*p2sat/p
37 pnew=p,
38 //calc of Pressure
39 while et>1e-6 do p=pnew,
40     f1=p1sat*(exp(b11*p1sat/(R*t)))*(exp((v1*(p-
        p1sat)/(R*t))))
41 f2=p2sat*(exp(b22*p2sat/(R*t)))*(exp((v2*(p-p2sat)/(
        R*t))))
42 while er>1e-6 do sold=snew,
43     fc1=exp((p/(R*t))*(b11+y2^2*d12))
44     fc2=exp((p/(R*t))*(b22+y1^2*d12))
45     k1=g1*f1/(fc1*p)
46     k2=g2*f2/(fc2*p)
47     snew=x1*k1+x2*k2
48     y1=x1*k1/snew
49     y2=x2*k2/snew

```

```

50 er=abs(snew-sold)
51 end
52 pnew=(x1*g1*f1/fc1)+(x2*g2*f2/fc2)
53 y1=x1*g1*f1/(fc1*pnew)
54 y2=x2*g2*f2/(fc2*pnew)
55 et=abs(pnew-p)
56 end
57 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

```

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

```

```

    *(6+2*x1)^2)
16  d4=(3*x1+x2)^3*(x1-x2)*(8*x1^2+6*x1*x2-24*x1+2*
    x2-16)
17  d5=((1-x1)^2)*((5-x1-x2)^2)*((6+2*x1)^3)
18  df1_dx1=d3-(d4/d5) //df1
    /dx1- partial derivative of f1 wrt to x1
19  d6=3*(x1-x2)*((3*x1+x2)^2)-(3*x1+x2)^3
20  d7=(1-x1)*(5-x1-x2)*((6+x1*2)^2)
21  d8=((x1-x2)*(3*x1+x2)^3)/((1-x1)*((5-x1-x2)^2)
    *(6+2*x1)^2)
22  df1_dx2=(d6/d7)+d8 //df1
    /dx2- partial derivative of f1 wrt to x2
23  d9=(x1-x2)*(5-x1-x2)
24  df2_dx1=3*x2/d9-(x2*(3*x1+x2)*(5-2*x1))/(d9^2)
    //df1/dx2- partial derivative of f1
    wrt to x2
25  d10=(3*x1+2*x2)/d9
26  d11=x2*(3*x1+x2)*(2*x2-5)/(d9^2)
27  df2_dx2=d10-d11 //
    df1/dx2- partial derivative of f1 wrt to x2
28  dm=df1_dx1*df2_dx2-df1_dx2*df2_dx1
29  delta_x1=(f2*df1_dx2-f1*df2_dx2)/dm
30  delta_x2=(f1*df2_dx1-f2*df1_dx1)/dm
31  x1new=x1+delta_x1 //updating
    the values of x1 & x2
32  x2new=x2+delta_x2
33  r1=abs(x1-x1new), r2=abs(x2new-x2)
34  end
35  disp(x2,x1,"the value of X1 and X2 respectively is")
    ;
36  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.
37  total=m_CO2+m_H2+m_CO+m_H2O+m_CH4
38  disp(m_CO2,m_H2,m_CO,m_H2O,m_CH4,"the moles at eqm
    of CH4,H2O,CO,H2,CO2 are")
39  disp(total,"total number of moles at eqm. is")

```

Scilab code Exa 3.10 adiabatic flame temperature

```
1  clc
2  disp("the soln of eg 3.10-->Adiabatic Flame
      Temperature");
3  u1=1,u2=3.5,u3=2,u4=3 //moles
      given 1-C2H6, 2-O2, 3-CO2, 4-H2O
4  a1=1.648,a2=6.085,a3=5.316,a4=7.7
5  b1=4.124e-2,b2=.3631e-2,b3=1.4285e-2,b4=.04594e-2
6  c1=-1.53e-5,c2=-.1709e-5,c3=-.8362e-5,c4=.2521e-5
7  d1=1.74e-9,d2=.3133e-9,d3=1.784e-9,d4=-.8587e-9
8  n1=1,n2=4,n3=10,n4=0,t0=298.15,t1=25,e1=1
9  t1=t1+273.15
10 //calc. of sum of co-efficients of heat capacity of
      the rxn.
11 sa=n1*a1+n2*a2+n3*a3+n4*a4
12 sb=n1*b1+n2*b2+n3*b3+n4*b4
13 sc=n1*c1+n2*c2+n3*c3+n4*c4
14 sd=n1*d1+n2*d2+n3*d3+n4*d4
15 da=u4*a4+u3*a3-u2*a2-u1*a1
16 db=u4*b4+u3*b3-u2*b2-u1*b1
17 dc=u4*c4+u3*c3-u2*c2-u1*c1
18 dd=u4*d4+u3*d3-u2*d2-u1*d1
19 h0=(u4*(-57.798)+u3*(-94.05)-u2*0-u1*(-20.236))*1000
      //enthalpy of the rxn.
20 tnew=1000
21 while e1>1e-6 do t=tnew,
22     function f=ft(t),
23     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)
24 endfunction
25 dr=derivative(ft,t),
26 tnew=t-ft(t)/dr,
```

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

Chapter 4

INITIAL VALUE PROBLEMS

Scilab code Exa 4.1 solution of ordinary differential equation

```
1 clc
2 disp("the solution of e.g. 4.1 -->Integration of
   Ordinary Differential Equation")
3 // in this problem dy/dx=x+y
4 x_0=0 //initial values given
5 y_0=0
6
7 function ydash=fs(x,y),
8     ydash=x+y,
9 endfunction
10
11 for x_0=0:0.1:0.2,
12     h=0.1 //step
13     // increment of 0.1
14     f_0=fs(x_0,y_0)
15     k1=h*f_0
16     k2=h*fs(x_0+h/2,y_0+k1/2)
17     k3=h*fs(x_0+h/2,y_0+k2/2)
18     k4=h*fs(x_0+h,y_0+k3)
19     y_0=y_0+(k1+2*k2+2*k3+k4)/6
20 end
```

```

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

```

Scilab code Exa 4.2 solution of ordinary differential equation

```

1 clc
2 disp("the solution of e.g. 4.2 -->Ordinary
    Differential Eqn.-Runge Kutta method")
3 // in this problem dy/dx=-y/(1+x)
4 x_0=0 //initial values given
5 y_0=2
6 function ydash=fr(x,y),
7     ydash=-y/(1+x),
8 endfunction
9 for x_0=0:0.01:2.5,
10     h=0.01 //step
        increment of 0.01
11     f_0=fr(x_0,y_0)
12     k1=h*f_0
13     k2=h*fr(x_0+h/2,y_0+k1/2)
14     k3=h*fr(x_0+h/2,y_0+k2/2)
15     k4=h*fr(x_0+h,y_0+k3)
16     y_0=y_0+(k1+2*k2+2*k3+k4)/6
17 end
18 y_0=y_0-(k1+2*k2+2*k3+k4)/6 //
    final value at x=2.5
19 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

```
1  clc
2  disp("the solution of e.g. 4.3 -->Double Pipe Heat
      Exchanger");
3  rho=1000, v=1, dia=2.4*10^-2, Cp=4184           //
      given data
4  mdot=rho*v*%pi*dia^2/4
5  t1=mdot*Cp
6  U=200
7  Ts=250
8  z=0                                           //initial values given
9  // dT/dz=U*pi*dia*(Ts-T)/(mdot*Cp)
10 function Tgrad=fr(z,T),
11     Tgrad=U*%pi*dia*(Ts-T)/(mdot*Cp),
12 endfunction
13 T=20
14 for z=0:0.01:10,
15     h=0.01                                     //step
      increment of 0.01
16     k1=h*fr(z,T)
17     k2=h*fr(z+h/2,T+k1/2)
18     k3=h*fr(z+h/2,T+k2/2)
19     k4=h*fr(z+h,T+k3)
20     T=T+(k1+2*k2+2*k3+k4)/6
21     if z==5 then T=T-(k1+2*k2+2*k3+k4)/6,
22         disp(T,"the value of T in deg Celsius at z=5
      m using Runge Kutta method is");
23     end
24 end
25 T=T-(k1+2*k2+2*k3+k4)/6                       //final
      value at z=10 m
26 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

```
1  clc
2  disp("the solution of e.g. 4.4 -->Stirred Tank with
      Coil Heater")
3  vol=.5*.5*2           //given data
4  rho=1000
5  m=vol*rho
6  vol_rate=.001
7  mdot=vol_rate*rho
8  out_A=1
9  U=200
10 Cp=4184
11 T1=20, Ts=250, T_exit=28
      //temp in Celsius
12 t1=(mdot*Cp*T1+U*out_A*Ts)/(m*Cp)
      //terms of the function
13 t2=(mdot*Cp+U*out_A)/(m*Cp)
14 //dt/dt=(mdot*Cp(T1-T)+Q_dot)/m*Cp
15 function tgrad=fv(tm,T),
16     tgrad=t1-t2*T
17 endfunction
18 T=20           //
      initial value
19 funcprot(0)
20 for tm=0:1:5000,
21     h=1           //step
      increment of 1 sec
22     k1=h*fv(tm,T)
23     k2=h*fv(tm+h/2,T+k1/2)
24     k3=h*fv(tm+h/2,T+k2/2)
25     k4=h*fv(tm+h,T+k3)
26     e1=abs(T-T_exit)
27     if e1<1e-3 then disp(tm,"the time at which exit
```

```

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

```

Scilab code Exa 4.5 stirred tank with coil heater

```

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

```

```

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

```

Scilab code Exa 4.6 pneumatic conveying

```

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

```

```

25     kk1=h*fy(z,y)
26     kk2=h*fy(z+h/2,y+kk1/2)
27     kk3=h*fy(z+h/2,y+kk2/2)
28     kk4=h*fy(z+h,y+kk3)
29     y=y+(kk1+2*kk2+2*kk3+kk4)/6
30     end
31     v=sqrt(y)                //final value of
    velocity
32     disp(v,"the value of v at the end of the pneumatic
    conveyor is");

```

Scilab code Exa 4.7 simultaneous ordinary differential equations

```

1  clc
2  disp("the soln of eg 4.7-->Simultaneous O.D.E.")
3  function dx_dt=fw(t,x,y);
4      dx_dt=x+2*y,
5  endfunction
6  function dy_dt=fq(t,x,y);
7      dy_dt=3*x+2*y
8  endfunction
9  y=4,x=6                //initial values
10 //solving by Runge-Kutta method
11 for t=0:.1:.2,
12     h=.1                //step
    increment of 0.1
13     k1=h*fw(t,x,y)
14     l1=h*fq(t,x,y)
15     k2=h*fw(t+h/2,x+k1/2,y+l1/2)
16     l2=h*fq(t+h/2,x+k1/2,y+l1/2)
17     k3=h*fw(t+h/2,x+k2/2,y+l2/2)
18     l3=h*fq(t+h/2,x+k2/2,y+l2/2)
19     k4=h*fw(t+h,x+k3,y+l3)
20     l4=h*fq(t+h,x+k3,y+l3)
21     x=x+(k1+2*k2+2*k3+k4)/6

```

```

22     y=y+(l1+2*l2+2*l3+l4)/6
23     end
24 x=x-(k1+2*k2+2*k3+k4)/6
25 y=y-(l1+2*l2+2*l3+l4)/6
26 disp(y,x,"the values of x and y repectively are");
27 t_an=.2
28 x_an=4*exp(4*t)+2*exp(-t)
29 y_an=6*exp(4*t)-2*exp(-t)
30 disp(y_an,x_an,"the analytical values of x and y are
    respectively");

```

Scilab code Exa 4.8 simultaneous ordinary differential equations

```

1  clc
2  disp("the soln of eg 4.8-->Simultaneous O.D.E.")
3  function dy_dx=fw(x,y,z);
4      dy_dx=z,
5  endfunction
6  function dz_dx=fq(x,y,z);
7      dz_dx=-y
8  endfunction
9  y=2,z=1 //initial values
10 for x=0:.1:3,
11     h=.1 //step
12     //increment of 0.1
12     k1=h*fw(x,y,z)
13     l1=h*fq(x,y,z)
14     k2=h*fw(x+h/2,y+k1/2,z+l1/2)
15     l2=h*fq(x+h/2,y+k1/2,z+l1/2)
16     k3=h*fw(x+h/2,y+k2/2,z+l2/2)
17     l3=h*fq(x+h/2,y+k2/2,z+l2/2)
18     k4=h*fw(x+h,y+k3,z+l3)
19     l4=h*fq(x+h,y+k3,z+l3)
20     y=y+(k1+2*k2+2*k3+k4)/6
21     z=z+(l1+2*l2+2*l3+l4)/6

```

```

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

```

Scilab code Exa 4.9 simultaneous ordinary differential equations

```

1  clc
2  disp("the soln of eg 4.9-->Simultaneous O.D.E.")
3  function dy_dx=fw(x,y,z); //let us
    have dy/dx=z, therefore d2y/dx2=dz/dx
4      dy_dx=z,
5  endfunction
6  function dz_dx=fq(x,y,z);
7      dz_dx=-y*x,
8  endfunction
9  y=2,z=1
10 for x=0:.1:3,
11     h=.1 //step
        increment of 0.1
12     k1=h*fw(x,y,z)
13     l1=h*fq(x,y,z)
14     k2=h*fw(x+h/2,y+k1/2,z+l1/2)
15     l2=h*fq(x+h/2,y+k1/2,z+l1/2)
16     k3=h*fw(x+h/2,y+k2/2,z+l2/2)
17     l3=h*fq(x+h/2,y+k2/2,z+l2/2)

```

```

18     k4=h*fw(x+h,y+k3,z+l3)
19     l4=h*fq(x+h,y+k3,z+l3)
20     y=y+(k1+2*k2+2*k3+k4)/6
21     z=z+(l1+2*l2+2*l3+l4)/6
22     end
23 y=y-(k1+2*k2+2*k3+k4)/6
24 z=z-(l1+2*l2+2*l3+l4)/6
25 disp(z,y,"the values of y and z repectively are");

```

Scilab code Exa 4.10 series of stirred tank with coil heater

```

1  clc
2  disp("the solution of eg 4.10 -->Series of Stirred
      Tanks with Coil Heaters")
3
4  Cp=2000,A=1,U=200,m=1000,mdot=2,Ts=250
      //given data
5  T0=20, T1=0, T2=0, T3=0
6
7  //from energy balances for the tanks we have
      accumulation=inlet-outlet
8  T1_steady=(mdot*Cp*(T0)+U*A*(Ts))/(mdot*Cp+U*A)
9  disp(T1_steady,"the steady state temperature of tank
      1 is");
10 T2_steady=(mdot*Cp*(T1_steady)+U*A*(Ts))/(mdot*Cp+U*
      A)
11 disp(T2_steady,"the steady state temperature of tank
      2 is");
12 T3_steady=(mdot*Cp*(T2_steady)+U*A*(Ts))/(mdot*Cp+U*
      A)
13 disp(T3_steady,"the steady state temperature of tank
      3 is");
14 final_T3=.99*T3_steady
15 function dT1_by_dt=f1(t,T1,T2,T3),
16     dT1_by_dt=(mdot*Cp*(T0-T1)+U*A*(Ts-T1))/(m*Cp),

```

```

17 endfunction
18 function dT2_by_dt=f2(t,T1,T2,T3),
19     dT2_by_dt=(mdot*Cp*(T1-T2)+U*A*(Ts-T2))/(m*Cp),
20 endfunction
21 function dT3_by_dt=f3(t,T1,T2,T3),
22     dT3_by_dt=(mdot*Cp*(T2-T3)+U*A*(Ts-T3))/(m*Cp),
23 endfunction
24 T1=20,T2=20,T3=20
25 //solving by Newton's Method
26 for t=0:1:10000,
27     h=1 //step
28     increment of 1
29     k1=h*f1(t,T1,T2,T3)
30     l1=h*f2(t,T1,T2,T3)
31     m1=h*f3(t,T1,T2,T3)
32     k2=h*f1(t+h/2,T1+k1/2,T2+l1/2,T3+m1/2)
33     l2=h*f2(t+h/2,T1+k1/2,T2+l1/2,T3+m1/2)
34     m2=h*f3(t+h/2,T1+k1/2,T2+l1/2,T3+m1/2)
35     k3=h*f1(t+h/2,T1+k2/2,T2+l2/2,T3+m2/2)
36     l3=h*f2(t+h/2,T1+k2/2,T2+l2/2,T3+m2/2)
37     m3=h*f3(t+h/2,T1+k2/2,T2+l2/2,T3+m2/2)
38     k4=h*f1(t+h,T1+k3,T2+l3,T3+m3)
39     l4=h*f2(t+h,T1+k3,T2+l3,T3+m3)
40     m4=h*f3(t+h,T1+k3,T2+l3,T3+m3)
41     T1=T1+(k1+2*k2+2*k3+k4)/6
42     T2=T2+(l1+2*l2+2*l3+l4)/6
43     e1=abs(T3-final_T3)
44     if e1<1e-3 then disp(t,"the approx. time when
45         Temperature in 3rd tank is 99% of steady
46         value is"); break
47     end
48     T3=T3+(m1+2*m2+2*m3+m4)/6
49 end

```

Scilab code Exa 4.11 batch and stirred tank reactor


```

1  clc
2  //batch reactors
3  disp("the solution of e.g. 4.11 -->Batch and Stirred
      Tank Reactors")
4  //rxn given A--> B
5  rate_const_k=1
6  function dCa_by_dt=fs1(t,Ca),
7      dCa_by_dt=-rate_const_k*Ca,
8  endfunction
9  Ca=1
10 for t=0.1:0.1:3,
11     h=0.1                                     //step
12     increment of 0.1
13     k1=h*fs1(t,Ca)
14     k2=h*fs1(t+h/2,Ca+k1/2)
15     k3=h*fs1(t+h/2,Ca+k2/2)
16     k4=h*fs1(t+h,Ca+k3)
17     Ca=Ca+(k1+2*k2+2*k3+k4)/6
18 end
19 disp(Ca,"the value of conc. at t=3 using Runge Kutta
      method is");
20 Ca_anl=exp(-t)                                //analytical
      solution
21 disp(Ca_anl,"the analytical soln. is")

```

Scilab code Exa 4.12 batch and stirred tank reactor

```

1  clc
2  //rxn A-->B
3  //input=FCa0, output=FCa
4  //applying mass balance of component A we get  $d(V*Ca)/dt=F*Ca_0-F*Ca-k*Ca*V$ 
5  disp("the solution of e.g. 4.12 -->Batch and Stirred
      Tank Reactors")
6  rate_const_k=1

```

```

7 Ca0=1,F=1, V=10
8
9 function dVCa_by_dt=fr(t,Ca1),
10     dVCa_by_dt=F*Ca0-F*Ca1-rate_const_k*Ca1*V,
11 endfunction
12 Ca1=1
13 for t=0.1:0.1:10,
14     h=0.1                                     //step
15     increment of 0.1
16     k1=h*fr(t,Ca1)
17     k2=h*fr(t+h/2,Ca1+k1/2)
18     k3=h*fr(t+h/2,Ca1+k2/2)
19     k4=h*fr(t+h,Ca1+k3)
20     Ca1=Ca1+(k1+2*k2+2*k3+k4)/6
21 end                                           //final value
22 disp(Ca1,"the value of Ca at t=10 s using Runge
23     Kutta method is");
24 Ca_steady=F*Ca0/(F+rate_const_k*V)
25 disp(Ca_steady,"the steady state value of conc. is")
26 ;

```

Scilab code Exa 4.13 batch and stirred tank reactor

```

1 clc
2 //given rxn A→B→C
3 k1=1, k2=1                                     //given data
4 disp("the solution of eg 4.13 →Batch Reactors")
5 function dA_by_dt=f1a(t,A,B,C),               //
6     dA_by_dt=-A,
7 endfunction
8 function dB_by_dt=f2a(t,A,B,C),
9     dB_by_dt=A-B,
10 endfunction
11 function dC_by_dt=f3a(t,A,B,C),

```

```

12     dC_by_dt=B,
13 endfunction
14 A=1,B=0,C=0           //initial values
15 for t=0.1:.1:10,
16     h=.1               //step
17     increment of 0.1
18     k1=h*f1a(t,A,B,C)
19     l1=h*f2a(t,A,B,C)
20     m1=h*f3a(t,A,B,C)
21     k2=h*f1a(t+h/2,A+k1/2,B+l1/2,C+m1/2)
22     l2=h*f2a(t+h/2,A+k1/2,B+l1/2,C+m1/2)
23     m2=h*f3a(t+h/2,A+k1/2,B+l1/2,C+m1/2)
24     k3=h*f1a(t+h/2,A+k2/2,B+l2/2,C+m2/2)
25     l3=h*f2a(t+h/2,A+k2/2,B+l2/2,C+m2/2)
26     m3=h*f3a(t+h/2,A+k2/2,B+l2/2,C+m2/2)
27     k4=h*f1a(t+h,A+k3,B+l3,C+m3)
28     l4=h*f2a(t+h,A+k3,B+l3,C+m3)
29     m4=h*f3a(t+h,A+k3,B+l3,C+m3)
30     A=A+(k1+2*k2+2*k3+k4)/6
31     B=B+(l1+2*l2+2*l3+l4)/6
32     C=C+(m1+2*m2+2*m3+m4)/6
33     if t==.5 |t==1|t==2|t==5 then disp(C,B,A,"secs
34         is",t,"the conc. of A,B,C after");
35     end
36 end
37 disp(C,B,A,"the conc. of A,B,C after 10 secs
38     respectively is");

```

Scilab code Exa 4.14 batch and stirred tank reactor

```

1  clc
2  //given rxn A+B $\xrightarrow{k_1}$ C
3  //          B+C $\xrightarrow{k_2}$ D
4  k1=1, k2=1           //given rate
   constants

```

```

5 disp("the solution of eg 4.14 -->Batch Reactors")
6 function dA_by_dt=f1a(t,A,B,C,D),
7     dA_by_dt=-A*B,
8 endfunction
9 function dB_by_dt=f2a(t,A,B,C,D),
10    dB_by_dt=-A*B-B*C,
11 endfunction
12 function dC_by_dt=f3a(t,A,B,C,D),
13    dC_by_dt=A*B-B*C,
14 endfunction
15 function dD_by_dt=f4a(t,A,B,C,D),
16    dD_by_dt=B*C,
17 endfunction
18 A=1,B=1,C=0,D=0                                     //initial
    values
19 for t=.1:.1:10,
20    h=.1                                             //step
        increment of 0.1
21    k1=h*f1a(t,A,B,C,D)
22    l1=h*f2a(t,A,B,C,D)
23    m1=h*f3a(t,A,B,C,D)
24    n1=h*f4a(t,A,B,C,D)
25    k2=h*f1a(t+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
26    l2=h*f2a(t+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
27    m2=h*f3a(t+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
28    n2=h*f4a(t+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
29    k3=h*f1a(t+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
30    l3=h*f2a(t+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
31    m3=h*f3a(t+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
32    n3=h*f4a(t+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
33    k4=h*f1a(t+h,A+k3,B+l3,C+m3,D+n3)
34    l4=h*f2a(t+h,A+k3,B+l3,C+m3,D+n3)
35    m4=h*f3a(t+h,A+k3,B+l3,C+m3,D+n3)
36    n4=h*f4a(t+h,A+k3,B+l3,C+m3,D+n3)
37    A=A+(k1+2*k2+2*k3+k4)/6
38    B=B+(l1+2*l2+2*l3+l4)/6
39    C=C+(m1+2*m2+2*m3+m4)/6
40    D=D+(n1+2*n2+2*n3+n4)/6

```

```

41     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");
42     end
43 end
44     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

```

1  clc
2  disp("the solution of eg 4.15 -->Plug Flow Reactor")
3  rc_k1=1 //given rate constant
4  u=1 //mean axial velocity
5  function dCa_by_dx=fm(x,Ca),
6      dCa_by_dx=-Ca,
7  endfunction
8  Ca=1
9  for x=.1:.1:10,
10     h=0.1 //step
        increment of 0.1
11     k1=h*fm(x,Ca)
12     k2=h*fm(x+h/2,Ca+k1/2)
13     k3=h*fm(x+h/2,Ca+k2/2)
14     k4=h*fm(x+h,Ca+k3)
15     Ca=Ca+(k1+2*k2+2*k3+k4)/6
16     if x==.5|x==1|x==2|x==5 then disp(Ca,"length is"
        ,x,"the value of conc. at");
17     end
18 end
19 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

```

1  clc
2  //given rxn A→B→C
3  rc_k1=1, rc_k2=1                                //given
   rate constants
4  u=1                                              //mean axial
   velocity
5  disp("the solution of eg 4.16 →Plug Flow Reactor")
6  function dA_by_dx=f1e(x,A,B,C),
7     dA_by_dx=-A,
8  endfunction
9  function dB_by_dx=f2e(x,A,B,C),
10     dB_by_dx=A-B,
11 endfunction
12 function dC_by_dx=f3e(x,A,B,C),
13     dC_by_dx=B,
14 endfunction
15 A=1,B=0,C=0
16 for x=.1:.1:10,
17     h=.1                                          //step
   increment of 0.1
18     k1=h*f1e(x,A,B,C)
19     l1=h*f2e(x,A,B,C)
20     m1=h*f3e(x,A,B,C)
21     k2=h*f1e(x+h/2,A+k1/2,B+l1/2,C+m1/2)
22     l2=h*f2e(x+h/2,A+k1/2,B+l1/2,C+m1/2)
23     m2=h*f3e(x+h/2,A+k1/2,B+l1/2,C+m1/2)
24     k3=h*f1e(x+h/2,A+k2/2,B+l2/2,C+m2/2)
25     l3=h*f2e(x+h/2,A+k2/2,B+l2/2,C+m2/2)
26     m3=h*f3e(x+h/2,A+k2/2,B+l2/2,C+m2/2)
27     k4=h*f1e(x+h,A+k3,B+l3,C+m3)
28     l4=h*f2e(x+h,A+k3,B+l3,C+m3)
29     m4=h*f3e(x+h,A+k3,B+l3,C+m3)
30     A=A+(k1+2*k2+2*k3+k4)/6
31     B=B+(l1+2*l2+2*l3+l4)/6
32     C=C+(m1+2*m2+2*m3+m4)/6
33     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");
34     end

```

```

35 end
36     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

```

1  clc
2  //given rxn A+B $\xrightarrow{k_1}$ C
3  //          B+C $\xrightarrow{k_2}$ D
4  rc_k1=1,rc_k2=1 //rate
   constants
5  disp("the solution of eg 4.17  $\rightarrow$ Plug Flow Reactor")
6  function dA_by_dx=f1a(x,A,B,C,D),
7      dA_by_dx=-A*B,
8  endfunction
9  function dB_by_dx=f2a(x,A,B,C,D),
10     dB_by_dx=-A*B-B*C,
11 endfunction
12 function dC_by_dx=f3a(x,A,B,C,D),
13     dC_by_dx=A*B-B*C,
14 endfunction
15 function dD_by_dx=f4a(x,A,B,C,D),
16     dD_by_dx=B*C,
17 endfunction
18 A=1,B=1,C=0,D=0
19 for x=.1:.1:10,
20     h=.1 //step
       increment of 0.1
21     k1=h*f1a(x,A,B,C,D)
22     l1=h*f2a(x,A,B,C,D)
23     m1=h*f3a(x,A,B,C,D)
24     n1=h*f4a(x,A,B,C,D)
25     k2=h*f1a(x+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
26     l2=h*f2a(x+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
27     m2=h*f3a(x+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)

```

```

28     n2=h*f4a(x+h/2,A+k1/2,B+l1/2,C+m1/2,D+n1/2)
29     k3=h*f1a(x+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
30     l3=h*f2a(x+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
31     m3=h*f3a(x+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
32     n3=h*f4a(x+h/2,A+k2/2,B+l2/2,C+m2/2,D+n2/2)
33     k4=h*f1a(x+h,A+k3,B+l3,C+m3,D+n3)
34     l4=h*f2a(x+h,A+k3,B+l3,C+m3,D+n3)
35     m4=h*f3a(x+h,A+k3,B+l3,C+m3,D+n3)
36     n4=h*f4a(x+h,A+k3,B+l3,C+m3,D+n3)
37     A=A+(k1+2*k2+2*k3+k4)/6
38     B=B+(l1+2*l2+2*l3+l4)/6
39     C=C+(m1+2*m2+2*m3+m4)/6
40     D=D+(n1+2*n2+2*n3+n4)/6
41     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");
42     end
43 end
44     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

```

1  clc
2  disp("the solution of eg 4.18-->Non- Isothermal Plug
        Flow Reactor")
3  T=294.15
4  //rxn A-->B
5  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
6  b=E/(R*T1),k1=5.64*10^13*exp(-b), A=%pi*dia^2/4,
        na0=mdot*1000/MW, Ts=388.71
7  k=k1*exp(b*(1-T1/T))
8
9  //dX_by_dV=ra/na0

```



```

10 //dX_by_dV=k*(1-X)/F
11 //from energX balance
12 //mdot*Cp*dT_by_dz+ra*A*H_RXN-q=0
13 //q=U*%pi*dia*(Ts-T)
14 //-mdot*Cp*dT_by_dV+4*U/dia*(Ts-T)-ra*H_rxn=0
15 F=mdot/rho
16 t1=A*k1/F
17
18 s1=mdot*Cp/A
19 s2=4*U/dia
20 s3=H_rxn*t1
21
22 function dX_by_dz=fg1(z,X,T),
23         dX_by_dz=t1*(1-X)*exp(b*(1-T1/T))
24 endfunction
25 function dT_by_dz=fd1(z,X,T),
26         ra=na0/A*(t1*(1-X)*exp(b*(1-T1/T)))
27         dT_by_dz=(ra*H_rxn-s2*(Ts-T))/-s1
28
29 endfunction
30
31 X=0,T=294.15
32 for z=0:.1:350,
33     h=.1 //szep
34         incremenz of 0.1
35     k1=h*fg1(z,X,T)
36     l1=h*fd1(z,X,T)
37     k2=h*fg1(z+h/2,X+k1/2,T+l1/2)
38     l2=h*fd1(z+h/2,X+k1/2,T+l1/2)
39     k3=h*fg1(z+h/2,X+k2/2,T+l2/2)
40     l3=h*fd1(z+h/2,X+k2/2,T+l2/2)
41     k4=h*fg1(z+h,X+k3,T+l3)
42     l4=h*fd1(z+h,X+k3,T+l3)
43     X=X+(k1+2*k2+2*k3+k4)/6
44     T=T+(l1+2*l2+2*l3+l4)/6
45 //condition for height calc. for 90% conversion
46 if X>.9 &X<.9005 then disp(z,"the height of the
    tower required for 90% conversion in mtrs is"

```

```
    ); break
46   end
47   end
```

Chapter 5

BOUNDARY VALUE PROBLEMS

Scilab code Exa 5.1 discretization in 1 D space

```
1 clc
2 //finite difference method
3 disp("the solution of eg 5.1-->Discretization in 1-D
      space");
4 //given  $d^2y_{by}dx^2 - 2=0$  hence it is dirichlet 's
      problem
5
6  $x_1=0$ ,  $y_1=0$  //initial boundary
      conditions
7  $x_3=1$ ,  $y_3=0$ 
8
9  $\Delta x=.5$  //since we have to find
       $y_2$  at  $x=.5$  so  $x_2=.5$ 
10 //in the central difference method substituting  $i=2$ 
      we have
11  $/(y_3 - 2*y_2 + y_1) / (\Delta x)^2 = 2$ 
12 //since  $y_1$  &  $y_3=0$  as per B.C.
13  $y_2 = (y_3 + y_1 - 2*\Delta x^2) / 2$ 
14 disp( $y_2$ , "the value of y at  $x=.5$  from finite
```

```

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

```

Scilab code Exa 5.2 discretization in 1 D space

```

1 clc
2 disp("the solution of eg 5.2 -->Discretization in 1-
    D space");
3 //boundary conditions are: x=0 at y=0; dy/dx=1 at x
    =1
4 disp("to solve this problem we will take delta x=.5
    since we have to find the value at x=.5");
5 delta_x=.5
6 y_1=0
7 //using central difference eqn
8 dy_by_dx=1 //at x=1, i=3
9 //y_4=dy/dx*2*delta_x+y_2 //since from B.C.
    at node 3
10
11 //y_2=delta_x^2+y_3-delta_x //on
    substituting the value of y_4
12 y_3=-(2*delta_x^2+2*(delta_x^2-delta_x)-y_1) //on
    substituting the value of y_2
13 y_2=delta_x^2+y_3-delta_x
14 disp(y_2,"the value of y at x=.5 is");

```

Scilab code Exa 5.3 discretization in 1 D space

```

1 clc

```

```

2 disp("the solution of eg 5.3 -->Discretization in 1-
   D space");
3 //given the source term f(x)=4x^2-2x-4
4 //given eqn d2y/dx2-2y=f(x)
5 y_1=0
6 y_4=-1
7 delta_x=1/3           //since given 3 parts and
   length=1
8 for i=0:3,j=0:delta_x:1;
9 end
10 //given to divide the line in 3 parts
11 //at node 2
12 //y_3-2*y_2
13 function d=fx3(x),
14 d=(4*x^2-2*x-4)
15 endfunction
16 f2=fx3(j(2))
17 f3=fx3(j(3))
18 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)
19 y_2=(f3+2*y_3)*delta_x^2+2*y_3-y_4
20 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

```

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

```

```

 9 //given to divide the line in 3 parts
10 function d=fx3(x),
11 d=(4*x^2-2*x-4)
12 endfunction
13 f2=fx3(j(2))
14 f3=fx3(j(3))
15 f4=fx3(j(4))
16 disp("the exact analytical soln are");
17 for i=1:3,y(i)=-2*j(i+1)^2+j(i+1),disp(y(i));
18 end
19 //using B.C.-2 at node 4 we get
20 //(y_5-y_3)/(2*delta_x)=-3
21 //eqn at node 2
22 //-20*y_2+9*y_3=f2
23 //at node 3
24 //9*y_2-20*y_3+9*y_4=f3
25 //at node 4
26 //18*y_3-20*y_4=16
27 disp(f4,f3,f2,"the value of f(x) at node 2 3 and 4
    are");
28 //now solving the equations using TDMA method
29
30 a(2)=9,a(3)=18 //sub diagonal assignment
31
32 for j=1:3, b(j)=-20; //main diagonal
    assignment
33 end
34 for k=1:2, c(k)=9; //super diagonal
    assignment
35 end
36 d(1)=f2 //given values
    assignment
37 d(2)=f3
38 d(3)=16
39
40 i=1;
41 n=3;
42 beta1(i)=b(i); //initial b is equal

```

```

    to beta since a1=0
43 gamma1(i)=d(i)/beta1(i);           //since c7=0
44 m=i+1;
45 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
46 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
47 end
48 x(n)=gamma1(n);                     //since c7=0
49 n1=n-i;
50 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
    j);
51 end
52 disp("the values of y2, y3, y4 by TDMA method are");
53 for i=1:3, disp(x(i));
54 end

```

Scilab code Exa 5.5 discretization in 1 D space

```

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

```

```

16 end
17 c(1)=2
18 for k=2:9, c(k)=1;           //super diagonal
    assignment
19 end
20 for l=1:9, d(l)=0;
21     end                       //given values
    assignment
22 d(10)=-1
23 i=1;
24 n=10;
25 beta1(i)=b(i);              //initial b is equal
    to beta since a1=0
26 gamma1(i)=d(i)/beta1(i);    //since c7=0
27 m=i+1;
28 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
29 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
30 end
31 x(n)=gamma1(n);             //since c7=0
32 n1=n-i;
33 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
    j);
34 end
35 disp("the values of y from y1 to y10 by TDMA method
    are");
36 for i=1:10, disp(x(i));
37 end

```

Scilab code Exa 5.6 1 D steady state heat conduction

```

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

```



```

5 delta_x=(T_10-T_1)/9           //since 9
   divisions are to be made
6 //using central difference method we get
7 //for node 2--> 2*T_2-T_3=100
8 for i=2:8, a(i)=-1             //sub diagonal
   assignment
9 end
10 for j=1:8, b(j)=2;            //main diagonal
   assignment
11 end
12 for k=1:7, c(k)=-1;          //super diagonal
   assignment
13 end
14 d(1)=100, d(8)=200
15 for l=2:7, d(l)=0;
16 end                           //given values assignment
17 i=1;
18 n=8;
19 beta1(i)=b(i);                //initial b is equal
   to beta since a1=0
20 gamma1(i)=d(i)/beta1(i);      //since c7=0
21 m=i+1;
22 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
23 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
24 end
25 x(n)=gamma1(n);               //since c7=0
26 n1=n-i;
27 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
   j);
28 end
29 disp("the values of T from T2 to T9 by TDMA method
   are");
30 for i=1:8, disp(x(i));
31 end

```

Scilab code Exa 5.7 1 D steady state heat conduction

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

```

30 d(1)=300
31 i=1;
32 n=4;
33 beta1(i)=b(i);           //initial b is equal
    to beta since a1=0
34 gamma1(i)=d(i)/beta1(i); //since c7=0
35 m=i+1;
36 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
37 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
38 end
39 x(n)=gamma1(n);         //since c7=0
40 n1=n-i;
41 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
    j);
42 end
43 disp("the values of T from T1 to T4 in Celsius by
    TDMA method are");
44 for i=1:4, disp(x(i)-T_surr);
45 end

```

Scilab code Exa 5.8 chemical reaction and diffusion in pore

```

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

```

```

        method
11  for i=2:99, a(i)=1           //sub diagonal
        assignment
12  end
13  a(100)=2                    //since C99=C100 using B.C
        .
14  for j=1:100, b(j)=-2.0001, //main diagonal
        assignment
15  end
16  for k=1:99, c(k)=1;        //super diagonal
        assignment
17  end
18  d(1)=-1
19  for l=2:100, d(l)=0;
20  end                        //given values assignment
21  i=1;
22  n=100;
23  beta1(i)=b(i);            //initial b is equal
        to beta since a1=0
24  gamma1(i)=d(i)/beta1(i); //since c7=0
25  m=i+1;
26  for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
27  gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
28  end
29  x(n)=gamma1(n);          //since c7=0
30  n1=n-i;
31  for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
        j);
32  end
33  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

```
1 //given convective diffusive eqn  $\rightarrow -u*(dc/dx)+D*($   
    $d^2C/dx^2)=0$   
2 C_ini=0 //at x=0  
3 C_end=1 //at x=1  
4 clc  
5 disp("the soln of eg 6.1");  
6  
7 //using central difference method for both diffusion  
   and convective term  
8  $-u*(C(i+1)-C(i-1))/(2*delta\_x) + D*(C(i+1)+C(i-1)$   
    $-2*C(i))/delta\_x^2 = 0$   
9 delta_x=1/50  
10 //on solving the given eqns and by using the given  
   boundary eqns we have  
11 Pe=50 //given  
12 Pe_local=50*delta_x //u/D=50 as l=1  
13 alpha=Pe_local-2 //co-eff of C(i  
   +1)  
14 Beta=Pe_local+2 //co-eff of C(i
```

```

-1)
15 //multiplying with  $-2*\delta_x^2/D$  we get
16 // $-(Pe_{local}+2)*C(i-1) + 4*C(i) + (Pe_{local}-2)*C(i$ 
+1)=0
17 //solving eqns using TDMA method
18 for i=2:49, a(i)=-Beta //sub diagonal
assignment
19 end
20 for j=1:49, b(j)=4, //main diagonal
assignment
21 end
22 for k=1:48, c(k)=alpha; //super diagonal
assignment
23 end
24 d(1)=Beta*C_ini, d(49)=-alpha*C_end
25 for l=2:48, d(l)=0;
26 end //given values assignment
27 i=1;
28 n=49;
29 beta1(i)=b(i); //initial b is equal
to beta since a1=0
30 gamma1(i)=d(i)/beta1(i); //since c7=0
31 m=i+1;
32 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
33 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
34 end
35 x(n)=gamma1(n); //since c7=0
36 n1=n-i;
37 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
j);
38 end
39 disp("the values of conc. using CDS method for x=.84
to 1 are")
40 for i=42:49, disp(x(i));
41 end
42 //part (ii) using CDS and UDS method
43 //multiplying with  $-\delta_x^2/D$  we get
44 // $-(Pe_{local}+1)*C(i-1) + (Pe_{local}+2)*C(i)-C(i+1)=0$ 

```

```

45 BEta=Pe_local+2
46 Gamma=Pe_local+1
47 for i=2:49, a(i)=-Gamma           //sub diagonal
    assignment
48 end
49 for j=1:49, b(j)=BEta,           //main diagonal
    assignment
50 end
51 for k=1:48, c(k)=-1;             //super diagonal
    assignment
52 end
53 d(1)=Gamma*C_ini, d(49)=C_end
54 for l=2:48, d(l)=0;
55 end                               //given values assignment
56 i=1;
57 n=49;
58 beta1(i)=b(i);                   //initial b is equal
    to beta since a1=0
59 gamma1(i)=d(i)/beta1(i);         //since c7=0
60 m=i+1;
61 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
62 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
63 end
64 x(n)=gamma1(n);                  //since c7=0
65 n1=n-i;
66 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
    j);
67 end
68 disp("the values of conc. using CDS/UDS method for x
    =.84 to 1 are")
69 for i=42:49, disp(x(i));
70 end
71 disp("the analytical soln is");
72 for i=42:49;
73     C_an(i)=C_ini+((exp(Pe*.02*i)-1)*(C_end-C_ini)
        /(exp(Pe)-1));
74     disp(C_an(i));
75     end

```

Scilab code Exa 6.2 upwind scheme

```
1 //given convective diffusive eqn  $\rightarrow -u*(dc/dx)+D*($   
     $d^2C/dx^2)=0$   
2 C_ini=0 //at x=0  
3 C_end=1 //at x=1  
4 clc  
5 disp("the soln of eg 6.2-->");  
6  
7 //using central difference method for both diffusion  
    and convective term  
8  $-u*(C(i+1)-C(i-1))/(2*\text{delta}_x) + D*(C(i+1)+C(i-1)$   
     $-2*C(i))/\text{delta}_x^2 = 0$   
9 delta_x=1/50  
10 //on solving the given eqns and by using the given  
    boundary eqns we have  
11 Pe=500 //given  
12 Pe_local=500*delta_x //u/D=50 as l=1  
13 alpha=Pe_local-2 //co-eff of C(i  
    +1)  
14 Beta=Pe_local+2 //co-eff of C(i  
    -1)  
15 //multiplying with  $-2*\text{delta}_x^2/D$  we get  
16  $-(\text{Pe\_local}+2)*C(i-1) + 4*C(i) + (\text{Pe\_local}-2)*C(i$   
     $+1)=0$   
17 //solving eqns using TDMA method  
18 for i=2:49, a(i)=-Beta //sub diagonal  
    assignment  
19 end  
20 for j=1:49, b(j)=4, //main diagonal  
    assignment  
21 end  
22 for k=1:48, c(k)=alpha; //super diagonal  
    assignment
```



```

23 end
24 d(1)=Beta*C_ini , d(49)=-alpha*C_end
25 for l=2:48, d(l)=0;
26 end //given values assignment
27 i=1;
28 n=49;
29 beta1(i)=b(i); //initial b is equal
    to beta since a1=0
30 gamma1(i)=d(i)/beta1(i); //since c7=0
31 m=i+1;
32 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
33 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
34 end
35 x(n)=gamma1(n); //since c7=0
36 n1=n-i;
37 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
    j);
38 end
39 disp("the values of conc. using CDS method for x=.84
    to 1 are")
40 for i=42:49, disp(x(i));
41 end
42 //part (ii) using CDS and UDS method
43 //multiplying with  $-\Delta x^2/D$  we get
44  $-(Pe_{local}+1)*C(i-1) + (Pe_{local}+2)*C(i)-C(i+1)=0$ 
45 BEta=Pe_local+2
46 Gamma=Pe_local+1
47 for i=2:49, a(i)=-Gamma //sub diagonal
    assignment
48 end
49 for j=1:49, b(j)=BEta, //main diagonal
    assignment
50 end
51 for k=1:48, c(k)=-1; //super diagonal
    assignment
52 end
53 d(1)=Gamma*C_ini , d(49)=C_end
54 for l=2:48, d(l)=0;

```

```

55 end //given values assignment
56 i=1;
57 n=49;
58 beta1(i)=b(i); //initial b is equal
    to beta since a1=0
59 gamma1(i)=d(i)/beta1(i); //since c7=0
60 m=i+1;
61 for j=m:n, beta1(j)=b(j)-a(j)*c(j-1)/beta1(j-1);
62 gamma1(j)=(d(j)-a(j)*gamma1(j-1))/beta1(j);
63 end
64 x(n)=gamma1(n); //since c7=0
65 n1=n-i;
66 for k=1:n1, j=n-k; x(j)=gamma1(j)-c(j)*x(j+1)/beta1(
    j);
67 end
68 disp("the values of conc. using CDS/UDS method for x
    =.84 to 1 are")
69 for i=42:49, disp(x(i));
70 end
71 disp("the analytical soln is");
72 for i=42:49;
73     C_an(i)=C_ini+((exp(Pe*.02*i)-1)*(C_end-C_ini)
        /(exp(Pe)-1));
74     disp(C_an(i));
75     end

```

Chapter 7

TUBULAR REACTOR WITH AXIAL DISPERSION

Scilab code Exa 7.1 boundary value problem in chemical reaction engineering

```
1 clc
2 disp("soln of eg 7.1-->First Order Reaction-50 parts
      ")
3 e1=1,e2=1

      //assumed values
4 u=1,D=10^-4,k=1,C_a_in=1,delta_x=10/50
      //given data
5 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
6 cf_ca2_n1=2*D/delta_x^2+u/delta_x
7 cf_da1_n1=-(2*u^2/D+2*u/delta_x)*C_a_in
      //right hand side co-
      efficient
8 cf_ca1_n2=D/delta_x^2+u/delta_x
9 cf_ca2_n2=-2*D/delta_x^2-u/delta_x-k
10 cf_ca3_n2=D/delta_x^2
11 cf_da1_n2=0
```

```

12 cf_ca2_n3=cf_ca1_n2
13 cf_ca3_n3=cf_ca2_n2
14 cf_ca4_n3=cf_ca3_n2
15 cf_da1_n3=0
16 cf_ca50_n51=2*D/delta_x^2+u/delta_x
                                     //co-efficient of C-A50
    at node 51
17 cf_ca51_n51=-2*D/delta_x^2-u/delta_x-k
18 cf_da51_n51=0
19 for i=2:50, a(i)=cf_ca1_n2,
20 end
21 a(51)=cf_ca2_n1, c(1)=cf_ca2_n1
22 for i=2:50, c(i)=cf_ca3_n2,
23 end
24 d(1)=cf_da1_n1
25 for i=2:51, d(i)=cf_da1_n2
26 end
27 for i=1:51, x(i)=0,
28 end
29 b(1)=cf_ca1_n1,
30 for i=2:51, b(i)=cf_ca2_n2, end
31 while e1>1e-6 & e2>1e-6 do for i=1:51, x1(i)=x(i), end
    ,
32 i=1, n=51, Beta(i)=b(i),
33   Gamma(i)=d(i)/Beta(i)
34   i1=i+1,
35   for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
36     Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
37   end
38   x(n)=Gamma(n)
39   n1=n-i,
40   for k=1:n1, j=n-k, x(j)=Gamma(j)-c(j)*x(j+1)/Beta
      (j)
41   end
42 e1=abs(x(42)-x1(42)), e2=abs(x(18)-x1(18))
43 end
44 for i=1:51, disp(x(i))
45 end

```

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

```
1 clc
2 disp("soln of eg 7.2-->Second Order Reaction-20
      parts")
3 e1=1,e2=1
4 u=1,D=10^-4,k=1,C_a_in=1,delta_x=10/20
5 cff_ca2_n1=2*D/delta_x^2+u/delta_x
      //co-efficient of C-A2 at
      node 1
6 cff_da1_n1=-(2*u^2/D+2*u/delta_x)*C_a_in
      //right hand side co-efficient
7 cff_ca1_n2=D/delta_x^2+u/delta_x
8 cff_ca3_n2=D/delta_x^2
      //co-efficient
      of C-A3 at node 2
9 cff_da1_n2=0
10 cff_ca2_n3=cf_ca1_n2
11 cff_ca4_n3=cf_ca3_n2
12 cff_da1_n3=0
13 cff_ca20_n21=2*D/delta_x^2+u/delta_x
      //co-efficient of C-A20 at
      node 21
14 cff_da21_n21=0
15 for i=2:20, a(i)=cff_ca1_n2,
16 end
17 a(21)=cff_ca2_n1,c(1)=cff_ca2_n1
18 for i=2:20,c(i)=cff_ca3_n2,
19 end
20 d(1)=cff_da1_n1
21 for i=2:21,d(i)=cff_da1_n2
22 end
23 for i=1:21,x(i)=0,
```

```

24 end
25 while e1>1e-6 & e2>1e-6 do for i=1:21,x1(i)=x(i),end
    ,
26   cff_ca1_n1=-2*D/delta_x^2-3*u/delta_x-x1(1)-2*u
      ^2/D //main diagonal elements
      dependence on conc.
27   b(1)=cff_ca1_n1 ,
28   for i=2:21,b(i)=-2*D/delta_x^2-u/delta_x-x(i),end
29
30 //solving by TDMA method
31   i=1, n=21, Beta(i)=b(i),
32   Gamma(i)=d(i)/Beta(i)
33   i1=i+1,
34   for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
35     Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
36   end
37   x(n)=Gamma(n)
38   n1=n-i,
39   for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
      (j)
40   end
41 e1=abs(x(1)-x1(1)), e2=abs(x(21)-x1(21))
42 end
43 for i=1:21,disp(x(i))
44 end

```

Chapter 8

CHEMICAL REACTION AND DIFFUSION IN SPHERICAL CATALYST PELLET

Scilab code Exa 8.1 first order reaction

```
1 clc
2 disp("the soln of eg 8.1-->");
3 for i=1:100, x(i)=0
4 end
5 iter=0, e1=1, f=1
6 while e1>1e-6 & f>1e-6 do iter=iter+1, for i=1:100,
    x1(i)=x(i),
7     end, for i=2:100, a(i)=1-(1/(i-1))
8     end, b(1)=-6.01, for i=2:100, b(i)=-2.01
9     end, c(1)=6, for i=2:99, c(i)=1+(1/(i-1))
10    end, for i=1:99, d(i)=0, end, d(100)=-100/99,
11    i=1, n=100, Beta(i)=b(i),
12    Gamma(i)=d(i)/Beta(i)
13    i1=i+1,
14    for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
15        Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
16    end
```

```

17     x(n)=Gamma(n)
18     n1=n-i,
19     for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
        (j)
20     end
21     e1=abs(x(1)-x1(1)),
22     f=abs(x(100)-x1(100)),
23 end
24 disp(iter,"the solution by TDMA of node 77 to 99 by
    1st order rxn. is");
25 for i=78:100,
26     disp(x(i));
27 end

```

Scilab code Exa 8.2 second order reaction

```

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

```



```

18     x(n)=Gamma(n)
19     n1=n-i,
20     for k=1:n1, j=n-k, x(j)=Gamma(j)-c(j)*x(j+1)/Beta
        (j)
21     end
22     e1=abs(x(1)-x1(1)),
23     f=abs(x(100)-x1(100)),
24 end
25 disp("the solution by TDMA of node 77 to 99 by 2nd
        order rxn. is");
26 for i=77:100,
27     disp(x(i));
28 end

```

Scilab code Exa 8.3 non isothermal condition

```

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

```

```

14     Gamma(i)=d(i)/Beta(i)
15     i1=i+1,
16     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
17         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
18     end
19     x(n)=Gamma(n)
20     n1=n-i,
21     for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
        (j)
22     end
23     e2=abs(x(1)-x1(1)),
24     f1=abs(x(100)-x1(100)),
25 end
26 disp("the solution by TDMA of node 77 to 100 by 1st
        order rxn. is");
27 for i=76:100,
28     disp(x(i));
29 end

```

Scilab code Exa 8.4 non isothermal condition

```

1  clc
2  disp("the soln of eg 8.4-->");
3  for i=1:100, x(i)=0
4  end
5  iter=0, e1=1, f1=1
6  while e1>1e-6 & f1>1e-6 do iter=iter+1,for i=1:100,
        x1(i)=x(i),
7      end, for i=2:100, a(i)=1-(1/(i-1))
8      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)))
9      end, c(1)=6, for i=2:99,c(i)=1+(1/(i-1))
10     end, for i=1:99,d(i)=0, end, d(100)=-100/99,
11     i=1, n=100, Beta(i)=b(i),

```

```

12     Gamma(i)=d(i)/Beta(i)
13     i1=i+1,
14     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
15         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
16     end
17     x(n)=Gamma(n)
18     n1=n-i,
19     for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
20         (j)
21     end
22     e1=abs(x(1)-x1(1)),
23     f1=abs(x(100)-x1(100)),
24 end
25 disp("the solution by TDMA of node 77 to 99 by 1st
26     order rxn. is");
27 for i=76:100,
28     disp(x(i));
29 end

```

Chapter 9

ONE DIMENSIONAL TRANSIENT HEAT CONDUCTION

Scilab code Exa 9.1 transient conduction in rectangular slab

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

```

12 end
13 for i=1:4, x(i)=20 //
    initial temperature
14 end
15 for t=0.1:.1:3.1, for i=1:4, y(i)=x(i), end
    //TDMA method
16 d(1)=-.5*y(1),
17 d(2)=-y(2)
18 d(3)=-y(3)
19 d(4)=-y(4)-300
20 i=1, n=4
21 Beta(i)=b(i),
22 Gamma(i)=d(i)/Beta(i)
23 i1=i+1,
24 for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
25 Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
26 end
27 x(n)=Gamma(n)
28 n1=n-i,
29 for k=1:n1, j=n-k, x(j)=Gamma(j)-c(j)*x(j+1)/Beta
    (j)
30 end
31 for i=1:4, disp(x(i)); //solution of
    temperature
32 end
33 disp("-----")
34 end
35 disp("-----END-----");

```

Scilab code Exa 9.2 transient conduction in rectangular slab

```

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

```

```

3 delta_t=1, delta_x=.05, alpha=10^-5
4 t1=alpha*delta_t/delta_x^2
5 for i=2:9, a(i)=-t1
6 end
7 for i=1:9, b(i)=1+2*t1
8 end
9 for i=1:8, c(i)=-t1
10 end
11 t=1,tf=3000
12 for i=1:9, x(i)=300
13 end
14 e1=425,
15 disp("time when centre temp is 425 K in secs. is")
16 for t=1:1:tf, for i=1:9, y(i)=x(i), end
17     d(1)=y(1)+1.7,
18     d(9)=y(9)+2.4,
19     for i=2:8, d(i)=y(i)
20     end
21     i=1, n=9
22     Beta(i)=b(i),
23     Gamma(i)=d(i)/Beta(i)
24     i1=i+1,
25     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
26         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
27     end
28     x(n)=Gamma(n)
29     n1=n-i,
30     for k=1:n1, j=n-k, x(j)=Gamma(j)-c(j)*x(j+1)/Beta
31         (j)
32     end
33     if abs(x(5)-e1)>0 & abs(x(5)-e1)<.1 then disp(t)
34         ; break;
35     end
36 disp("the values of temp. at req. time is");
37 for i=1:9, disp(x(i)); end

```

Scilab code Exa 9.3 transient conduction in cylinder

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

```

                (j)
27     end
28     disp(x(4),x(3),x(2),x(1));
29     disp("-----");
30     end

```

Scilab code Exa 9.4 transient conduction in sphere

```

1  clc
2  disp("the soln of eg 9.4-->Transient conduction in
      Sphere");
3  delta_t=.1, delta_r=.001, alpha=10^-5
4  t1=alpha*delta_t/delta_r^2
5  a(2)=0,a(3)=.5,a(4)=.667
6  b(1)=-7*t1
7  for i=2:4,b(i)=-3
8  end
9  c(1)=6,c(2)=2,c(3)=1.5
10 for i=1:4, x(i)=20
11 end
12 disp("T1,T2,T2 & T4 at time interval of .1 sec is")
13 for t=.1:.1:1.4,for i=1:4, y(i)=x(i),end
14     d(4)=-y(4)-400,
15     for i=1:3,d(i)=-y(i)
16     end
17     i=1, n=4
18     Beta(i)=b(i),
19     Gamma(i)=d(i)/Beta(i)
20     i1=i+1,
21     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
22         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
23     end
24     x(n)=Gamma(n)
25     n1=n-i,
26     for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta

```



```

                (j)
27     end
28     disp(x(4),x(3),x(2),x(1));
29     disp("-----");
30     end

```

Scilab code Exa 9.5 transient diffusion in sphere

```

1  clc
2  disp("the soln of eg 9.5-->");
3  R=.326, D=3*10^-7
4  delta_t=1, delta_r=.0326, conc_ini=10/(1.33*%pi*R^3)
5  t1=D*delta_t/delta_r^2
6  disp(conc_ini,"the initial conc. of drug is");
7  for i=2:10, a(i)=-((1-1/(i-1)))
8  end
9  b(1)=591.4
10 for i=2:10, b(i)=3544.5
11 end
12 c(1)=-1, for i=2:9, c(i)=-((1+1/(i-1)))
13 end
14 for i=1:10, x(i)=conc_ini
15 end
16 disp("Conc. at centre at t=3hr, 12 hr, 24 hr,48 hr
      is");
17 for t=1:delta_t:172800, for i=1:10, y(i)=x(i), end
18     d(1)=y(1)*590.4,
19     for i=2:10, d(i)=3542.5*y(i)
20     end
21     i=1, n=10
22     Beta(i)=b(i),
23     Gamma(i)=d(i)/Beta(i)
24     i1=i+1,
25     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
26         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),

```

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

Chapter 10

TWO DIMENSIONAL STEADY AND TRANSIENT HEAT CONDUCTION

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

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

```

12     tnew(4)=(told(5)+tnew(1)+told(7)+20)/4
13     tnew(5)=(tnew(2)+told(8)+told(6)+tnew(4))/4
14     tnew(6)=(tnew(3)+tnew(5)+told(9)+400)/4
15     tnew(7)=(tnew(4)+told(8)+40)/4
16     tnew(8)=(tnew(5)+tnew(7)+told(9)+20)/4
17     tnew(9)=(tnew(6)+420+tnew(8))/4
18     for i=1:9,e(i)=abs(tnew(i)-told(i))
19     end
20 end
21 disp("the values of T from 1st element to last is");
22 for i=1:9,disp(tnew(i));
23 end

```

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

```

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

```

Scilab code Exa 10.3 discretization in 2 D space

```
1 clc
2 disp("the soln of eg 10.3-->Red-Black Gauss-Seidel
      Method");
3 for j=1:5,tn(j,1)=400,end //
      defining conditions
4 for j=2:4,tn(1,j)=20,tn(5,j)=20,tn(j,5)=20, end
5 for i=1:9,e(i)=1
6 end
7 for i=2:4,j=2:4,tn(i,j)=60
8 end
9 t1=1e-6
10 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
11 //using red-black gauss-seidel method
12 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
13 //now getting the absolute difference of the 9
      variables
14 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)),
15 end
16 //now defining positions of the various nodes
      according to red black combination
17 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)
18 disp(R5,R4,R3,R2,R1,"the value of RED points
      respectively is");
19 disp(B4,B3,B2,B1,"the value of BLUE points
```

respectively is");

Scilab code Exa 10.8 discretization in 2 D space

```
1  clc
2  disp("the soln of eg 10.8-->Gauss Seidel Method");
3  for i=1:9,tnew(i)=101,e(i)=1 //
   assumed values
4  end
5  t=1e-6
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
7     for i=1:9, told(i)=tnew(i),end
8     //using eqn 10.10 for the interior nodes and
   convective boundary conditions for corner
   nodes
9     tnew(1)=(told(2)+50+.5*told(4)+100/3)*3/7
10    tnew(2)=(tnew(1)+told(3)+told(5)+100)/4
11    tnew(3)=(tnew(2)+told(6)+600)/4
12    tnew(4)=(told(5)+.5*tnew(1)+.5*told(7)+100/3)
   *3/7
13    tnew(5)=(tnew(2)+told(8)+told(6)+tnew(4))/4
14    tnew(6)=(tnew(3)+tnew(5)+told(9)+500)/4
15    tnew(7)=(.5*tnew(4)+.5*told(8)+100/3)*3/4
16    tnew(8)=(tnew(5)+.5*tnew(7)+.5*told(9)+100/3)
   *3/7
17    tnew(9)=(tnew(6)+100/3+.5*tnew(8)+250)*3/7
18    for i=1:9,e(i)=abs(tnew(i)-told(i))
19    end
20  end
21  disp("the values of T from 1st element to last is");
22  for i=1:9,disp(tnew(i));
23  end
```

Scilab code Exa 10.9 discretization in 2 D space

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

Scilab code Exa 10.10 ADI method

```
1  clc
2  disp("the soln of eg 10.10-->Alternating Direction
      Implicit Method");
3  nmax=25
4  a(2)=1, a(3)=1, //
      defining variables
5  b(1)=-4, b(2)=-4, b(3)=-4
6  c(1)=1, c(2)=1
7  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
8  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
9  for i=2:4, for j=2:4 t(i,j)=20
10     end
11 end
12 //solving by TDMA Method
13 for nn=1:nmax, for jj=2:4, d(1)=-t(1,jj)-t(2,jj+1)-
      tstar(2,jj-1),
14     d(2)=-t(3,jj+1)-tstar(3,jj-1),
15     d(3)=-t(5,jj)-t(4,jj+1)-tstar(4,jj-1)
16     i=1, n=3
17     Beta(i)=b(i),
18     Gamma(i)=d(i)/Beta(i)
19     i1=i+1,
20     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
21     Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
22     end
23     x(n)=Gamma(n)
24     n1=n-i,
```



```

25     for k=1:n1, j=n-k, x(j)=Gamma(j)-c(j)*x(j+1)/Beta
        (j)
26     end
27     tstar(2,jj)=x(1)
28     tstar(3,jj)=x(2)
29     tstar(4,jj)=x(3)
30 end
31 for ii=2:4, d(1)=-t(ii,1)-tstar(ii+1,2)-tstar(ii-1,2)
    ,
32     d(2)=-tstar(ii+1,3)-tstar(ii-1,3)
33     d(3)=-t(ii,5)-tstar(ii+1,4)-tstar(ii-1,4)
34     i=1, n=3
35     Beta(i)=b(i),
36     Gamma(i)=d(i)/Beta(i)
37     i1=i+1,
38     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
39         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
40     end
41     x(n)=Gamma(n)
42     n1=n-i,
43     for k=1:n1, j=n-k, x(j)=Gamma(j)-c(j)*x(j+1)/Beta
        (j)
44     end
45     t(ii,2)=x(1)
46 t(ii,3)=x(2)
47 t(ii,4)=x(3)
48 end
49 end
50 disp("the soln by ADI method is");
51 disp(t(2,4),t(2,3),t(2,2));
52 disp("-----");
53 disp(t(3,4),t(3,3),t(3,2));
54 disp("-----");
55 disp(t(4,4),t(4,3),t(4,2));

```

Scilab code Exa 10.11 ADI method for transient heat conduction

```

1  clc
2  disp("the soln of eg 10.11-->");
3  for k=2:10, a(k)=-1, b(k)=2.4, c(k)=-1
4  end
5  alpha=1,delta_t=.05,delta_x=.1
6  m=alpha*delta_t/delta_x^2
7  b(1)=2.4
8  c(1)=-2
9  for k=1:11, t(11,k)=400, tstar(11,k)=400, t(k,11)
      =400,tstar(k,11)=400
10 end
11 for i=1:10, for j=1:10, t(i,j)=0, tstar(i,j)=0
12     end
13 end
14 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,
15     i=1, n=10
16     Beta(i)=b(i),
17     Gamma(i)=d(i)/Beta(i)
18     i1=i+1,
19     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
20         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
21     end
22     x(n)=Gamma(n)
23     n1=n-i,
24     for k=1:n1, j=n-k,x(j)=Gamma(j)-c(j)*x(j+1)/Beta
        (j)
25     end, for count=1:10, tstar(count,jj)=x(count),
        end
26 end
27 for ii=1:10,
28     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
29     i=1, n=10
30     Beta(i)=b(i),
31     Gamma(i)=d(i)/Beta(i)
32     i1=i+1,
33     for j=i1:n, Beta(j)=b(j)-a(j)*c(j-1)/Beta(j-1),
34         Gamma(j)=(d(j)-a(j)*Gamma(j-1))/Beta(j),
35     end
36     x(n)=Gamma(n)
37     n1=n-i,
38     for k=1:n1, j=n-k, x(j)=Gamma(j)-c(j)*x(j+1)/Beta
        (j)
39     end,
40     for count=1:10, t(ii,count)=x(count), end
41 end
42 end
43 disp("the soln by ADI is");
44 for i=1:10, j=1:10, disp(t(i,j));
45 end
46 disp("the table is actually interchanged row &
        column.. horizontally are the elements of the
        column");

```
