Chapter 1

Incomplete LU decomposition

Question-1:

Code for Incomplete LU decomposition (ILU):

Consider 2D conduction problem governed by equation:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$$

Take a square domain of width and height of 1 unit. The boundary conditions are:

At x = 0: T = 0

At x = 1: T = 0

At y = 0: T = 0

At y = 1: T = 1

Discretise the governing equation and solve the obtained system of linear equations by writing code for Incomplete LU decomposition (ILU). This problem has an analytical solution which is given by series solution given by

$$T(x,y) = \frac{2}{\pi} \sum_{n=1}^{\infty} \frac{-1^{n+1}+1}{n} \sin n\pi x \frac{\sinh n\pi y}{\sinh n\pi} \qquad \text{Eq. (1)}$$

In your code you also write a subroutine to calculate temperature variation using the above analytical expression given by Eq. (1)

Your results should contain at least the following:

(a) Compare the temperature contours obtained by the code and with those obtained by using the above analytical expression (show the temperature contours figures side by side).

(b) Compare the temperature variation with y along mid-vertical plane x = 0.5 from the code and the above analytical expression. Plot both temperature profiles in the same figure.

(c) Compare the temperature variation with y along mid-horizontal plane y = 0.5 from the code and the above analytical expression. Plot both temperature profiles in the same figure.

1.1 Physical System

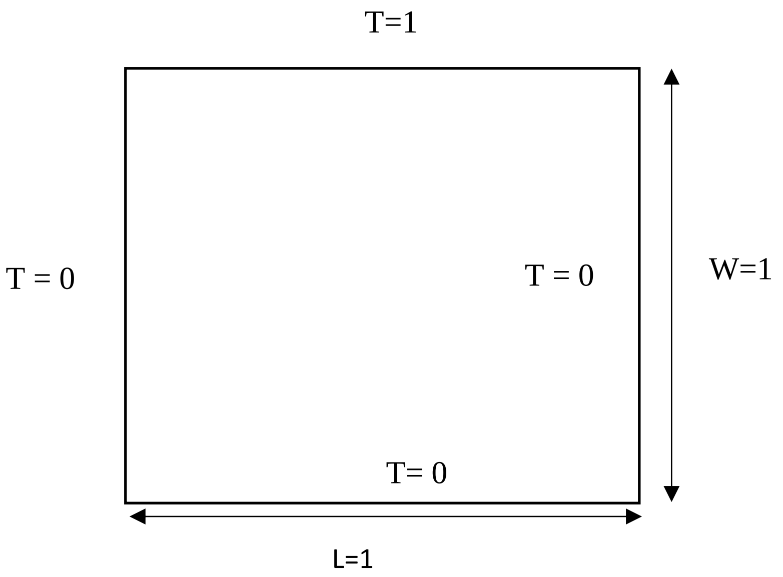
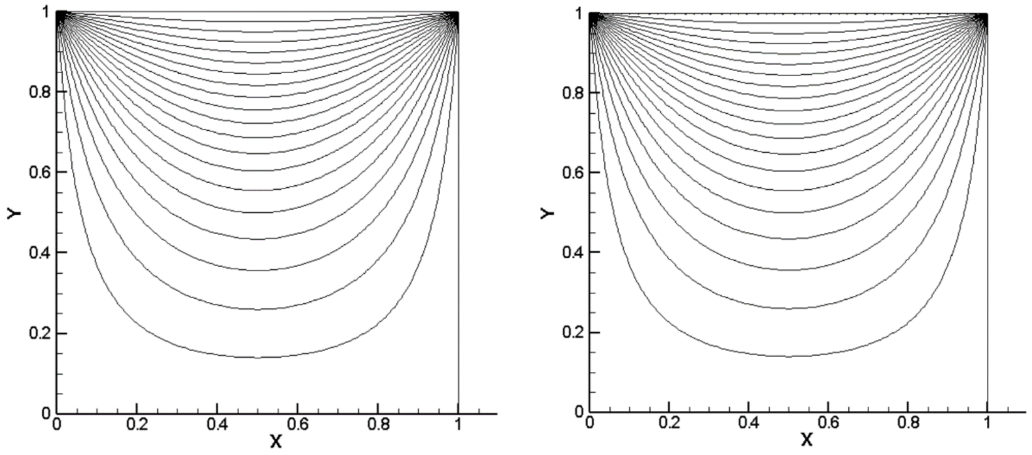


Figure 1.1: The physical system



(a) ILU decomposition result

(b) Analytical result

Figure 1.2: Temperature contours

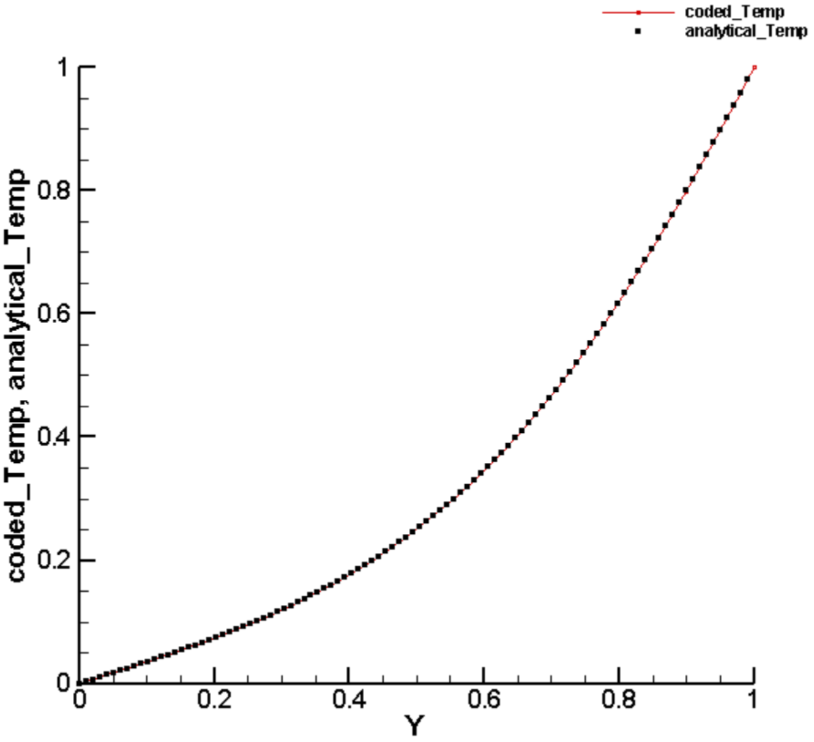


Figure 1.3: Temperature variation with y along mid-vertical plane x = 0.5

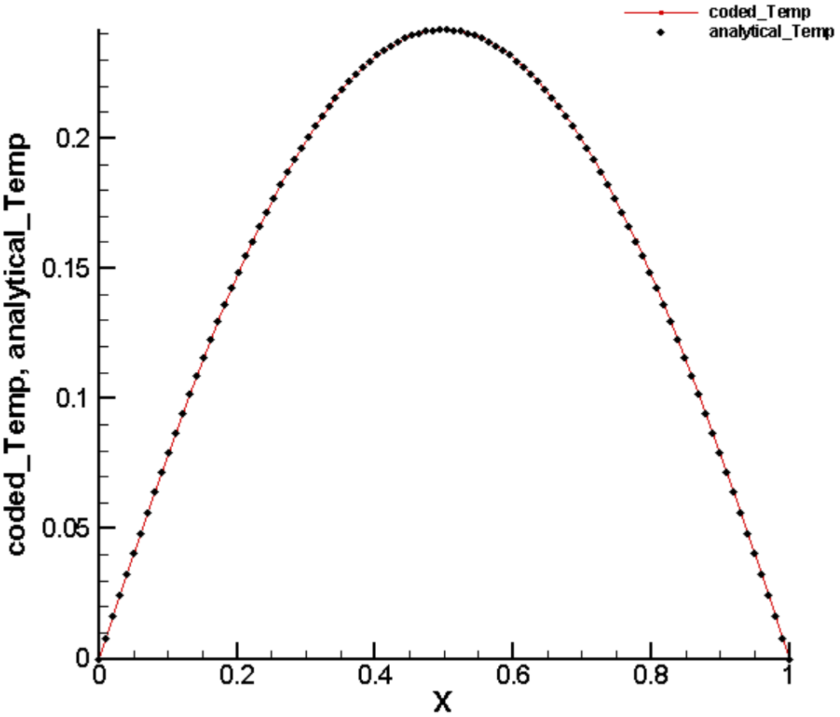


Figure 1.4: Temperature variation with x along mid-horizontal plane y = 0.5

1.2 The code of ILU Decomposition

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#define nx 100
#define ny 100
#define l 1
#define W 1
#define pi 3.14159265
double
Tnew[nx*ny],Told[nx*ny],beta,b[nx*ny],Mp[nx*ny],Mnw[nx*ny],Mw[nx*ny],Mn[nx*ny],Me[nx*ny],Ms[nx*ny],Mse[nx*ny];
double Lw[nx*ny],Lp[nx*ny],Ls[nx*ny],Un[nx*ny],Ue[nx*ny],N,dx,dy;
double ILU();
double output_files();
double analytic();
double initializtion();
double solver();
int main()
{
    initializtion();
    ILU();
    solver();
    analytic();
    output_files();
    return 0;
}
double initializtion()
{
    int i,j,L,N;N=nx*ny;
    for(j=1;j<=ny;j++)
    {
        for(i=1;i<=nx;i++)
        {
            L=(i-1)*ny+j;
            Tnew[L]=0.0;
            Told[L]=0.0;
            b[L]=0.0;
        }

    }
}
double ILU()
{
    int i,j,L;N=nx*ny;
    int Nj;
    Nj=ny;dx=l/(double)(nx-1);
    dy=W/(double)(ny-1);
    beta=dx/dy;
    for(j=1;j<=ny;j++)
    {
        for(i=1;i<=nx;i++)
        {
            L=(i-1)*ny+j;
            if(i==1)
            {
                Mp[L]=1.0;
                Mw[L]=0.0;
                Mn[L]=0.0;
                Ms[L]=0.0;
                Me[L]=0.0;
                b[L]=0.0;
            }
            else if((i==nx)||(j==1))
            {
                Mp[L]=1.0;
                Mw[L]=0.0;
                Mn[L]=0.0;
                Ms[L]=0.0;
                Me[L]=0.0;
                b[L]=0.0;
            }
            else if(j==ny)
            {
                Mp[L]=1.0;
                Mw[L]=0.0;
                Mn[L]=0.0;
                Ms[L]=0.0;
                Me[L]=0.0;
                b[L]=1.0;
            }
            else
            {
                Mp[L]=-2.0*(1+beta*beta);
                Mw[L]=beta*beta;
                Mn[L]=1.0;
                Ms[L]=1.0;
                Me[L]=beta*beta;
                b[L]=0.0;
            }
        }

    }

    for(L=1;L<=N;L++)
    {

        Lw[L]=Mw[L];

        Ls[L]=Ms[L];
        Lp[L]=Mp[L];
        if((L-1)>=1)
        {
            Lp[L]=Lp[L]-Ls[L]*Un[L-1];
        }

        if((L-Nj)>=1)
        {
            Lp[L]=Lp[L]-Lw[L]*Ue[L-Nj];
        }
        Un[L]=Mn[L]/Lp[L];

        Ue[L]=Me[L]/Lp[L];
    }
    for(L=1;L<=(N);L++)
    {
        Mnw[L]=0.0;
        if((L-Nj)>=1)
```

```
        {
            Mnw[L]=Lw[L]*Un[L-Nj];
        }

    }
    for(L=1;L<=(N);L++)
    {
        Mse[L]=0.0;
        if((L-1)>=1)
        {
        Mse[L]=Ls[L]*Ue[L-1];
        }
    }
}
double output_files()
{
    FILE *oup,*output2,*output3;double dx,dy,x,y;dx=l/(double)(nx-
1);
    dy=l/(double)(ny-1);
    oup=fopen("ILU.dat","w");
    output2=fopen("ILU_vertical.dat","w");
    output3=fopen("ILU_horizontal.dat","w");
    int i,j,L;
    fprintf( oup,
"VARIABLES=\"X\",\"Y\",\"coded_Temp\",\"analytical_Temp\"\n");
    fprintf(oup,"ZONE F=POINT\n");
    fprintf( oup, "I=%d, J=%d\t\n", nx, ny );
    for(j=1;j<=ny;j++)
    {
        for(i=1;i<=nx;i++)
        {
            x=(i-1)*dx;
            y=(j-1)*dy;
            L=(i-1)*ny+j;

    fprintf(oup,"%lf\t%lf\t%lf\t%lf\n",x,y,Tnew[L],Told[L]);
        }

    }
    int x1, x2,y1, y2,L1,L2;double avg1,avg2,avg,ana;
    x1=nx/2;
        x2=(nx/2)-1;
        fprintf( output2,
"VARIABLES=\"Y\",\"coded_Temp\",\"analytical_Temp\"\n");
        for(j=1;j<=ny;j++)
        {
                L1=(x1-1)*ny+j;
                avg1=Tnew[L1];
                L2=(x2-1)*ny+j;
                avg2=Tnew[L2];
                ana=(Told[L1]+Told[L2])/2;
                avg=(avg1+avg2)/2;
                y=(j-1)*dy;
                fprintf(output2,"%5.8f\t%5.8f\t%5.8f\n",y,avg,ana);
        }
        y1=ny/2;
        y2=(ny/2)-1;
        fprintf( output3,
"VARIABLES=\"X\",\"coded_Temp\",\"analytical_Temp\"\n");
        for(i=1;i<=nx;i++)
        {
                L1=(i-1)*ny+y1;
                avg1=Tnew[L1];
                L2=(i-1)*ny+y2;
                avg2=Tnew[L2];
                ana=(Told[L1]+Told[L2])/2;
                avg=(avg1+avg2)/2;
                x=(i-1)*dx;
                fprintf(output3,"%5.8f\t%5.8f\t%5.8f\n",x,avg,ana);
        }
        fclose(oup);

}
double solver()
{
        double Y[nx*ny],e,en,ed;
        int N,i,j,L;
        N=nx*ny;
        e=1.0;
        while(e>1.0e-14)
```

```
{

        Y[1]=(b[1]+Mse[1]*Told[1+(ny-1)])/Lp[1];
        for(L=2;L<=N;L++)
        {
                Y[L]=b[L]-Ls[L]*Y[L-1];
                if((L-(ny-1))>=1)
                {
                        Y[L]=Y[L]+Mnw[L]*Told[L-(ny-1)];
                }
                if((L+(ny-1))<=N)
                {
                        Y[L]=Y[L]+Mse[L]*Told[L+(ny-1)];
                }
                if((L-ny)>=1)
                {
                        Y[L]=Y[L]-Lw[L]*Y[L-ny];
                }
                Y[L]=Y[L]/Lp[L];
        }
        Tnew[N]=Y[N];
        for(L=N-1;L>=1;L--)
        {
                Tnew[L]=Y[L]-Un[L]*Tnew[L+1];
                if((L+ny)<=N)
                {
                        Tnew[L]=Tnew[L]-Ue[L]*Tnew[L+ny];
                }

        }
        en=0.0;
        ed=0.0;
        for(i=1;i<=N;i++)
        {
                en+=fabs(Tnew[i]-Told[i]);
                ed+=fabs(Tnew[i]);
        }
        e=en/ed;
        for(i=1;i<=N;i++)
        {
            Told[i]=Tnew[i];
            Y[i]=0.0;
        }printf("%lf\n",e);
    }

}
double analytic()
{
int i,j,k,L;double dx,dy,x,y,a[nx*ny],sum[nx*ny],e,term;
    dx=l/(double)(nx-1);
    dy=l/(double)(ny-1);
    for(j=1;j<=ny;j++)
    {
        for(i=1;i<=nx;i++)
        {
            L=(i-1)*ny+j;
            x=dx*(i-1);
            y=dy*(j-1);
            e=1.0;
            sum[L]=0.0;k=1;
            for(k=1;k<=110;k++)
            {
                sum[L]+=((pow(-
1,k+1)+1)/k)*(sin(k*pi*x))*sinh(k*pi*y)/(sinh(k*pi))*2/pi;
            }
            Told[L]=sum[L];
        }
    }
}
```

Chapter 2

Strongly Implicit Procedure

Question-2:

Code for Strongly Implicit procedure (SIP):
Consider 2D conduction governed by equation:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$$

Take a domain of width L and height W. The boundary conditions are:
At x = 0: T = 0
At x = L: T = 0
At y = 0: T = 0
At y = W: T = $T_m \sin\frac{\pi x}{L}$ Take L = 2 and W = 2.
Take T_m= 2. Discretise the governing and solve the obtained system of linear equations by writing code for Strongly Implicit procedure (SIP). This problem has an analytical solution which is given by expression

$$\theta(x, y) = T_m \frac{\sinh\frac{\pi y}{L}}{\sinh\frac{\pi y}{W}} \sin\frac{\pi x}{L} \qquad \text{Eq. (2)}$$

In your code you also write a subroutine to calculate temperature variation using the above analytical expression given by Eq. (2). Your results should contain at least the following:
(a)Compare the temperature contours obtained by the code and with those obtained by using the above analytical expression (show the temperature contours figures side by side).
(b) Compare the temperature variation with y along mid-vertical plane x = L/2 from the code.
and the above analytical expression. Plot both temperature profiles in the same figure.
(c) Compare the temperature variation with x along mid-horizontal plane y = W/2 from the code and the above analytical expression. Plot both temperature profiles in the same figure.

2.1 Physical System

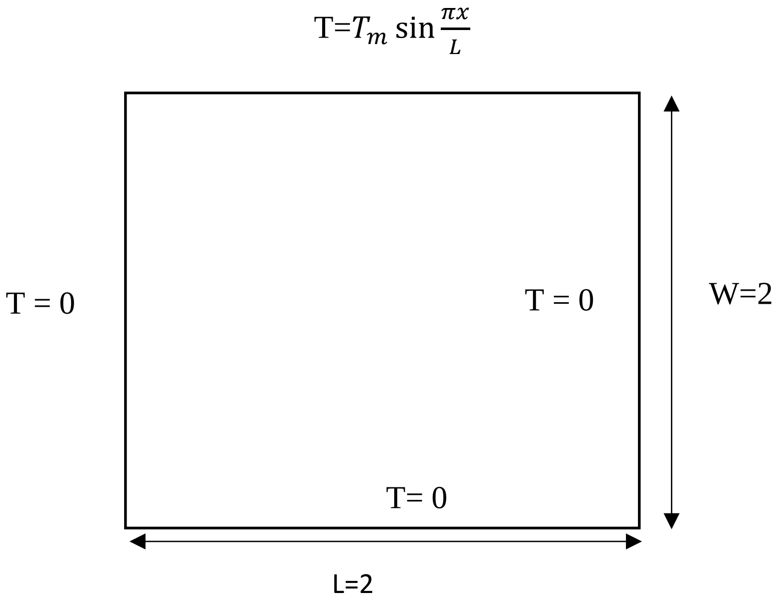
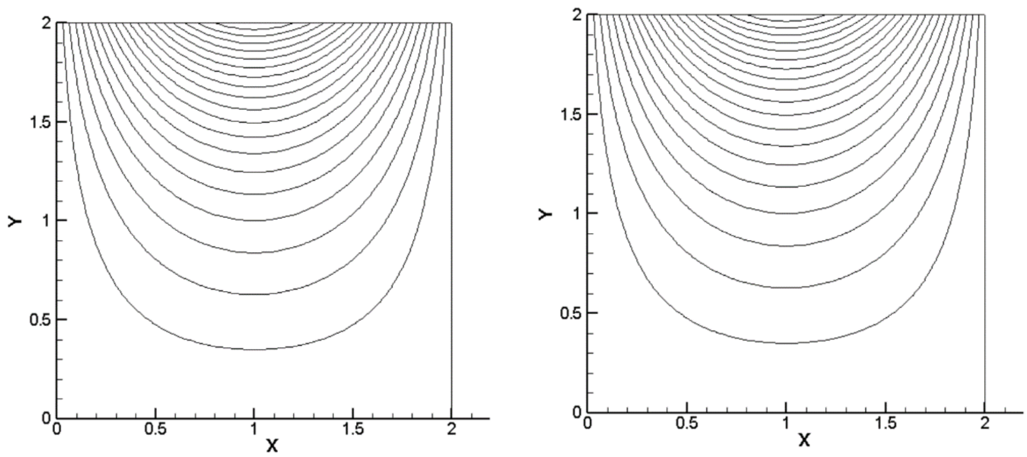


Figure 2.1: The physical system



(a) SIP result

(b) Analytical result

Figure 2.2: Temperature contours

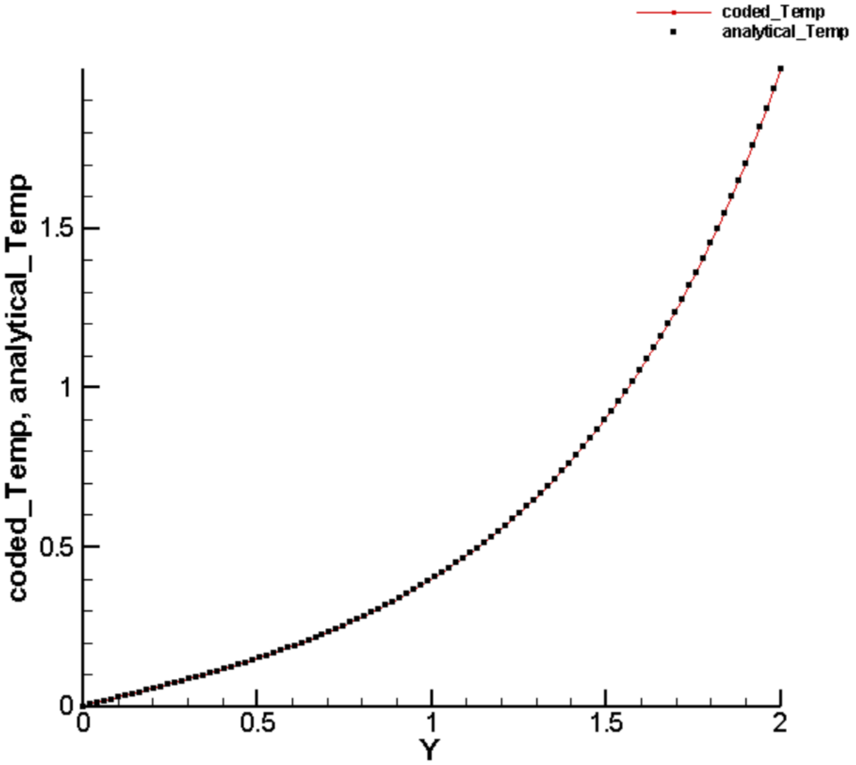


Figure 2.3: Temperature variation with y along mid-horizontal plane x = 0.5

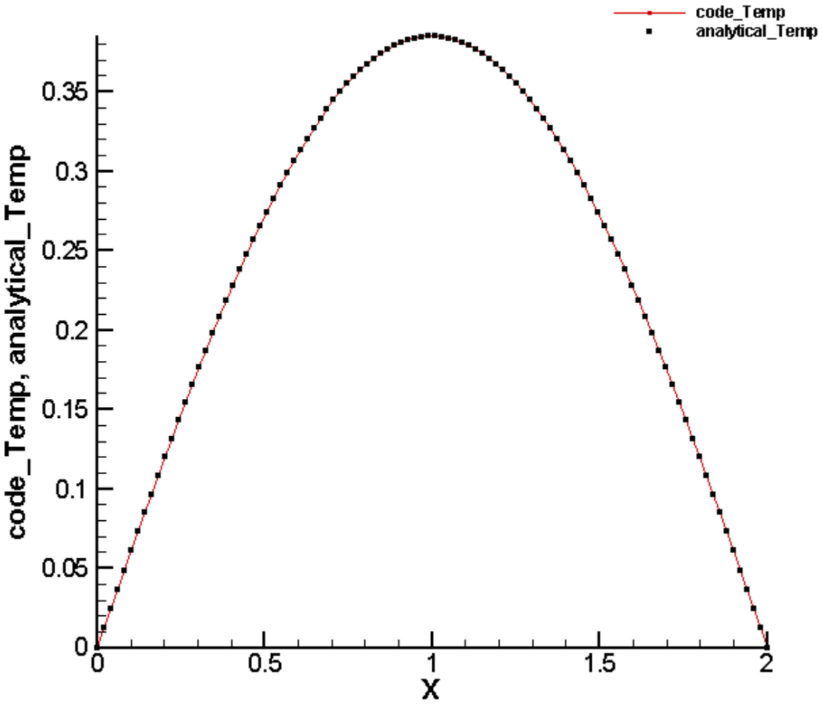


Figure 2.4: Temperature variation with x along mid-horizontal plane y = 0.5

2.2 The code of Strongly Implicit Method

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#define nx 100
#define ny 100
#define Tm 2
#define l 2.0
#define W 2.0
#define alpha 0.1
#define pi 3.14159265
double
Tnew[(nx*ny)+1],Told[(nx*ny)+1],b[(nx*ny)+1],Mp[(nx*ny)+1],Mnw[(nx*ny
)+1],Mw[(nx*ny)+1],Mn[(nx*ny)+1],Me[(nx*ny)+1],Ms[(nx*ny)+1],Mse[(nx*
ny)+1];
double
Lw[(nx*ny)+1],beta,Lp[(nx*ny)+1],Ls[(nx*ny)+1],Un[(nx*ny)+1],Ue[(nx*ny)
+1],dx,dy;int N;
double SIP();
double output_files();
double analytic();
double initializtion();
double solver();
int main()
{
        initializtion();
        SIP();
        solver();
```

```
	analytic();
	output_files();
	return 0;
}
double initializtion()
{
	int i,j,L,N;N=nx*ny;
	for(j=1;j<=ny;j++)
	{
		for(i=1;i<=nx;i++)
		{
			L=(i-1)*ny+j;
			Tnew[L]=0.0;
			Told[L]=0.0;
		}

	}
}
double SIP()
{
	int i,j,L;N=nx*ny;
	int Nj;double xpos;
	Nj=ny;dx=l/(double)(nx-1);
	dy=W/(double)(ny-1);
	beta=dx/dy;
	for(j=1;j<=ny;j++)
	{
```

```
for(i=1;i<=nx;i++)
{
          xpos=(i-1)*dx;
          L=(i-1)*ny+j;
          if(i==1)
          {
                    Mp[L]=1.0;
                    Mw[L]=0.0;
                    Mn[L]=0.0;
                    Ms[L]=0.0;
                    Me[L]=0.0;
              b[L]=0.0;
          }
          else if(i==nx)
          {
                    Mp[L]=1.0;
                    Mw[L]=0.0;
                    Mn[L]=0.0;
                    Ms[L]=0.0;
                    Me[L]=0.0;
                    b[L]=0.0;
          }
          else if(j==1)
          {
                    Mp[L]=1.0;
                    Mw[L]=0.0;
                    Mn[L]=0.0;
```

```
                Ms[L]=0.0;
                Me[L]=0.0;
                b[L]=0.0;
            }
            else if(j==ny)
            {
                Mp[L]=1.0;
                Mw[L]=0.0;
                Mn[L]=0.0;
                Ms[L]=0.0;
                Me[L]=0.0;
                b[L]=Tm*sin(pi*xpos/l);
            }
            else
            {
                Mp[L]=-2.0*(1+beta*beta);
                Mw[L]=beta*beta;
                Mn[L]=1.0;
                Ms[L]=1.0;
                Me[L]=beta*beta;
                b[L]=0.0;
            }
        } // i

    } // j
```

```
for(L=1;L<=N;L++)
{

        Lw[L]=Mw[L];
        if((L-ny)>=1)
        {
                Lw[L]=Lw[L]/(1.0+alpha*Un[L-ny]);
        }
        Ls[L]=Ms[L];
        if((L-1)>=1)
        {
                Ls[L]=Ls[L]/(1+alpha*Ue[L-1]);
        }
        Lp[L]=Mp[L];
        if((L-1)>=1)
        {
                Lp[L]=Lp[L]+alpha*Ls[L]*Ue[L-1]-Ls[L]*Un[L-1];
        }
        if((L-Nj)>=1)
        {
                Lp[L]=Lp[L]+alpha*Lw[L]*Un[L-Nj]-Lw[L]*Ue[L-Nj];
        }
        Un[L]=Mn[L]/Lp[L];
        if((L-Nj)>=1)
        {
                Un[L]=Un[L]-((alpha*Lw[L]*Un[L-ny])/Lp[L]);
        }
        Ue[L]=Me[L]/Lp[L];
        if((L-1)>=1)
        {
            Ue[L]=Ue[L]-((alpha*Ls[L]*Ue[L-1])/Lp[L]);
        }
    }
    for(L=1;L<=(N);L++)
    {
        Mnw[L]=0.0;
        if((L-Nj)>=1)
        {
            Mnw[L]=Lw[L]*Un[L-Nj];
        }

    }
    for(L=1;L<=(N);L++)
    {
        Mse[L]=0.0;
        if((L-1)>=1)
        {
        Mse[L]=Ls[L]*Ue[L-1];
        }
    }
}
double output_files()
{
    FILE *oup,*output2,*output3;double dx,dy,x,y;dx=l/(double)(nx-1);
    dy=W/(double)(ny-1);
    oup=fopen("SIP.dat","w");
    output2=fopen("SIP_vertical.dat","w");
    output3=fopen("SIP_horizontal.dat","w");
    int i,j,L;
    fprintf(oup,"VARIABLES=\"X\",\"Y\",\"coded_Temp\",\"analytical_Temp\"\n");
    fprintf(oup,"ZONE F=POINT\n");
    fprintf( oup, "I=%d, J=%d\t\n", nx, ny );
    for(j=1;j<=ny;j++)
    {
        for(i=1;i<=nx;i++)
        {
            x=(i-1)*dx;
            y=(j-1)*dy;
            L=(i-1)*ny+j;
            fprintf(oup,"%lf\t%lf\t%lf\t%lf\n",x,y,Tnew[L],Told[L]);
        }

    }
    int x1, x2,y1, y2,L1,L2;double avg1,avg2,avg,ana;
  x1=nx/2;
    x2=(nx/2)-1;
    fprintf( output2, "VARIABLES=\"Y\",\"coded_Temp\",\"analytical_Temp\"\n");
    for(j=1;j<=ny;j++)
    {
        L1=(x1-1)*ny+j;
```

```
            avg1=Tnew[L1];
            L2=(x2-1)*ny+j;
            avg2=Tnew[L2];
            ana=(Told[L1]+Told[L2])/2;
            avg=(avg1+avg2)/2;
            y=(j-1)*dy;
            fprintf(output2,"%5.8f\t%5.8f\t%5.8f\n",y,avg,ana);
      }
      y1=ny/2;
      y2=(ny/2)-1;
      fprintf( output3,
"VARIABLES=\"X\",\"code_Temp\",\"analytical_Temp\"\n");
      for(i=1;i<=nx;i++)
      {
            L1=(i-1)*ny+y1;
            avg1=Tnew[L1];
            L2=(i-1)*ny+y2;
            avg2=Tnew[L2];
            ana=(Told[L1]+Told[L2])/2;
            avg=(avg1+avg2)/2;
            x=(i-1)*dx;
            fprintf(output3,"%5.8f\t%5.8f\t%5.8f\n",x,avg,ana);
      }
      fclose(oup);
      fclose(output2);
      fclose(output3);
}
double solver()
```

```
{
    double Y[(nx*ny)+1],e,en,ed;
    int N,i,j,L;
    N=nx*ny;
    e=1.0;
    while(e>1.0e-14)
    {
        Y[1]=(b[1])/Lp[1];
        for(L=2;L<=N;L++)
        {
            Y[L]=b[L]-Ls[L]*Y[L-1];
            if((L-(ny-1))>=1)
            {
                Y[L]=Y[L]+Mnw[L]*(Told[L-(ny-1)]-alpha*(Told[L-ny]+Told[L+1]-Told[L]));
            }
            if((L+(ny-1))<=N)
            {
                Y[L]=Y[L]+Mse[L]*(Told[L+(ny-1)]-alpha*(Told[L+ny]+Told[L-1]-Told[L]));
            }
            if((L-ny)>=1)
            {
                Y[L]=Y[L]-Lw[L]*Y[L-ny];
            }
            Y[L]=Y[L]/Lp[L];
        }
        Tnew[N]=Y[N];
```

```
                for(L=N-1;L>=1;L--)
                {
                        Tnew[L]=Y[L]-Un[L]*Tnew[L+1];
                        if((L+ny)<=N)
                        {
                                Tnew[L]=Tnew[L]-Ue[L]*Tnew[L+ny];
                        }

                }
                en=0.0;
                ed=0.0;
                for(i=1;i<=N;i++)
                {
                        en+=fabs(Tnew[i]-Told[i]);
                        ed+=fabs(Tnew[i]);
                }
                e=en;
                for(i=1;i<=N;i++)
                {
                        Told[i]=Tnew[i];
                        Y[i]=0.0;
                }printf("%lf\n",e);
        }

}
double analytic()
{
```

```
int i,j,k,L;double dx,dy,x,y,sum[(nx*ny)+1],e,term;
        dx=l/(double)(nx-1);
        dy=W/(double)(ny-1);
        for(j=1;j<=ny;j++)
        {
                for(i=1;i<=nx;i++)
                {
                        L=(i-1)*ny+j;
                        x=dx*(i-1);
                        y=dy*(j-1);
                        e=1.0;
                        sum[L]=0.0;
                        sum[L]=(Tm*(sin((pi*x)/l))*sinh((pi*y)/l)/(sinh((pi*W)/l)));
                        Told[L]=sum[L];
                }
        }
}
```

Chapter 3

Conjugate Gradient Method

Question-3:

Code for conjugate gradient method:

Consider 2D conduction problem governed by equation:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = -8\pi^2[\sin 2\pi x \sin 2\pi y]$$

Take a square domain of width and height of 1 unit. The boundary conditions for all the boundaries are T = 0. Discretise the equation. Then solve the obtained system of linear equations by writing code for conjugate gradient method. This problem has an analytical solution which is given by:

$$T(x,y) = \sin 2\pi x \sin 2\pi y \qquad \text{Eq. (3)}$$

In your code you also write a subroutine to calculate temperature variation using the above analytical expression given by Eq. (3). Your results should contain at least the following:

(a) Compare the temperature contours obtained by the code and with those obtained by using the above analytical expression (show the temperature contours figures side by side).

(b) Compare the temperature variation with y along mid-vertical plane x = 0.5 from the code and the above analytical expression. Plot both temperature profiles in the same figure.

(c) Compare the temperature variation with y along mid-horizontal plane y = 0.5 from the code and the above analytical expression. Plot both temperature profiles in the same figure.

3.1 Physical System

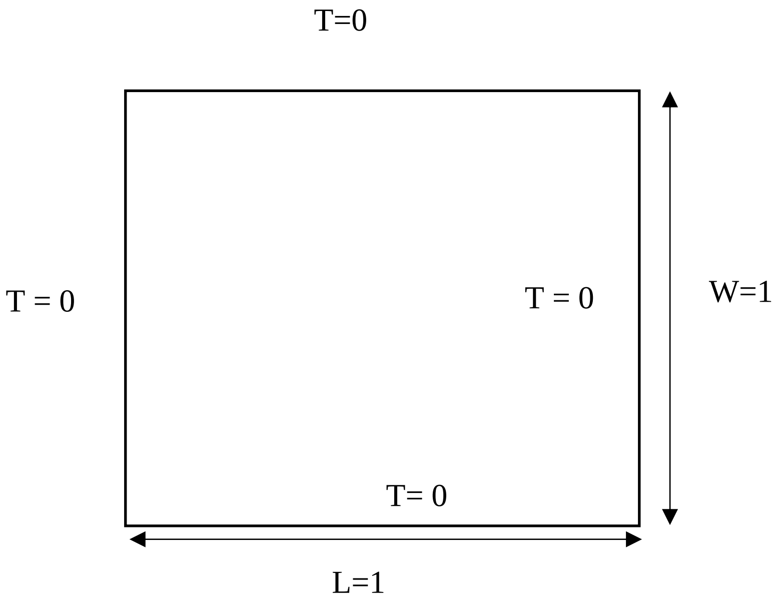
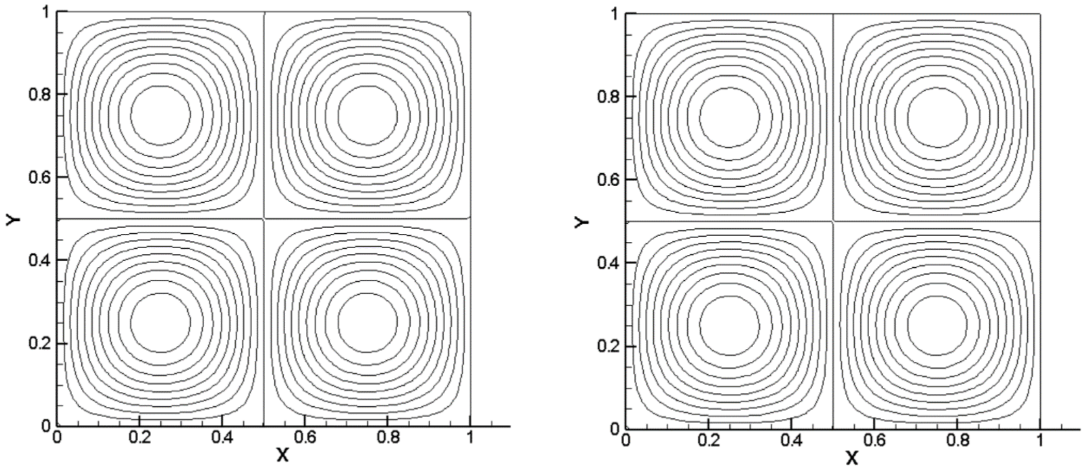


Figure 3.1: The physical system



(a) conjugated gradient method result

(b) Analytical result

Figure 3.2: Temperature contours

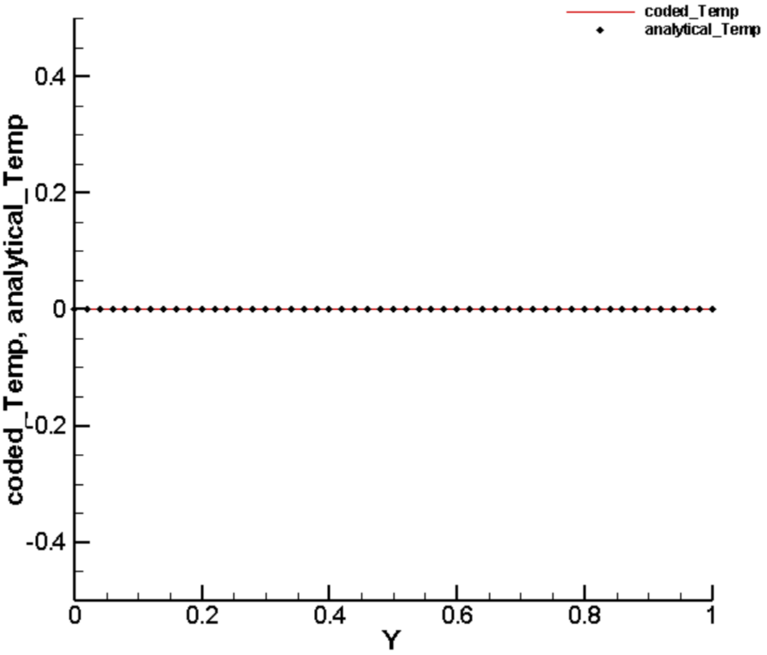


Figure 3.3: Temperature variation with y along mid-vertical plane x = 0.5

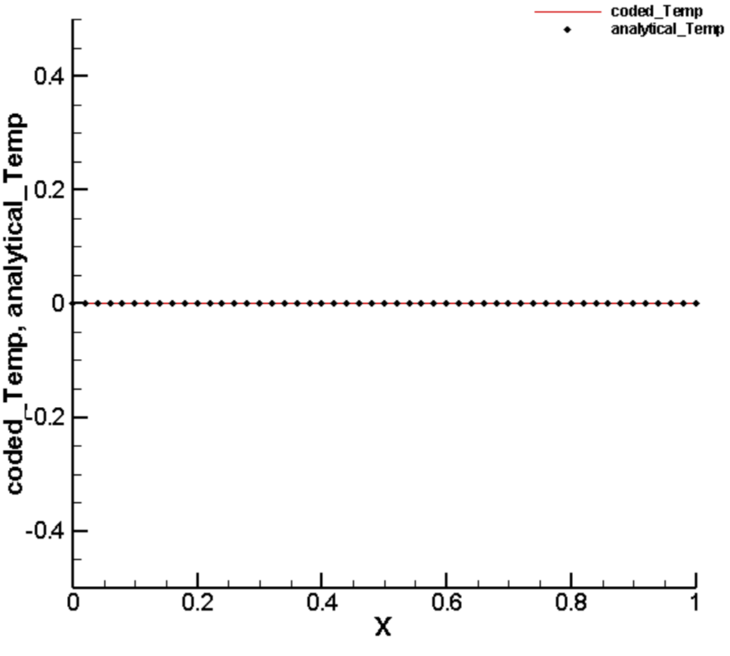


Figure 3.4: Temperature variation with x along mid-horizontal plane y = 0.5

3.2 The Code of Conjugate Gradient Method

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#define H 1.0
#define B 1.0
#define nx 100
#define ny 100
#define pi 3.14
double x[nx*ny],y[nx*ny],b[nx*ny],dx,dy,beta_matrix,actual[nx][ny];
double Mp[nx*ny],Me[nx*ny],Mw[nx*ny],Mn[nx*ny],Ms[nx*ny];
double solver();
double initialization();
double output_files();
double analytical();
int main()
{
   initialization();
        solver();
        analytical();
        output_files();
   return 0;
}
double output_files()
{
        FILE *output1,*output2,*output3;
   output1=fopen("conjugated.dat","w");
   output2=fopen("Mid_vertical_conjugated.dat","w");
   output3=fopen("Mid_horizontal_conjugated.dat","w");
   dx=H/(double)(nx-1.0);
   dy=B/(double)(ny-1.0);
   int i,j,L,n;n=nx*ny;
   fprintf( output1,
"VARIABLES=\"X\",\"Y\",\"coded_Temp\",\"analytical_Temp\"\n");
   fprintf(output1,"ZONE F=POINT\n");
        fprintf( output1, "I=%d, J=%d\t\n",nx,ny );
   dx=H/(double)(nx-1.0);
   dy=B/(double)(ny-1.0);
        double xp,yp;
   for(j=1;j<=nx;j++)
        {
      for(i=1;i<=ny;i++)
      {
        L=(i-1)*ny+j;
        yp=(j-1)*dy;xp=(i-1)*dx;

fprintf(output1,"%5.8f\t%5.8f\t%5.8f\t%5.8f\n",xp,yp,x[L],actual[i][j]);
      }
   }
   int x1,x2,y1,y2,L1,L2;
        double avg1,avg2,avg,ana;
   x1=nx/2;
        x2=(nx/2)-1;
        fprintf( output2,
"VARIABLES=\"Y\",\"coded_Temp\",\"analytical_Temp\"\n");
```

```
        for(j=1;j<=ny;j++)
        {
                L1=(x1-1)*ny+j;
                avg1=x[L1];
                L2=(x2-1)*ny+j;
                avg2=x[L2];
                ana=(actual[x1][j]+actual[x2][j])/2;
                avg=(avg1+avg2)/2;
                yp=(j-1)*dy;
                fprintf(output2,"%5.8f\t%5.8f\t%5.8f\n",yp,avg,ana);
        }
        y1=ny/2;
        y2=(ny/2)-1;
        fprintf( output3,
"VARIABLES=\"X\",\"coded_Temp\",\"analytical_Temp\"\n");
        for(i=1;i<=nx;i++)
        {
                L1=(i-1)*ny+y1;
                avg1=x[L1];
                L2=(i-1)*ny+y2;
                avg2=x[L2];
                ana=(actual[i][y1]+actual[i][y2])/2;
                avg=(avg1+avg2)/2;
                xp=(i-1)*dx;
                fprintf(output3,"%5.8f\t%5.8f\t%5.8f\n",xp,avg,ana);
        }
   fclose(output1);
   fclose(output2);
```

```
    fclose(output3);
}
double solver()
{
          int i,j;int n;n=nx*ny;
          dx=H/(double)(nx-1.0);
    dy=B/(double)(ny-1.0);
    for(i=1;i<=n;i++)
    {
        x[i]=0.0;
    }
    double p[n],r[n],rold[n],Ap[n],alpha,beta;
          double alpha_num, alpha_den, Ax[n];
    double error;
    int k;
          k=0;
    for (i=1; i<=n; i++)
    {
          if(i==1)
          {
                    Ax[i]=Mp[i]*x[i]+Mn[i]*x[i+1]+Me[i]*x[ny+i];
          }
                    else if((i>1)&&(i<=ny))
                    {
                              Ax[i]=Ms[i]*x[i-1]+ Mp[i]*x[i]
                                     +Mn[i]*x[i+1]+Me[i]*x[i+ny];
                    }
        else if((i>ny)&&(i<=(n-ny)))
        {
             Ax[i]=Mw[i]*x[i-ny]+ Ms[i]*x[i-1]+Mp[i]*x[i]
                 +Mn[i]*x[i+1]+Me[i]*x[i+ny];
        }
        else if((i>ny)&&(i<(n)))
        {
             Ax[i]=Mw[i]*x[i-ny]+Ms[i]*x[i-1]+Mp[i]*x[i]
                 +Mn[i]*x[i+1]+Me[i]*x[i+ny];
        }
        else if(i==n)
        {
             Ax[i]=Mw[i-1]*x[i-ny-1]+Ms[i-1]*x[i-2]
                 +Mp[i-1]*x[i-1];
        }
        p[i]=b[i]-Ax[i];
        r[i]=b[i]-Ax[i];
   }
   error=1.0;
do
{
  error=0;
        alpha_num=0;
        alpha_den=0;
        for (i=1; i<=n; i++)
  {
   if(i==1)
```

```
{
    Ap[i]=Mp[i]*p[i]+Mn[i]*p[i+1]+Me[i]*p[ny+i];
            alpha_num+=r[i]*r[i];
            alpha_den+=p[i]*Ap[i];
        }
        else if((i>1)&&(i<=ny))
        {
            Ap[i]=Ms[i]*p[i-1]+Mp[i]*p[i]
               +Mn[i]*p[i+1]+Me[i]*p[i+ny];
            alpha_num+=r[i]*r[i];
            alpha_den+=p[i]*Ap[i];
        }
        else if((i>ny)&&(i<=(n-ny)))
        {
            Ap[i]=Mw[i]*p[i-ny]+Ms[i]*p[i-1]+Mp[i]*p[i]
               +Mn[i]*p[i+1] + Me[i]*p[i+ny];
            alpha_num+=r[i]*r[i];
            alpha_den+=p[i]*Ap[i];
        }
        else if((i>(n-ny))&&(i<(n)))
        {
            Ap[i]=Mw[i]*p[i-ny]+Ms[i]*p[i-1]+Mp[i]*p[i]
               +Mn[i]*p[i+1];
            alpha_num+=r[i]*r[i];
            alpha_den+=p[i]*Ap[i];
        }
        else if(i==n)
```

```
                              {
                                        Ap[i]=Mw[i]*p[i-ny]+Ms[n]*p[i-1]+Mp[i]*p[i];
                                        alpha_num+=r[i]*r[i];
                                        alpha_den+=p[i]*Ap[i];
                              }
                    }
        alpha=alpha_num/alpha_den;
                    double beta_num,beta_den;
        beta_num=0;
                    beta_den=0;
        for (i=1; i<=n; i++)
        {
            x[i]=x[i]+alpha*p[i];
                              rold[i]=r[i];
                              beta_den+=rold[i]*rold[i];
            r[i]=r[i]-alpha*Ap[i];
            beta_num+=r[i]*r[i];
            error+=r[i]*r[i];
        }
        beta=beta_num/beta_den;
        for (i=1; i<=n; i++)
        {
            p[i]=r[i]+beta*p[i];
        }
        k++;
        printf("%d\t\t%5.8f\n",k,error);
    }while (error>1.0e-14);
```

```
}
double analytical()
{
        int i,j;double xp,yp;
        dx=H/(double)(nx-1.0);
   dy=B/(double)(ny-1.0);
        for(i=1;i<=nx;i++)
        {
                for(j=1;j<=ny;j++)
                {
                        xp=(i-1)*dx;
                        yp=(j-1)*dy;
                        actual[i][j]=sin(2.0*pi*xp)*sin(2.0*pi*yp);
                }
        }
}
double initialization()
{
        int i,j,L,n;
        n=nx*ny;
        double xp,yp;
        dx=H/(double)(nx-1.0);
   dy=B/(double)(ny-1.0);
        beta_matrix=dx/dy;
   for(i=1;i<=nx;i++)
   {
      for(j=1;j<=ny;j++)
```

```
{
    L=(i-1)*ny+j;
   if(i==1)
   {
       Mp[L]=1.0;
       Me[L]=0.0;
                         Mw[L]=0.0;
                         Mn[L]=0.0;
                         Ms[L]=0.0;
                         b[L]=0.0;

   }
               else if(i==nx)
               {
                         Mp[L]=1.0;
       Me[L]=0.0;
                         Mw[L]=0.0;
                         Mn[L]=0.0;
                         Ms[L]=0.0;
                         b[L]=0.0;
               }
               else if(j==1)
               {
                         Mp[L]=1.0;
       Me[L]=0.0;
                         Mw[L]=0.0;
                         Mn[L]=0.0;
```

```
                Ms[L]=0.0;
                b[L]=0.0;
           }
           else if(j==ny)
           {
                Mp[L]=1.0;
       Me[L]=0.0;
                Mw[L]=0.0;
                Mn[L]=0.0;
                Ms[L]=0.0;
                b[L]=0.0;
           }
     else
     {
       xp=(i-1)*dx;
                yp=(j-1)*dy;
       Mp[L]=-2.0*(1+beta_matrix*beta_matrix);
                Mw[L]=beta_matrix*beta_matrix;
                Mn[L]=1.0;
                Ms[L]=1.0;
                Me[L]=beta_matrix*beta_matrix;
       b[L]=-8.0*pi*pi*(sin(2.0*pi*xp)*sin(2.0*pi*yp))*dx*dx;
     }
    }
  }
}
```