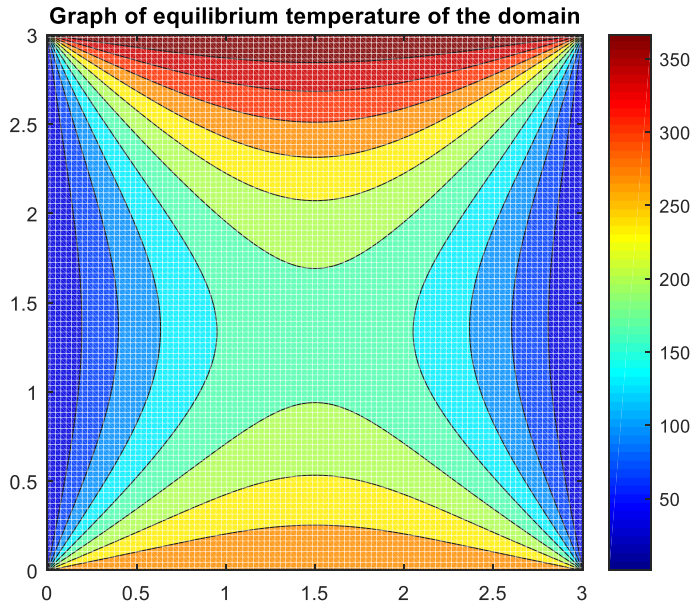
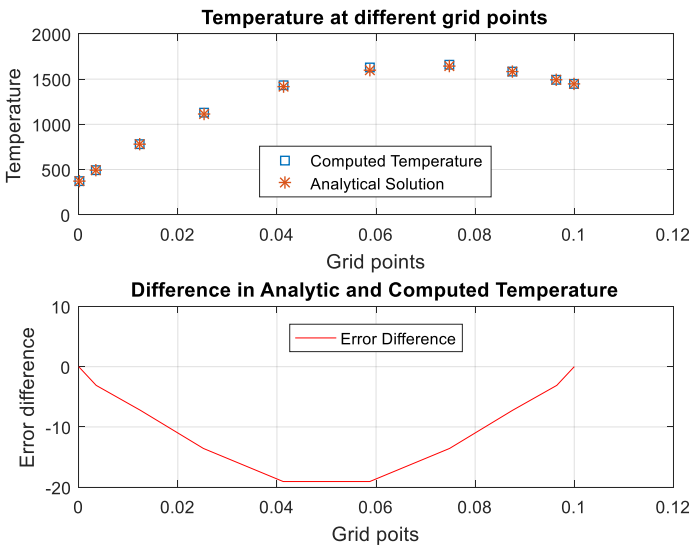
17/02/2018

ASSIGNMENT 1

COMPUTATIONAL FLUID DYNAMICS AND HEAT TRANSFER





173109003 | Sanit Prashant Bhatkar

Computational Fluid Dynamics and Heat Transfer (ME 415), Autumn 2018

Assignment 1: Heat Conduction - I

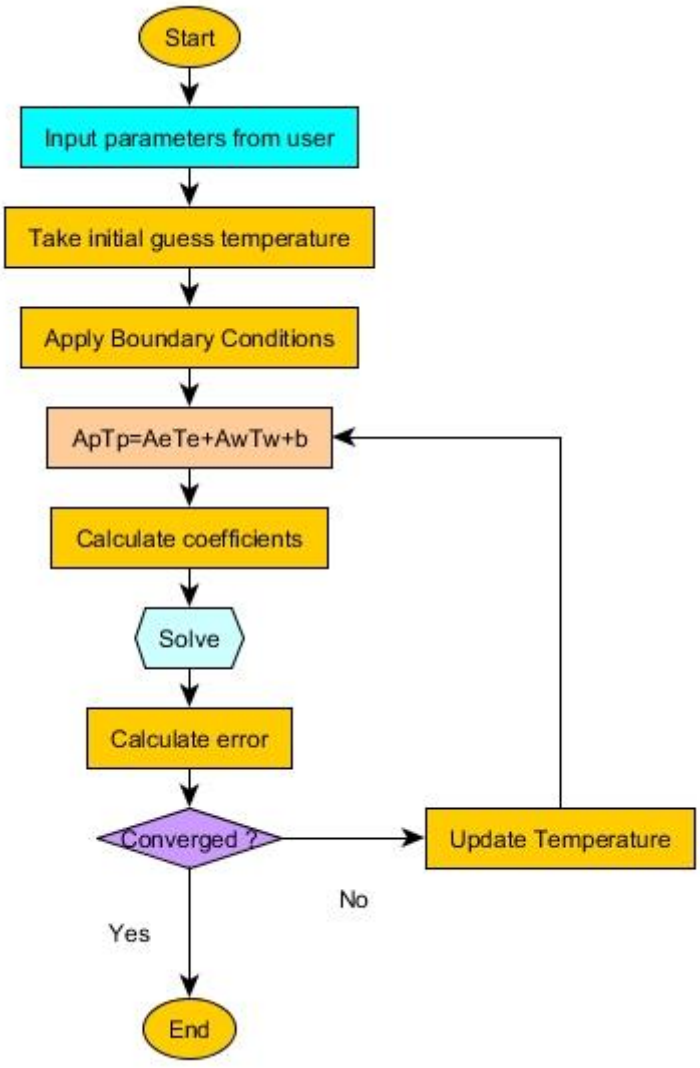
Name: Sanit Prashant Bhatkar

Roll No: 173109003

Problem 1:

In a large 10 cm thick heat generating Solid plate of k = 16 W/m.K, is uniformly generating at a rate of 8×106 W/m³. Left side of the plate is kept at 100°C while the other side is exposed to a flowing fluid environment at 20°C with a heat transfer coefficient of 200 W/m²K. Use the Finite Volume Method Formulation to solve the problem. Use the Grid generating equation to generate Non uniform Mesh. Consider maximum number of grid points as n and β=1.2

Algorithm



Grid details and the implemented boundary condition

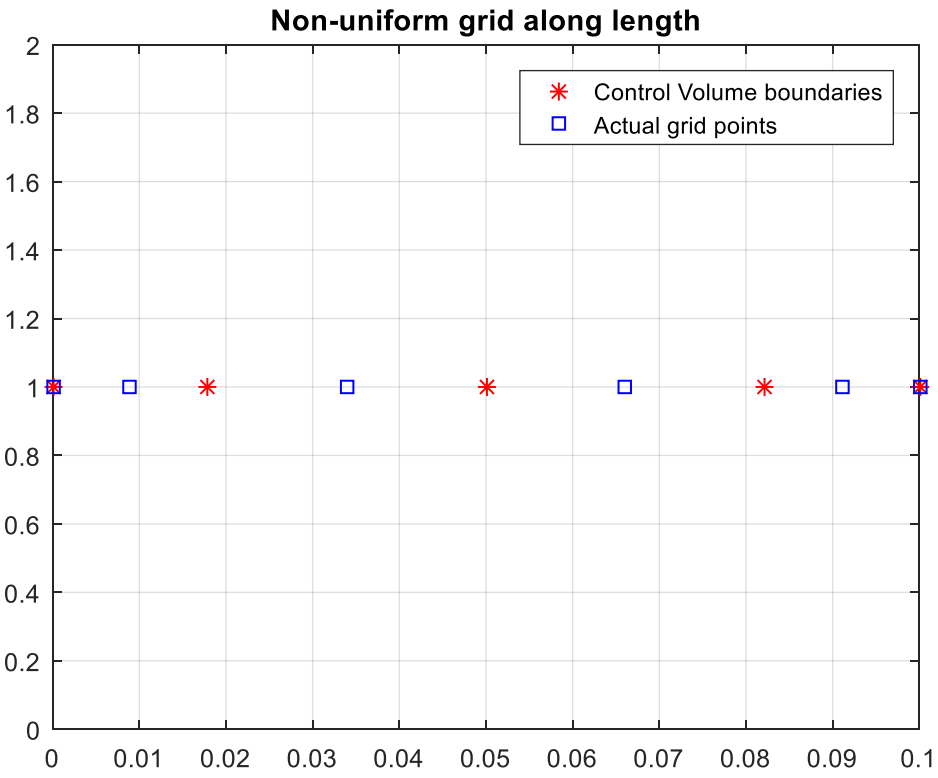


Fig 1. Generated grid for random Cartesian geometry

Boundary Conditions

- On the left boundary uniform temperature 373K is applied
- On the right boundary formula was worked out.

Integration of governing equation for the control volume

$$\int_w^e \frac{\mathrm{d}}{\mathrm{d}x}\left(k\frac{\mathrm{d}T}{\mathrm{d}x}\right)\cdot dx + \int_w^e S\cdot dx$$

Boundary condition at right end grid point

$$-k\frac{\mathrm{d}T}{\mathrm{d}x}\bigg|_{x=L} = h(T|_{x=L} - T_\infty)$$

Worked out formula

$$T|_{x=L} = \frac{hT_\infty + k\frac{T_p}{\delta x_w}}{h + \frac{k}{\delta x_w}}$$

Here Tp is penultimate point in the grid. All other symbols have the usual meaning.

Code of the problem

```
%============================================================%
% Assignment 1 CHDHT ME 415
% Problem Number 1. Heat generation in solid
% Progarm uses FVM to solve the problem of 1D conduction
% Heat generation is present and Steady state is present
% Designed only for rectangular co-ordinate system
%
% AUTHOR:
% Sanit P. Bhatkar (173109003@iitb.ac.in)
% Roll No: 173109003
% Place: IIT BOMBAY.
%============================================================%


%==============================INPUT OF VARIABLES============================
clc
clearvars
fprintf('\n============ Heat generation in solid ============');
fprintf('\n\n---------------------- INPUT ----------------------\n');
%l=0.1;
%n=10
%b=1.2;
%h=200;
%k=16;
%q=8*10^6;
%tbw=100+273;
%Tinf=20+273;
%Tini=0;
%eps=10^-6;



%------------Grid parameters---------

fprintf('\nGrid Parameters -->\n');
l=input('\nLength of the domain in m: ');
n=input('Maximum grid points along length: ');
b=input('Input value of beta: ');


%------------Bondary Conditions---------

fprintf('\nProperties and Bondary Conditions -->\n');
k=input('\nInput conductivity of the material: ');
h=input('Heat transfer coefficient at right boundary: ');
q=input('Input volumetric heat generation rate: ');
tbw=input('Input temperature at left boundary in K: ');
Tinf=input('Input ambient temperature K: ');
Tini=input('Input initial guess temperature in K: ');

%Convergence Definition
eps=input('\nEnter minimum convergence error: ');

%============================FINITE VOLUME METHOD============================

%STEP 1: Divide domain into finite sized subdomain called control volumes

%Our domain is non-uniform

%------------Nonuniform Grid Generation---------
```

```
%z represents zeta in uniform coordinates ranging 0 to 1
%dz is defined to get unit uniform grid size
dz=1/(n-2);
z=0:dz:1;

for i=1:n-1

%t represents numerator
t(i)=l*((b+1)*[(b+1)/(b-1)]^(2*z(i)-1)-(b-1));
%d represents denominator
d(i)=2*(1+[(b+1)/(b-1)]^(2*z(i)-1));
%x represents transformed co-ordinate
x(i)=t(i)/d(i);

end


%As the formula is giving us only the control volumes
%Calculating for control points
%Initial and final points will be the same
xx=zeros(1,n);
for i=1:n-2
    %Point must lie on the center of defined of control volume
    xx(i+1)=(x(i)+x(i+1))/2;
end
xx(1)=x(1);
xx(n)=x(n-1);

figure(1)

%Control volume plotting
%Red points represent the boundaries of control volume
y=ones(1,n-1);
plot(x,y,'*r');
xlim([0 x(n-1)]);
title('Non-uniform grid along length')
grid on
hold on

%Grid point plotting
%Blue squares represent the actual points
yy=ones(1,n);
plot(xx,yy,'bs');
hold off

%STEP 2: Integrate governing equation over boundaries of control volume

%STEP 3: Profile assumption

%Piecewise linear profile is assumed for integral
%We will get equation of type
%ApTp=AeTe+AwTw+AnTn+AsTs+B
%Assuming isotropic material and uniform control volume

%----------PART A OF THE ASSIGNMENT---------

%Solved by point to point gauss seidel


%=========================GAUSS SEIDEL METHOD============================
```

```
%Formulating solution of type [A][Tp]=[B]

%Definging matrix for temperature
%T represents matrix for visualization of grids.


%Initializing the Temperature matrix

T(1,1:n)=Tini;
%Boundary conditions at left boundary
T(1,1)=tbw;
dxw=xx(n)-xx(n-1);
T(1,n)=(h*Tinf+k*(T(1,n-1)/(dxw)))/(h+(k/(dxw)));

mxe=10;
dummy=T;
itr=0;

%Defining random points for contour plots vs iterations

%k represents x cordinate of random point

k1=find(xx==xx(round(n/3)));
k2=find(xx==xx(round(n/2)));
k3=find(xx==xx(round(2*n/3)));

while mxe > eps
    mxe=0;
for i=2:n-1

    dxe=xx(i+1)-xx(i);
    dxw=xx(i)-xx(i-1);
    dx=x(i)-x(i-1);

    ae=k/dxe;
    aw=k/dxw;
    ap=ae+aw;

    T(1,i)=(ae*T(1,i+1)+aw*T(1,i-1)+q*dx)/ap;

    %Absolute error definition is used
    error=abs(T(1,i)-dummy(1,i));

    %Identifying the max error
    if error>mxe
        mxe=error;
    end

end

%Boundary condition at east boundary
dxw=dxe;
%worked out formula
nu=h*Tinf+k*(T(1,n-1)/dxw);
dnm=h+(k/dxw);
T(1,n)=nu/dnm;

dummy=T;
%Calculating number of iterations
itr=itr+1;

%Storing change of temperature of random points in different matrices
```

```
p1(itr+1,1)=T(1,k1);
p2(itr+1,1)=T(1,k2);
p3(itr+1,1)=T(1,k3);

end

%Analytic solution
%Formula is worked out for this
c2=tbw;
c1=((q*l/h)+((q*l^2)/(2*k))-tbw+Tinf)/((k/h)+l);
TT(1,1:n)=((-q*(xx(1,1:n)).^2)/(2*k)+c1.*xx(1,1:n)+c2);

%Difference in calculated error between theory and computational model

diff=TT-T;

%============================== OUTPUT ======================================

fprintf('\n\n---------------------- OUTPUT -----------------------\n\n');
fprintf('Number of iterations for grid points = %d are %d\n\n',n,itr);

figure(2)

plot([1:itr+1],p1);
title('Random point temperature vs iterations')
hold on

plot([1:itr+1],p2);
hold on

plot([1:itr+1],p3);
hold off
xlabel('Iterations');
ylabel('Temperature');

legend(['p1 (',num2str(xx(round(n/3))),',0)'],['p2
(',num2str(xx(round(n/2))),',0)'],['p3
(',num2str(xx(round(2*n/3))),',0)'],'Location','South');
grid on

%Analytic and computed solution plot along with error differnce
%Error is Analytic - Computed

figure (3)

subplot(2,1,1)
plot(xx,T,'s')
title('Temperature at different grid points');
hold on
plot(xx,TT,'*')
hold off
grid on
xlabel('Grid points');
ylabel('Temperature');
legend('Computed Temperature','Analytical Solution','Location','South');

subplot(2,1,2)
plot(xx,diff,'r')
title('Difference in Analytic and Computed Temperature');
grid on
xlabel('Grid points');
ylabel('Error difference');
legend('Error Difference','Location','North');
```

```
%Energy Balance for control volume

for i=2:n-1

LHS = (T(1,i)-T(1,i-1))/(xx(i)-xx(i-1))-q*(x(i)-x(i-1))/k;
RHS= (T(1,i+1)-T(1,i))/(xx(i+1)-xx(i));
Flux_change(i-1)= RHS-LHS;

end

%flux change at boundary
LHS=-(T(1,n)-T(1,n-1))/(xx(n)-xx(n-1));
RHS=h*(T(1,n)-Tinf)/k;
Flux_change(n-1)=RHS-LHS;
Flux_change

%_____________________________________________________________________________________________
```

Results and Discussion

A. Temperature distribution of the wall for n=5,10and 20.

Table 1

Number of points	Temperature at wall
5	1439.666 K
10	1439.666 K
20	1439.666 K

B. Data values at specific grid points

Number of grid point: 10

Minimum convergence error: 10^-6

---------- OUTPUT ----------

Number of iterations for grid points = 10 are 417

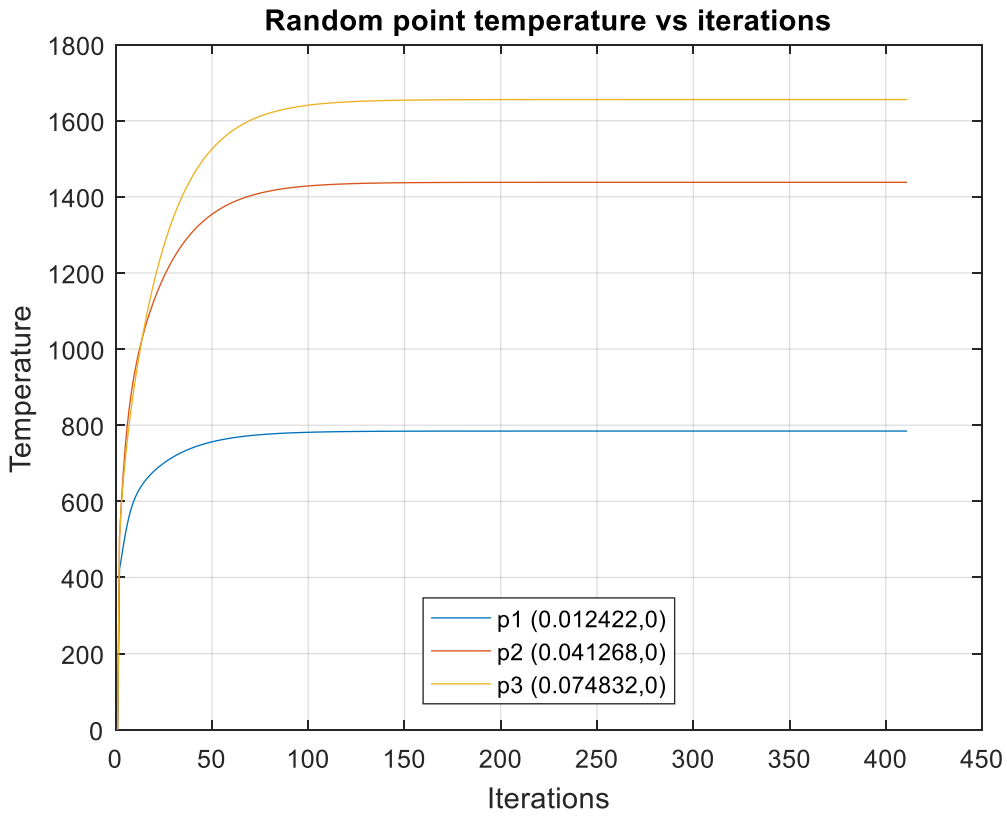


Fig 2. Variation of temperature with Iterations

C. Analytic solution and Comparison

Solving governing equation

$$\frac{\partial^2 T}{\partial x^2} = -\frac{q'''}{k}$$

Integrating twice will give us

$$\frac{\mathrm{d}T}{\mathrm{d}x} = \frac{-q'''x}{k} + C_1$$

$$T(x) = \frac{-q''' \cdot x^2}{2k} + C_1 \cdot x + C_2$$

Where values of constant C_1 and C_2 are

$$C_2 = T|_{x=0}$$

$$C_1 = \frac{\frac{q'''L}{h} + \frac{q'''L^2}{2k} - T|_{x=0} + T_\infty}{\frac{k}{h} + L}$$

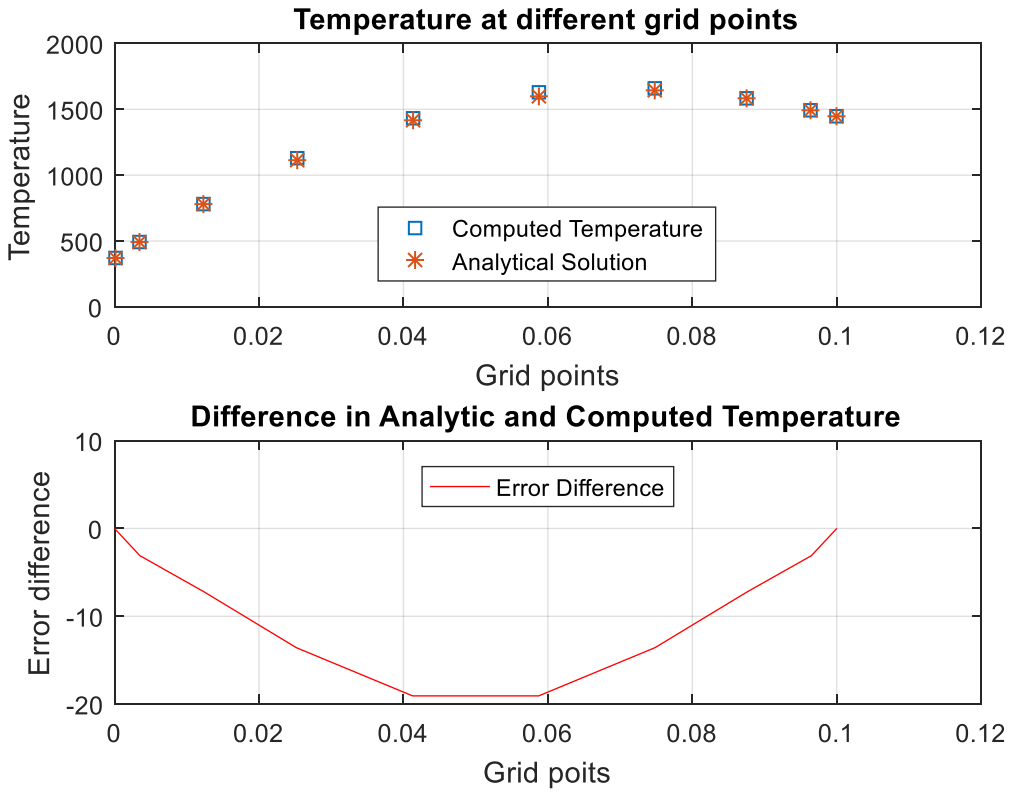


Fig 3. Error bar at different grid points

Computed value is subtracted from analytic value for the calculation of the error. Deviation is maximum at the middle section. This is because points have more spacing at mid-sections due to non-uniform gridding.

D. Heat balance for Numerical solution

General energy conservation

$$\dot{Q_{in}} - \dot{Q_{out}} + \dot{Q_{gen}} = 0$$

Substituting for the terms in equation

$$-k\frac{\mathrm{d}T}{\mathrm{d}x}\bigg|_W + q'''\Delta x = -k\frac{\mathrm{d}T}{\mathrm{d}x}\bigg|_E$$

Expansion of derivative at a point gives the result

$$k \cdot \frac{T_P - T_W}{x_P - x_W} + q''' \cdot (x_e - x_w) = k \cdot \frac{T_E - T_P}{x_E - x_P}$$

Net flux change is calculated in the code

$$Flux\,Change = LHS - RHS$$

After implementing the code various fluxes were calculated for different convergence criteria. Details of the flux change are provided in the Table 2. Number of control volumes will be always one size less than number of points. Table 2 has values valculated for 8 points with convergence error 10^-8

Table 2

Control Volume Number	Flux change
Ideal Case	0
1	2.1605e-07
2	2.7986e-07
3	3.5074e-07
4	4.7251e-07
5	7.0980e-07
6	1.7841e-06
7	1.4552e-11

E. Output

========== Heat generation in solid ==========

---------- INPUT ----------

Grid Parameters -->

Length of the domain in m: 0.1

Maximum grid points along length: 10

Input value of beta: 1.2

Properties and Bondary Conditions -->

Input conductivity of the material: 16

Heat transfer coefficient at right boundary: 200

Input volumetric heat generation rate: 8*10^6

Input temperature at left boundary in K: 373

Input ambient temperature K: 293

Input initial guess temperature in K: 400

Enter minimum convergence error: 10^-6

---------- OUTPUT ----------

Number of iterations for grid points = 10 are 410

Flux_change =

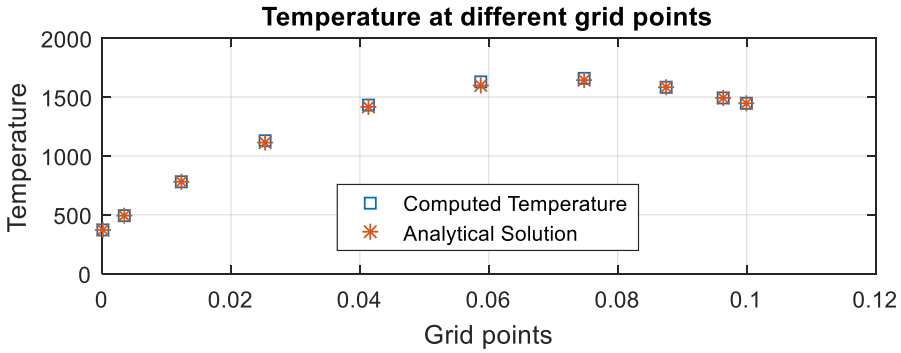
1.0e-03 *

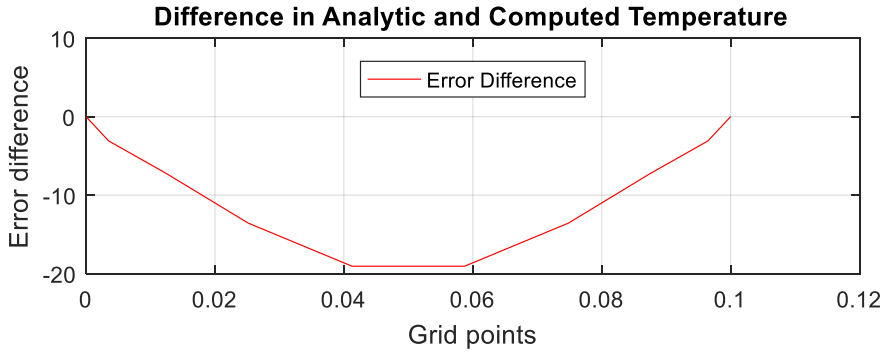
Columns 1 through 8

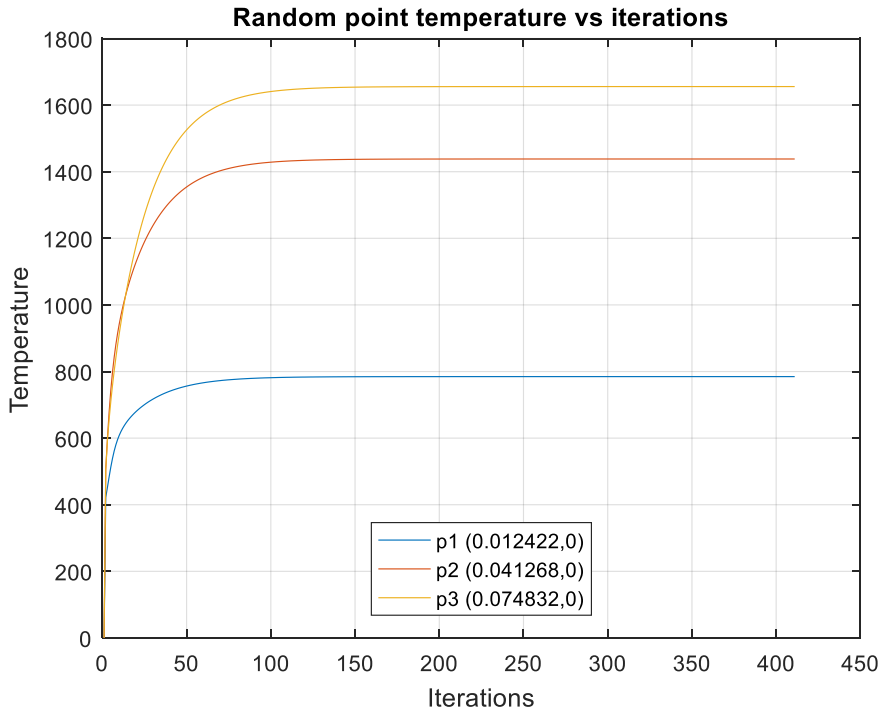
0.0226 0.0303 0.0369 0.0450 0.0572 0.0772 0.1107 0.2678

Column 9

0.0000





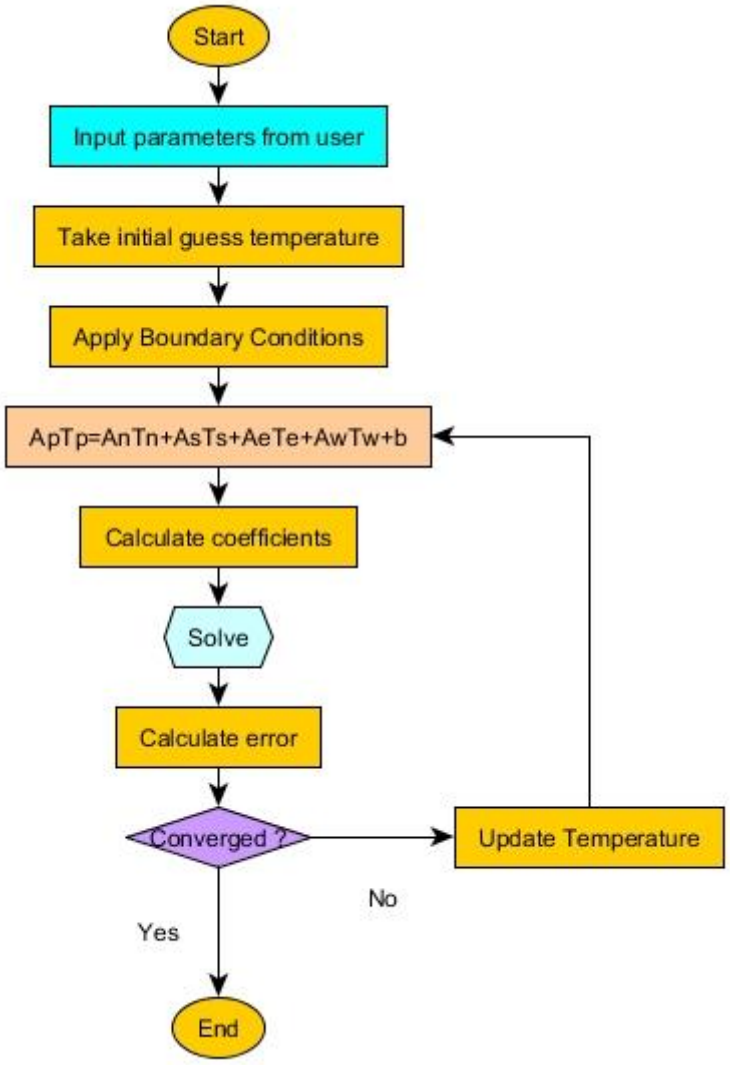


Problem 2:

A large industrial furnace is supported on a column of clay bricks which is of length L = 1.5 m in x-direction by H = 3 m in y-direction. The installation during the steady-state process is such that the column is subjected to following boundary conditions:
(i) $y = 0$, $T = 300^{o}$C, *(ii)*$x = 0$, $T = 30\,^{o}$C, *(iii)* $y = H$, $T = 400\,^{o}$C and *(iv)* $x = L$, $T = 30\,^{o}$C.

Choose a uniform grid size of 0.05 m in both directions.
a. Solve the problem using the point-by-point Gauss-Seidel iterative method.
b. Experiment with the initial guess and comment on the number of iterations required for convergence in each case.
c. Clearly explain your convergence criterion for the iterations and how it is implemented.
d. Plot the temperature contours as the output. Please provide values of some isotherms.

Algorithm



Grid details and the implemented boundary condition

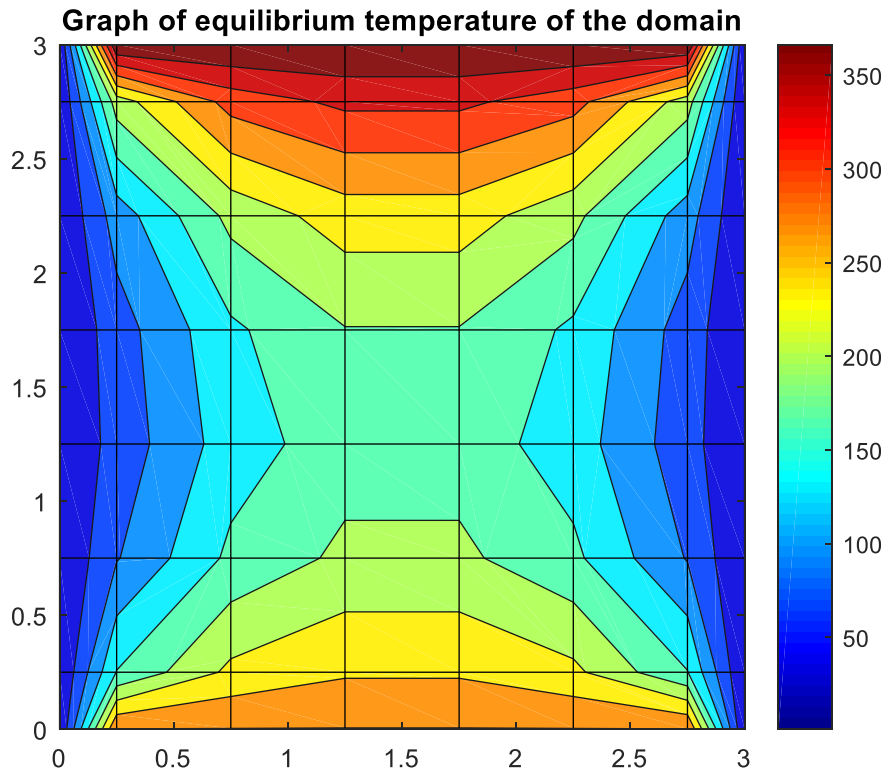


Fig 4. Generated uniform grid for random Cartesian geometry

Grid Specifications

A uniform grid was generated. Black lines indicate lines passing through the actual points on the domain. Control volume surfaces lie at half the unit distance from them.

Boundary Conditions

- On the left boundary uniform temperature 30^{o}C is applied
- On the right boundary uniform temperature 30^{o}C is applied
- On the top boundary uniform temperature 400^{o}C is applied
- On the bottom boundary uniform temperature 300^{o}C is applied

Code of the problem

```
%============================================================%
% Assignment 1 CHDHT ME 415
% Problem Number 2. Column of bricks supporting furnace
% Progarm uses FVM to solve the problem of 2D conduction
% Heat generation is zero and Steady state is present
% Designed only for rectangular co-ordinate system
%
% AUTHOR:
% Sanit P. Bhatkar (173109003@iitb.ac.in)
% Roll No: 173109003
% Place: IIT BOMBAY.
%============================================================%


%==============================INPUT OF VARIABLES==========================
clc
clearvars
fprintf('\n============ Clay brick coulumn problem ============');
fprintf('\n\n---------------------- INPUT ----------------------\n');
%l=1.5;
%h=3;
%dl=0.05;
%dh=0.05;
%k_cond=input('\nInput conductivity of the material: ');
%------------Geometry definition--------------

fprintf('\nGeometry Parameters -->\n');
l=input('\nLength of column in m: ');
h=input('Height of column in m: ');
dl=input('Grid size along length in m: ');
dh=input('Grid size along height in m: ');

%------------Boundary condition input---------

%tbn = temp on north boundary
%tbs = temp on south boundary
%tbe = temp on east boundary
%tbw = temp on west boundary

%tbn=400;
%tbs=300;
%tbe=30;
%tbw=30;

fprintf('\nBoundary Conditions -->\n');
tbn=input('\nBoundary condition at north boundary in degree C:  ');
tbs=input('Boundary condition at south boundary in degree C:  ');
tbe=input('Boundary condition at east boundary in degree C:  ');
tbw=input('Boundary condition at west boundary in degree C:  ');


%============================FINITE VOLUME METHOD=========================

%STEP 1: Divide domain into finite sized subdomain called control volumes

%Our domain is uniform
%But boundary and the first point grid space is different
%That is why separate calculation is done
%sx represents the number of points in a grid in x direction
%NOTE: dl has to be some multiple of length
```

```
sx=(l/dl)+2;
x(1,1)=0;
x(sx,1)=l;
x(2,1)=x(1,1)+(dl/2);
for m=3:sx-1
    x(m,1)=x(m-1,1)+dl;
end

%sy represents the number of points in a grid in y direction

sy=(h/dh)+2;
y(1,1)=0;
y(sy,1)=h;
y(2,1)=y(1,1)+(dh/2);

for m=3:sy-1
    y(m,1)=y(m-1,1)+dh;
end


%STEP 2: Integrate governing equation over boundaries of control volume

%STEP 3: Profile assumption

%Piecewise linear profile is assumed for integral
%We will get equation of type
%ApTp=AeTe+AwTw+AnTn+AsTs+B
%Assuming isotropic material and uniform control volume with no heat generation
%We get 4*Tp=Te+Tw+Tn+Ts

%----------PART A OF THE ASSIGNMENT---------

%Solve by point to point gauss seidel


%==========================GAUSS SEIDEL METHOD==============================

%Formulating solution of type [A][Tp]=[B]

%Definging matrix for temperature
%T represents matrix for visualization of grids.

%----------PART B OF THE ASSIGNMENT-------

%Play with initial values and find the iterations

T=zeros(sy,sx);
T(1:sy,1:sx)= input('\nInitial guess of temperature for the domain: ');
T_ini=T(1,1);

%Application of boundary conditions
T(1,1:sx)=tbn;
T(sy,1:sx)=tbs;
T(1:sy,sx)=tbe;
T(1:sy,1)=tbw;

%dummy is defined to store old values while error calculation
dummy=T;

%---------PART C OF THE ASSIGNMENT----------

%Definition of convergence
```

```
%Formula for residual will be used in the code

eps=input('\nEnter minimum convergence error: ');
%eps=10^-(any value);
mxe = 10;

%Defining random points for contour plots vs iterations

%k represents x cordinate of random point

k1=find(x==x(round(sx/3)));
k2=find(x==x(round(sx/2)));
k3=find(x==x(round(2*sx/3)));

%m represents x cordinate of random point

m1=find(y==y(round(sy/3)));
m2=find(y==y(round(sy/2)));
m3=find(y==y(round(2*sy/3)));

%Defining random points for contour plots vs iterations
p1(1,1)=T(m1,k1);
p2(1,1)=T(m2,k2);
p3(1,1)=T(m3,k3);

%As we are using gauss seidel we will update previous value every time step
%Gauss seidel equation will have form
%T22=(T23+T21+T12+T32)/4

itr=0;

while mxe > eps
    mxe=0;
for i=2:sy-1

for j=2:sx-1

%dxe is distance from point P to east point
%dxw is distance from point P to west point
%dyn is distance from point P to north point
%dys is distance from point P to south point

dxe=x(j+1,1)-x(j,1);
dxw=x(j,1)-x(j-1,1);
dyn=y(i,1)-y(i-1,1);
dys=y(i+1,1)-y(i,1);

%apTp=aeTe+awTw+anTn+asTs

ae=dh/dxe;
aw=dh/dxw;
an=dl/dyn;
as=dl/dys;
ap=ae+aw+an+as;

%Worked out formula

T(i,j)=(ae*T(i,j+1)+aw*T(i,j-1)+an*T(i-1,j)+as*T(i+1,j))/ap;

%ABSOLUTE ERROR calculation at every point
error=abs(T(i,j)-dummy(i,j));
```

```
%For identifying and replacing the max error in domain
if error > mxe
    mxe=error;
end

end

end

dummy=T;

%Number of iterations calculation
itr=itr+1;

%Storing change of temperature of random points in different matrices
p1(itr+1,1)=T(m1,k1);
p2(itr+1,1)=T(m2,k2);
p3(itr+1,1)=T(m3,k3);

end


%============================== OUTPUT ======================================

fprintf('\n\n---------------------- OUTPUT ----------------------\n\n');
fprintf('Number of iterations for initial guess T = %dC are %d\n\n',T_ini,itr);

figure(1)

plot([1:itr+1],p1);
title('Random point temperature vs iterations')
hold on

plot([1:itr+1],p2);
hold on

plot([1:itr+1],p3);
hold off
xlabel('Iterations');
ylabel('Temperature');

legend(['p1 (',num2str(x(round(sx/3))),',',num2str(y(round(sy/3))),')'],['p1
(',num2str(x(round(sx/2))),',',num2str(y(round(sy/2))),')'],['p1
(',num2str(x(round(2*sx/3))),',',num2str(y(round(2*sy/3))),')'],'Location','South')
grid on


%---------PART D OF THE ASSIGNMENT----------

%Give temperature contour values as the output.

figure(2)
%Showing values of some isotherms
colormap jet
[C,q] = contour(flipud(T),5);
clabel(C)
grid on
title('Values of contours')
pbaspect([1 h/l 1])

%Plot the temperature contours as the output.
```

```
figure(3)
colormap jet
contourf(x,y,flipud(T),10)
colorbar
title('Graph of equilibrium temperature of the domain')
hold on
%r taken as 1 for plotting 2D grid
r=ones(sy,sx);
%As mesh generation is not part of the exercise artificial mesh is created
%Grid transperency can be djusted by changing EdgeAlpha value
mesh(x,y,r,'FaceAlpha',.1,'EdgeAlpha',.4,'FaceColor','white','EdgeColor','white');
%pbaspect defines aspect ratio for the graph
pbaspect([1 h/l 1])
hold off


%_____________________________________________________________________________________
```

Results and Discussion

A. Data values at specific grid points

Initial guess of temperature for the domain: 10 ^{o}C

Minimum convergence error: 10^-6

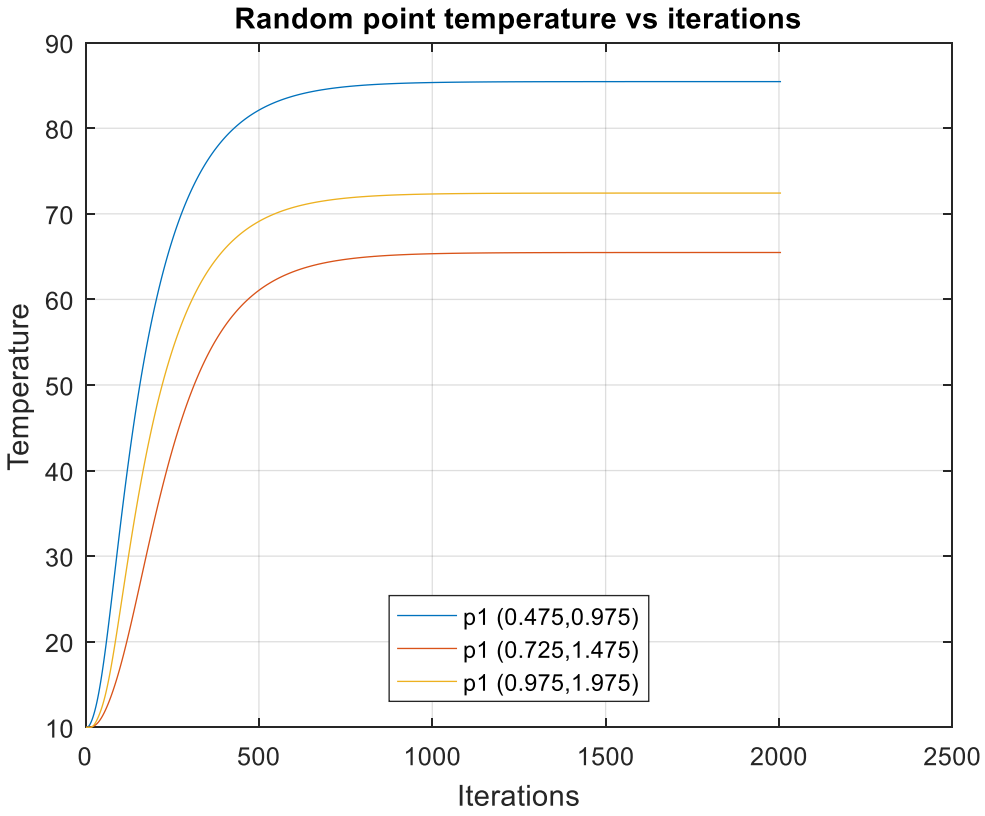


Fig 5. Variation of temperature with Iterations

B. Trend of iterations with initial guess value of temperature

Results are tabulated for minimum convergence error of 10^-6

Table 3

T_initial (^{o}C)	Iterations	T_initial (^{o}C)	Iterations	T_initial (^{o}C)	Iterations
10	2005	110	1772	210	2055
20	1986	120	1840	220	2066
30	1965	130	1886	230	2078
40	1940	140	1921	240	2088
50	1909	150	1949	250	2097
60	1871	160	1973	260	2106
70	1819	170	1993	270	2115
80	1737	180	2011	280	2123
90	1530	190	2027	290	2131
100	1640	200	2041	300	2138

B. Trend of iterations with initial guess value of temperature

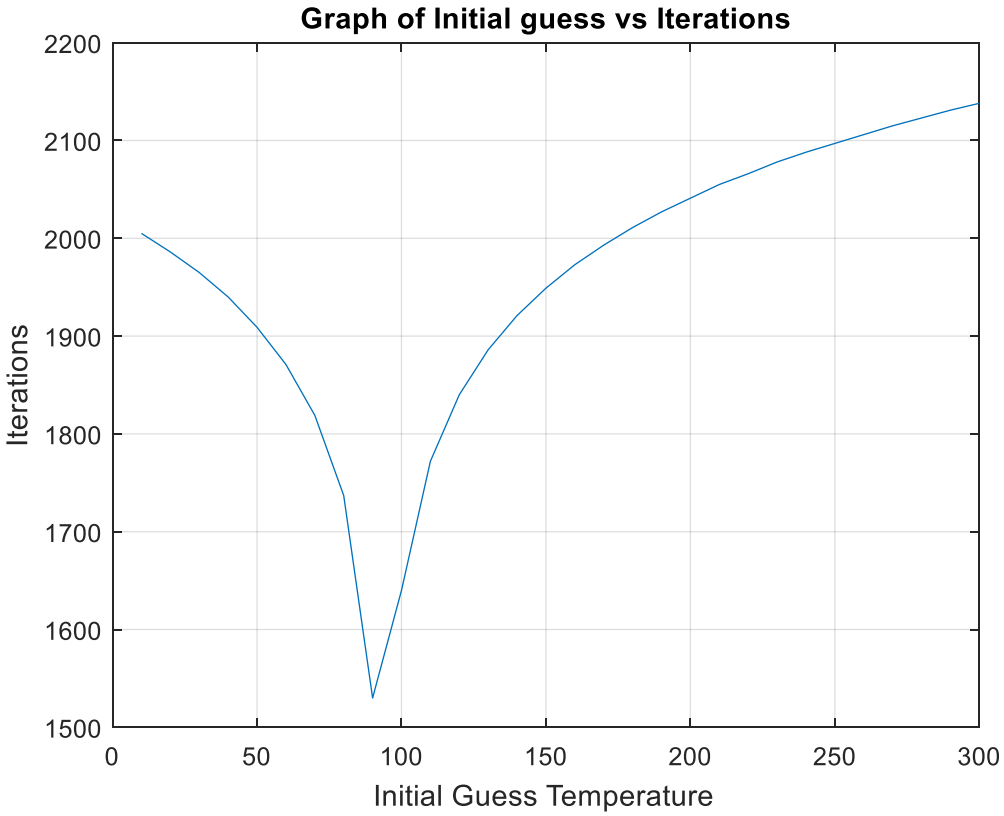


Fig 6. Variation of Iterations with initial guess value

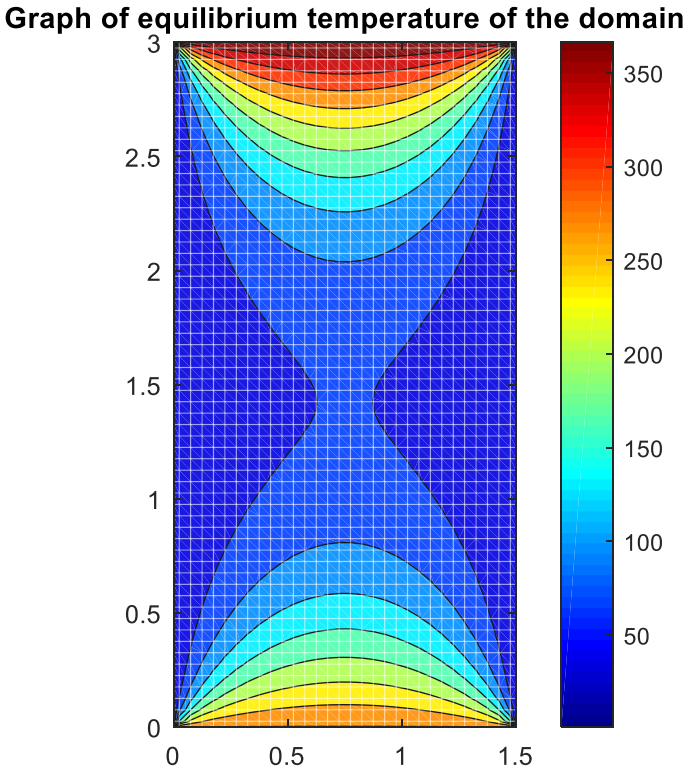
From the temperature data obtained at the end of the iteration, average temperature was calculated. The temperature turned out to be 118.2007 ^{o}C. It can be confirmed from Fig.1 that the iterations are minimum in the range of 80 ^{o}C to 120 ^{o}C. Drop in the number of iterations can be because of the range of the guessed value is closer to the mean temperature of the domain.

C. Convergence criteria and the implementation

Absolute error formulation was used for defining convergence. The convergence limit is taken as the input from the user in 10^ (- some integer value). While loop is implemented till the convergence is reached. Maximum error in the domain is matched with the convergence limit for accurate results.

$$Error^{k} = \left| T_{new}^{k} - T_{old}^{k} \right|$$

D. Output



========== Clay brick coulumn problem ==========

---------- INPUT ----------

Geometry Parameters -->

Length of column in m: 1.5
Height of column in m: 3
Grid size along length in m: 0.05
Grid size along height in m: 0.05

Boundary Conditions -->

Boundary condition at north boundary in degree C: 400
Boundary condition at south boundary in degree C: 300
Boundary condition at east boundary in degree C: 30
Boundary condition at west boundary in degree C: 30

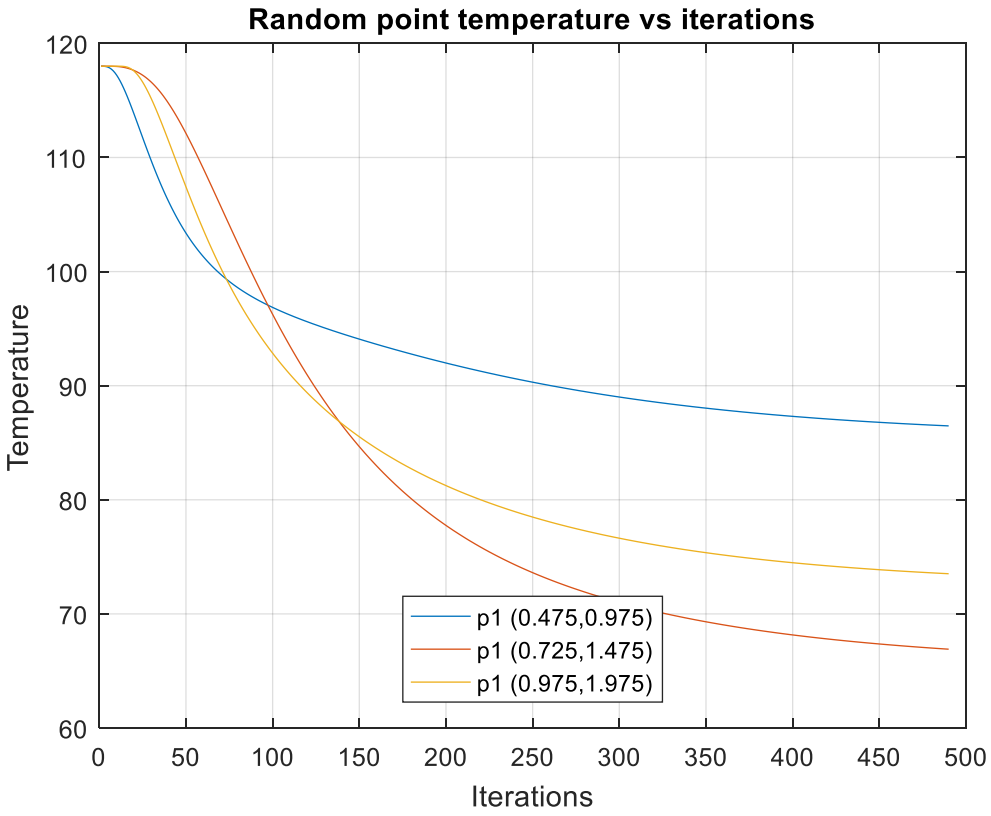
Initial guess of temperature for the domain: 10

Enter minimum convergence error: 10^-6

---------- OUTPUT ----------

Number of iterations for initial guess T = 10C are 2005

D. Output



========== Clay brick column problem ==========

---------- INPUT ----------

Geometry Parameters -->

Length of column in m: 1.5
Height of column in m: 3
Grid size along length in m: 0.05
Grid size along height in m: 0.05

Boundary Conditions -->

Boundary condition at north boundary in degree C: 400
Boundary condition at south boundary in degree C: 300
Boundary condition at east boundary in degree C: 30
Boundary condition at west boundary in degree C: 30

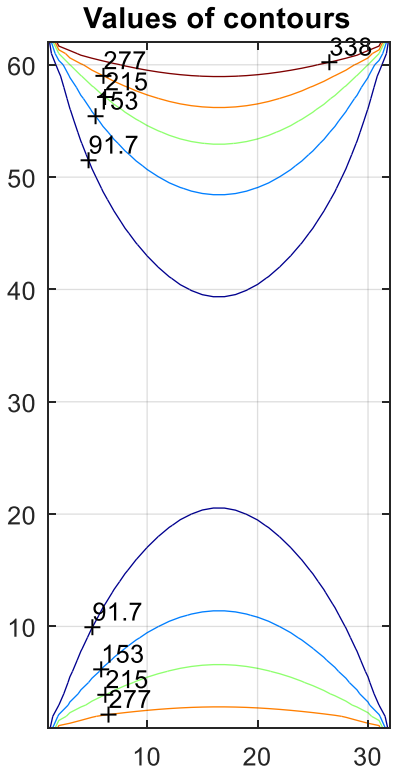
Initial guess of temperature for the domain: 118

Enter minimum convergence error: 10^-2

---------- OUTPUT ----------

Number of iterations for initial guess T = 118C are 489

D. Output



========== Clay brick column problem ==========

---------- INPUT ----------

Geometry Parameters -->

Length of column in m: 1.5
Height of column in m: 3
Grid size along length in m: 0.05
Grid size along height in m: 0.05

Boundary Conditions -->

Boundary condition at north boundary in degree C: 400
Boundary condition at south boundary in degree C: 300
Boundary condition at east boundary in degree C: 30
Boundary condition at west boundary in degree C: 30

Initial guess of temperature for the domain: 118

Enter minimum convergence error: 10^-1

---------- OUTPUT ----------

Number of iterations for initial guess T = 118C are 206