PS3Q1 $\frac{d}{dx}(\rho u \phi) = \frac{d}{dx}\left(\Gamma \frac{d\phi}{dx}\right)$

$$\frac{\rho u}{\Gamma} \frac{d\phi}{dx} = \frac{d^2\phi}{dx^2}$$

let $x' = \frac{x}{L} \implies 0 \le x' \le 1$

$$x = Lx'$$

$$\frac{\rho u}{\Gamma} \frac{d\phi}{d(Lx')} = \frac{d^2\phi}{d(Lx')^2}$$

multiply through by L^2

$$\frac{\rho u L}{\Gamma} \frac{d\phi}{dx'} = \frac{d^2\phi}{dx'^2} \qquad \phi(0) = 1, \ \phi(1) = 0$$

$$Pe \frac{d\phi}{dx'} = \frac{d^2\phi}{dx'^2}$$

W w P e E

$$\sum_f \left(\frac{d\phi}{dx} - Pe\phi\right) \cdot \hat{n}_f A_f = 0$$

$$= \left(\frac{d\phi}{dx}\bigg|_w - Pe\phi_w\right) A_w - \left(\frac{d\phi}{dx}\bigg|_e - Pe\phi_e\right) A_e$$

$A_e = A_w$

$$\frac{d\phi}{dx}\bigg|_w - \frac{d\phi}{dx}\bigg|_e - Pe\phi_w + Pe\phi_e = 0$$

let $\phi_e = \frac{1+\alpha_e}{2}\phi_P + \frac{1-\alpha_e}{2}\phi_E$

PS3 Q1 cont'd

let $\phi_w = \left(\frac{1+\alpha_w}{2}\right)\phi_W + \frac{(1-\alpha_w)}{2}\phi_P$

recall that $P_\Delta = \frac{\rho u \Delta x}{\Gamma}$, $\therefore$ all α will be the same

$$\frac{\phi_P-\phi_W}{\Delta x} - \frac{\phi_E-\phi_P}{\Delta x} - Pe\frac{(1+\alpha)}{2}\phi_W - Pe\frac{(1-\alpha)}{2}\phi_P + Pe\frac{(1+\alpha)}{2}\phi_P + Pe\left(\frac{1-\alpha}{2}\right)\phi_E = 0$$

At interior points

$$\left[\frac{1}{\Delta x} + \frac{1}{\Delta x} - Pe\frac{(1-\alpha)}{2} + Pe\frac{(1+\alpha)}{2}\right]\phi_P + \left[-\frac{1}{\Delta x} - Pe\left(\frac{1+\alpha}{2}\right)\right]\phi_W + \left[-\frac{1}{\Delta x} + Pe\left(\frac{1-\alpha}{2}\right)\right]\phi_E = 0$$

At $x'=0$, $\phi=1$ $\Rightarrow \frac{\phi_W+\phi_P}{2} = 1 \Rightarrow \phi_W = 2-\phi_P$

At the interior points $a_P\phi_P + a_W\phi_W + a_E\phi_E = 0$

now $a_P\phi_P + a_W(2-\phi_P) + a_E\phi_E = 0$

$$\boxed{(a_P-a_W)\phi_P + a_E\phi_E = -2a_W}$$

at $x'=1$, $\phi=0$ $\quad \frac{\phi_P+\phi_E}{2} = 0 \Rightarrow \phi_P = -\phi_E$

$$a_P\phi_P + a_W\phi_W - a_E\phi_P = 0$$

$$\boxed{(a_P-a_E)\phi_P + a_W\phi_W = 0}$$

PS3 Q1 cont'd. Solving analytically:

$$\frac{d^2\phi}{dx^2} = Pe \frac{d\phi}{dx}$$

let $\psi = \frac{d\phi}{dx} \Rightarrow \frac{d\psi}{dx} = Pe\,\psi$

$$\frac{1}{\psi} d\psi = Pe\,dx \Rightarrow \ln\psi = Pe\,x + C$$

$$\psi = C_1 e^{Pe\,x}$$

$$\phi = \int \psi\,dx = \frac{1}{Pe} C e^{Pe\,x} + D$$

$$\phi(x) = \frac{C}{Pe} e^{Pe\,x} + D$$

$\underline{\phi(0) = 1}$ $\qquad\qquad$ $\phi(1) = 0$

$\phi(0) = \frac{C}{Pe} + D = 1$ $\qquad\qquad$ $\phi(1) = \frac{C}{Pe} e^{Pe} + D$

Solve for C and D

$$C = \frac{-Pe}{e^{Pe} - 1}, \quad D = 1 + \frac{1}{e^{Pe} - 1}$$

$$\phi(x) = \left(\frac{-1}{e^{Pe} - 1}\right) e^{Pe\,x} + \frac{1}{e^{Pe} - 1} + 1$$

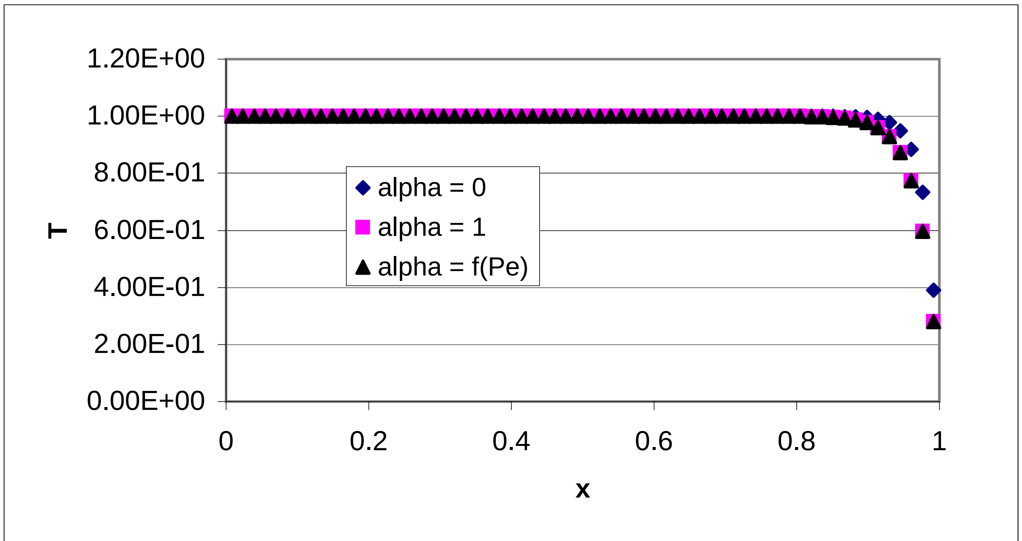
```
program PS3_Q1a
!***************Begin Header*************************************************
!This program was written by Dr. Seth Dworkin on February 17, 2010.
!This program solves problem of 1D convection/diffusion and
!Dirichlet boundary conditions using finite volumes
!**************************End Header*****************************************

!Variable declaration
implicit none
integer, parameter :: maxcvs=128
double precision, parameter :: pi = 3.14159265358979323846264338327950d0
double precision, dimension(maxcvs) :: e,f,g,b,T,x,T_exact,diff
integer :: numcvs,i,ii,osc
double precision :: length,ap,aw,ae,Pe,delx,alpha,L

!Variable initialization
Length = 1.d0
Pe = 50.d0
alpha = 0.d0  !Pe**2/(Pe**2+5.d0)
numcvs = 128

!do ii=1,32
!calculate the length of each cv
delx = Length/numcvs

!define the distance to the center of each c.v.
x(1) = delx/2.d0
do i=2,numcvs
    x(i) = x(i-1) + delx
end do

!determine the standard coefficients ap, aw, and ae
ap = (2.d0/delx) + Pe*alpha
aw = -1.d0/delx - Pe*(1.d0+alpha)/2.d0
ae = -1.d0/delx + Pe*(1.d0-alpha)/2.d0

!impose the west side boundary condition
e(1) = 0.d0
f(1) = ap-aw
g(1) = ae
b(1) = -2*aw

!impose the east side boundary condition
e(numcvs) = aw
f(numcvs) = ap-ae
g(numcvs) = 0.d0
b(numcvs) = 0.d0

!Enter the tridiagonal matrix A and rhs vector f to be solved
do i=2,numcvs-1
    e(i) = aw
    f(i) = ap
    g(i) = ae
    b(i) = 0.d0
end do

!solve the tridiagonal linear system using the Thomas algorithm
```

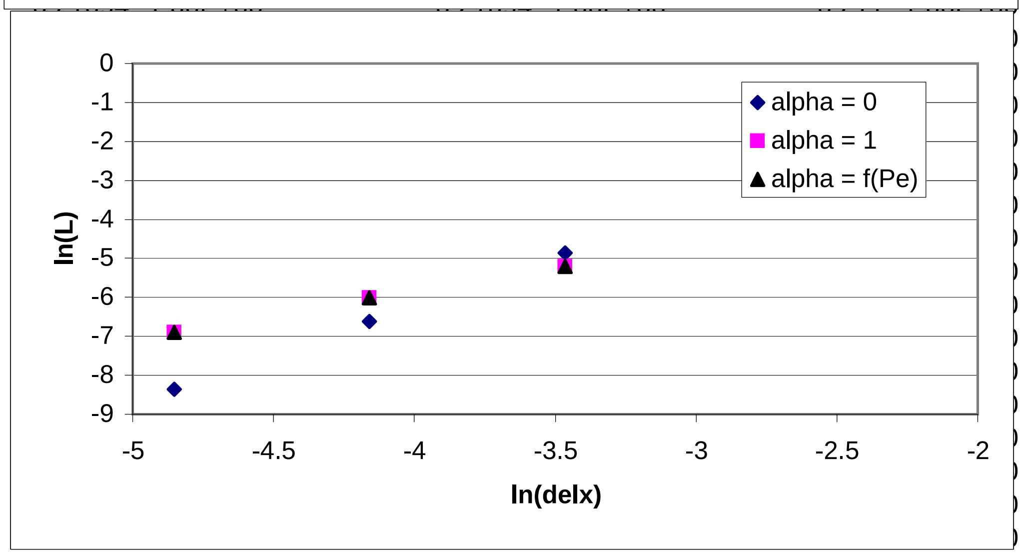
```
call Thomas(numcvs,e,f,g,b,T)

!Calculate the three errors, L1, L2, and L_inf
L = 0.d0
do i=1,numcvs
    T_exact(i) = (1.d0/(exp(Pe)-1.d0))*(1.d0-exp(Pe*x(i))) + 1.d0
    diff(i) = T_exact(i)-T(i)
    L = L + (T(i)-T_exact(i))**2
end do
L = dsqrt(L)/numcvs

!check for oscillations
osc = 0
do i=1,numcvs-1
    if(T(i+1)>T(i))then
    osc = osc + 1
    endif
end do

!if(numcvs==64)then
!write out the solution vector and error norms
write(*,*)' Solution vector:'
write(*,*)'  x(i)       T(i)      T_exact      Diff'
do i=1,numcvs
    write(*,10)x(i),T(i),T_exact(i),T_exact(i)-T(i)
end do
!endif
write(*,*)

write(*,*)'For N = ',numcvs
write(*,12) L
!numcvs=numcvs+1
!if(osc.le.1)stop
!end do

10 format(1x,F7.5,4x,F7.5,4x,F7.5,4x,F9.5)
12 format(' L = ',E17.5)
end program PS3_Q1a

!*******************************************************************
Subroutine Thomas(numcvs,e,f,g,b,x)
integer :: numcvs
double precision, dimension(numcvs) :: e,f,g,b,x
integer :: ii,jj,kk
!This subroutine solves a linear system using the thomas algorithm

!Part 1 (decomposition)
do kk=2,numcvs
    e(kk) = e(kk)/f(kk-1)
    f(kk) = f(kk) - e(kk)*g(kk-1)
end do

!Part 2 (forward substitution)
do kk=2,numcvs
    b(kk) = b(kk) - e(kk)*b(kk-1)
end do
```

```
!Part 2 (backward substitution)
x(numcvs)=b(numcvs)/f(numcvs)
do kk=numcvs-1,1,-1
    x(kk) = (b(kk) - g(kk)*x(kk+1))/f(kk)
end do

End Subroutine Thomas
```

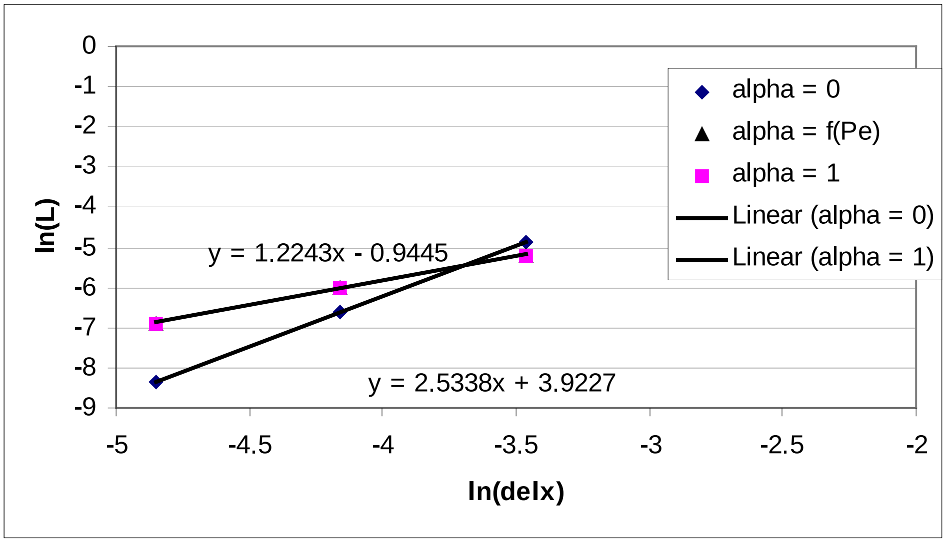
		alpha		
delx	N	0	1	function
0.03125	32	7.80E-03	5.52E-03	5.50E-03
0.015625	64	1.33E-03	2.45E-03	2.44E-03
0.007813	128	2.32E-04	1.01E-03	1.01E-03

	ln(L)		
ln(deltx)	0	1	function
-3.46574	-4.85416	-5.19965	-5.20295
-4.15888	-6.62393	-6.01236	-6.01494
-4.85203	-8.36671	-6.89691	-6.89919





1b



The two equations are depicted on the graph.
They intercept at approx ln(delx) = -3.75 (delx = 0.0235) and Pe_delta = 1.18.

```
program PS3_Q1c
!****************Begin Header*****************************************************
!This program was written by Dr. Seth Dworkin on February 17, 2010.
!This program solves problem of 1D convection/diffusion and
!Dirichlet boundary conditions using finite volumes
!This program solves PS3 Q1c and determines the minimum number of grid points
!for which the solution no longer oscillates
!****************************End Header*****************************************

!Variable declaration
implicit none
integer, parameter :: maxcvs=128
double precision, parameter :: pi = 3.14159265358979323846264338327950d0
double precision, dimension(maxcvs) :: e,f,g,b,T,x,T_exact,diff
integer :: numcvs,i,ii,osc
double precision :: length,ap,aw,ae,Pe,delx,alpha,L

!Variable initialization
Length = 1.d0
Pe = 50.d0
alpha = 0.d0  !Pe**2/(Pe**2+5.d0)
numcvs = 8

do ii=1,32
!calculate the length of each cv
delx = Length/numcvs

!define the distance to the center of each c.v.
x(1) = delx/2.d0
do i=2,numcvs
    x(i) = x(i-1) + delx
end do

!determine the standard coefficients ap, aw, and ae
ap = (2.d0/delx) + Pe*alpha
aw = -1.d0/delx - Pe*(1.d0+alpha)/2.d0
ae = -1.d0/delx + Pe*(1.d0-alpha)/2.d0

!impose the west side boundary condition
e(1) = 0.d0
f(1) = ap-aw
g(1) = ae
b(1) = -2*aw

!impose the east side boundary condition
e(numcvs) = aw
f(numcvs) = ap-ae
g(numcvs) = 0.d0
b(numcvs) = 0.d0

!Enter the tridiagonal matrix A and rhs vector f to be solved
do i=2,numcvs-1
    e(i) = aw
    f(i) = ap
    g(i) = ae
    b(i) = 0.d0
end do
```

```
!solve the tridiagonal linear system using the Thomas algorithm
call Thomas(numcvs,e,f,g,b,T)

!Calculate the three errors, L1, L2, and L_inf
L = 0.d0
do i=1,numcvs
    T_exact(i) = (1.d0/(exp(Pe)-1.d0))*(1.d0-exp(Pe*x(i))) + 1.d0
    diff(i) = T_exact(i)-T(i)
    L = L + (T(i)-T_exact(i))**2
end do
L = dsqrt(L)/numcvs

!check for oscillations
osc = 0
do i=1,numcvs-1
    if(T(i+1)>T(i))then
    osc = osc + 1
    endif
end do

!if(numcvs==64)then
!write out the solution vector and error norms
write(*,*)' Solution vector:'
write(*,*)'  x(i)       T(i)      T_exact      Diff'
do i=1,numcvs
    write(*,10)x(i),T(i),T_exact(i),T_exact(i)-T(i)
end do
!endif
write(*,*)

write(*,*)'For N = ',numcvs
write(*,12) L
numcvs=numcvs+1
if(osc.le.1)then
    write(*,*)'Pe_delta = ',Pe/numcvs
    stop
endif
end do




10 format(1x,F7.5,4x,F7.5,4x,F7.5,4x,F9.5)
12 format(' L = ',E17.5)
end program PS3_Q1c

!*******************************************************************
Subroutine Thomas(numcvs,e,f,g,b,x)
integer :: numcvs
double precision, dimension(numcvs) :: e,f,g,b,x
integer :: ii,jj,kk
!This subroutine solves a linear system using the thomas algorithm

!Part 1 (decomposition)
do kk=2,numcvs
    e(kk) = e(kk)/f(kk-1)
```

```
    f(kk) = f(kk) - e(kk)*g(kk-1)
end do

!Part 2 (forward substitution)
do kk=2,numcvs
    b(kk) = b(kk) - e(kk)*b(kk-1)
end do

!Part 2 (backward substitution)
x(numcvs)=b(numcvs)/f(numcvs)
do kk=numcvs-1,1,-1
    x(kk) = (b(kk) - g(kk)*x(kk+1))/f(kk)
end do

End Subroutine Thomas
```

Output

```
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.06250    0.98442    1.00000      0.01558
0.18750    1.04490    1.00000     -0.04490
0.31250    0.92750    1.00000      0.07250
0.43750    1.15539    1.00000     -0.15539
0.56250    0.71302    1.00000      0.28698
0.68750    1.57173    1.00000     -0.57173
0.81250    -.09517    0.99992      1.09508
0.93750    3.14058    0.95606     -2.18451

For N =            8
L =       0.31653E+00
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.05556    1.00314    1.00000     -0.00314
0.16667    0.98979    1.00000      0.01021
0.27778    1.01816    1.00000     -0.01816
0.38889    0.95789    1.00000      0.04211
0.50000    1.08596    1.00000     -0.08596
0.61111    0.81381    1.00000      0.18619
0.72222    1.39212    1.00000     -0.39212
0.83333    0.16321    0.99976      0.83655
0.94444    2.77464    0.93782     -1.83681

For N =            9
L =       0.22965E+00
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.05000    0.99948    1.00000      0.00052
0.15000    1.00192    1.00000     -0.00192
0.25000    0.99622    1.00000      0.00378
0.35000    1.00951    1.00000     -0.00951
0.45000    0.97852    1.00000      0.02148
0.55000    1.05082    1.00000     -0.05082
0.65000    0.88211    1.00000      0.11789
0.75000    1.27578    1.00000     -0.27578
0.85000    0.35722    0.99945      0.64223
0.95000    2.50052    0.91792     -1.58261

For N =           10
L =       0.17350E+00
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.04545    1.00007    1.00000     -0.00007
0.13636    0.99971    1.00000      0.00029
0.22727    1.00064    1.00000     -0.00064
0.31818    0.99826    1.00000      0.00174
0.40909    1.00437    1.00000     -0.00437
0.50000    0.98865    1.00000      0.01135
0.59091    1.02908    1.00000     -0.02908
0.68182    0.92512    1.00000      0.07488
0.77273    1.19244    0.99999     -0.19246
0.86364    0.50503    0.99891      0.49387
0.95455    2.27266    0.89697     -1.37569

For N =           11
L =       0.13423E+00
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.04167    0.99999    1.00000      0.00001
0.12500    1.00003    1.00000     -0.00003
0.20833    0.99992    1.00000      0.00008
0.29167    1.00026    1.00000     -0.00026
0.37500    0.99929    1.00000      0.00071
0.45833    1.00204    1.00000     -0.00204
0.54167    0.99420    1.00000      0.00580
0.62500    1.01651    1.00000     -0.01651
0.70833    0.95302    1.00000      0.04698
0.79167    1.13374    0.99997     -0.13377
0.87500    0.61937    0.99807      0.37870
0.95833    2.08334    0.87549     -1.20786

For N =           12
```

```
 L =       0.10616E+00
  Solution vector:
   x(i)       T(i)      T_exact      Diff
 0.03846    1.00000    1.00000      0.00000
 0.11538    1.00000    1.00000      0.00000
 0.19231    1.00001    1.00000     -0.00001
 0.26923    0.99997    1.00000      0.00003
 0.34615    1.00009    1.00000     -0.00009
 0.42308    0.99971    1.00000      0.00029
 0.50000    1.00092    1.00000     -0.00092
 0.57692    0.99710    1.00000      0.00290
 0.65385    1.00918    1.00000     -0.00918
 0.73077    0.97093    1.00000      0.02907
 0.80769    1.09205    0.99993     -0.09212
 0.88462    0.70850    0.99688      0.28838
 0.96154    1.92308    0.85384     -1.06923

 For N =           13
 L =       0.85514E-01
  Solution vector:
   x(i)       T(i)      T_exact      Diff
 0.03571    1.00000    1.00000      0.00000
 0.10714    1.00000    1.00000      0.00000
 0.17857    1.00000    1.00000      0.00000
 0.25000    1.00000    1.00000      0.00000
 0.32143    0.99999    1.00000      0.00001
 0.39286    1.00003    1.00000     -0.00003
 0.46429    0.99989    1.00000      0.00011
 0.53571    1.00040    1.00000     -0.00040
 0.60714    0.99860    1.00000      0.00140
 0.67857    1.00497    1.00000     -0.00497
 0.75000    0.98237    1.00000      0.01763
 0.82143    1.06251    0.99987     -0.06264
 0.89286    0.77839    0.99529      0.21690
 0.96429    1.78571    0.83232     -0.95339

 For N =           14
 L =       0.69995E-01
Press any key to continue
  Solution vector:
   x(i)       T(i)      T_exact      Diff
 0.03333    1.00000    1.00000      0.00000
 0.10000    1.00000    1.00000      0.00000
 0.16667    1.00000    1.00000      0.00000
 0.23333    1.00000    1.00000      0.00000
 0.30000    1.00000    1.00000      0.00000
 0.36667    1.00000    1.00000      0.00000
 0.43333    1.00001    1.00000     -0.00001
 0.50000    0.99996    1.00000      0.00004
 0.56667    1.00016    1.00000     -0.00016
 0.63333    0.99935    1.00000      0.00065
 0.70000    1.00260    1.00000     -0.00260
 0.76667    0.98958    0.99999      0.01041
 0.83333    1.04167    0.99976     -0.04191
 0.90000    0.83333    0.99326      0.15993
 0.96667    1.66667    0.81112     -0.85554

 For N =           15
 L =       0.58096E-01
  Solution vector:
   x(i)       T(i)      T_exact      Diff
 0.03125    1.00000    1.00000      0.00000
 0.09375    1.00000    1.00000      0.00000
 0.15625    1.00000    1.00000      0.00000
 0.21875    1.00000    1.00000      0.00000
 0.28125    1.00000    1.00000      0.00000
 0.34375    1.00000    1.00000      0.00000
 0.40625    1.00000    1.00000      0.00000
 0.46875    1.00000    1.00000      0.00000
 0.53125    0.99999    1.00000      0.00001
 0.59375    1.00006    1.00000     -0.00006
 0.65625    0.99971    1.00000      0.00029
 0.71875    1.00131    1.00000     -0.00131
 0.78125    0.99405    0.99998      0.00593
 0.84375    1.02710    0.99960     -0.02751
 0.90625    0.87652    0.99079      0.11427
```

```
0.96875    1.56250    0.79039     -0.77211

For N =           16
L =       0.48814E-01
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.02941    1.00000    1.00000      0.00000
0.08824    1.00000    1.00000      0.00000
0.14706    1.00000    1.00000      0.00000
0.20588    1.00000    1.00000      0.00000
0.26471    1.00000    1.00000      0.00000
0.32353    1.00000    1.00000      0.00000
0.38235    1.00000    1.00000      0.00000
0.44118    1.00000    1.00000      0.00000
0.50000    1.00000    1.00000      0.00000
0.55882    1.00000    1.00000      0.00000
0.61765    1.00002    1.00000     -0.00002
0.67647    0.99988    1.00000      0.00012
0.73529    1.00062    1.00000     -0.00062
0.79412    0.99675    0.99997      0.00322
0.85294    1.01707    0.99936     -0.01771
0.91176    0.91036    0.98787      0.07750
0.97059    1.47059    0.77021     -0.70038

For N =           17
L =       0.41464E-01
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.02778    1.00000    1.00000      0.00000
0.08333    1.00000    1.00000      0.00000
0.13889    1.00000    1.00000      0.00000
0.19444    1.00000    1.00000      0.00000
0.25000    1.00000    1.00000      0.00000
0.30556    1.00000    1.00000      0.00000
0.36111    1.00000    1.00000      0.00000
0.41667    1.00000    1.00000      0.00000
0.47222    1.00000    1.00000      0.00000
0.52778    1.00000    1.00000      0.00000
0.58333    1.00000    1.00000      0.00000
0.63889    1.00001    1.00000     -0.00001
0.69444    0.99996    1.00000      0.00004
0.75000    1.00027    1.00000     -0.00028
0.80556    0.99832    0.99994      0.00162
0.86111    1.01031    0.99904     -0.01127
0.91667    0.93669    0.98450      0.04780
0.97222    1.38889    0.75065     -0.63824

For N =           18
L =       0.35563E-01
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.02632    1.00000    1.00000      0.00000
0.07895    1.00000    1.00000      0.00000
0.13158    1.00000    1.00000      0.00000
0.18421    1.00000    1.00000      0.00000
0.23684    1.00000    1.00000      0.00000
0.28947    1.00000    1.00000      0.00000
0.34211    1.00000    1.00000      0.00000
0.39474    1.00000    1.00000      0.00000
0.44737    1.00000    1.00000      0.00000
0.50000    1.00000    1.00000      0.00000
0.55263    1.00000    1.00000      0.00000
0.60526    1.00000    1.00000      0.00000
0.65789    1.00000    1.00000      0.00000
0.71053    0.99999    1.00000      0.00001
0.76316    1.00011    0.99999     -0.00012
0.81579    0.99920    0.99990      0.00070
0.86842    1.00587    0.99861     -0.00726
0.92105    0.95694    0.98069      0.02376
0.97368    1.31579    0.73174     -0.58405

For N =           19
L =       0.30767E-01
 Solution vector:
  x(i)       T(i)       T_exact      Diff
0.02500    1.00000    1.00000      0.00000
```

```
0.07500    1.00000    1.00000      0.00000
0.12500    1.00000    1.00000      0.00000
0.17500    1.00000    1.00000      0.00000
0.22500    1.00000    1.00000      0.00000
0.27500    1.00000    1.00000      0.00000
0.32500    1.00000    1.00000      0.00000
0.37500    1.00000    1.00000      0.00000
0.42500    1.00000    1.00000      0.00000
0.47500    1.00000    1.00000      0.00000
0.52500    1.00000    1.00000      0.00000
0.57500    1.00000    1.00000      0.00000
0.62500    1.00000    1.00000      0.00000
0.67500    1.00000    1.00000      0.00000
0.72500    1.00000    1.00000      0.00000
0.77500    1.00004    0.99999     -0.00005
0.82500    0.99966    0.99984      0.00018
0.87500    1.00309    0.99807     -0.00502
0.92500    0.97222    0.97648      0.00426
0.97500    1.25000    0.71350     -0.53650

For N =            20
L =        0.26827E-01
 Solution vector:
  x(i)       T(i)      T_exact      Diff
0.02381    1.00000    1.00000      0.00000
0.07143    1.00000    1.00000      0.00000
0.11905    1.00000    1.00000      0.00000
0.16667    1.00000    1.00000      0.00000
0.21429    1.00000    1.00000      0.00000
0.26190    1.00000    1.00000      0.00000
0.30952    1.00000    1.00000      0.00000
0.35714    1.00000    1.00000      0.00000
0.40476    1.00000    1.00000      0.00000
0.45238    1.00000    1.00000      0.00000
0.50000    1.00000    1.00000      0.00000
0.54762    1.00000    1.00000      0.00000
0.59524    1.00000    1.00000      0.00000
0.64286    1.00000    1.00000      0.00000
0.69048    1.00000    1.00000      0.00000
0.73810    1.00000    1.00000      0.00000
0.78571    1.00001    0.99998     -0.00003
0.83333    0.99987    0.99976     -0.00012
0.88095    1.00144    0.99740     -0.00404
0.92857    0.98344    0.97188     -0.01155
0.97619    1.19048    0.69592     -0.49455

For N =            21
L =        0.23557E-01
 Solution vector:
  x(i)       T(i)      T_exact      Diff
0.02273    1.00000    1.00000      0.00000
0.06818    1.00000    1.00000      0.00000
0.11364    1.00000    1.00000      0.00000
0.15909    1.00000    1.00000      0.00000
0.20455    1.00000    1.00000      0.00000
0.25000    1.00000    1.00000      0.00000
0.29545    1.00000    1.00000      0.00000
0.34091    1.00000    1.00000      0.00000
0.38636    1.00000    1.00000      0.00000
0.43182    1.00000    1.00000      0.00000
0.47727    1.00000    1.00000      0.00000
0.52273    1.00000    1.00000      0.00000
0.56818    1.00000    1.00000      0.00000
0.61364    1.00000    1.00000      0.00000
0.65909    1.00000    1.00000      0.00000
0.70455    1.00000    1.00000      0.00000
0.75000    1.00000    1.00000      0.00000
0.79545    1.00000    0.99996     -0.00004
0.84091    0.99996    0.99965     -0.00032
0.88636    1.00056    0.99659     -0.00396
0.93182    0.99130    0.96693     -0.02437
0.97727    1.13636    0.67902     -0.45735

For N =            22
L =        0.20819E-01
 Solution vector:
```

x(i)	T(i)	T_exact	Diff
0.02174	1.00000	1.00000	0.00000
0.06522	1.00000	1.00000	0.00000
0.10870	1.00000	1.00000	0.00000
0.15217	1.00000	1.00000	0.00000
0.19565	1.00000	1.00000	0.00000
0.23913	1.00000	1.00000	0.00000
0.28261	1.00000	1.00000	0.00000
0.32609	1.00000	1.00000	0.00000
0.36957	1.00000	1.00000	0.00000
0.41304	1.00000	1.00000	0.00000
0.45652	1.00000	1.00000	0.00000
0.50000	1.00000	1.00000	0.00000
0.54348	1.00000	1.00000	0.00000
0.58696	1.00000	1.00000	0.00000
0.63043	1.00000	1.00000	0.00000
0.67391	1.00000	1.00000	0.00000
0.71739	1.00000	1.00000	0.00000
0.76087	1.00000	0.99999	-0.00001
0.80435	1.00000	0.99994	-0.00006
0.84783	0.99999	0.99950	-0.00049
0.89130	1.00015	0.99564	-0.00451
0.93478	0.99638	0.96164	-0.03473
0.97826	1.08696	0.66276	-0.42420

For N = 23
L = 0.18506E-01
Solution vector:

x(i)	T(i)	T_exact	Diff
0.02083	1.00000	1.00000	0.00000
0.06250	1.00000	1.00000	0.00000
0.10417	1.00000	1.00000	0.00000
0.14583	1.00000	1.00000	0.00000
0.18750	1.00000	1.00000	0.00000
0.22917	1.00000	1.00000	0.00000
0.27083	1.00000	1.00000	0.00000
0.31250	1.00000	1.00000	0.00000
0.35417	1.00000	1.00000	0.00000
0.39583	1.00000	1.00000	0.00000
0.43750	1.00000	1.00000	0.00000
0.47917	1.00000	1.00000	0.00000
0.52083	1.00000	1.00000	0.00000
0.56250	1.00000	1.00000	0.00000
0.60417	1.00000	1.00000	0.00000
0.64583	1.00000	1.00000	0.00000
0.68750	1.00000	1.00000	0.00000
0.72917	1.00000	1.00000	0.00000
0.77083	1.00000	0.99999	-0.00001
0.81250	1.00000	0.99992	-0.00008
0.85417	1.00000	0.99932	-0.00068
0.89583	1.00002	0.99453	-0.00549
0.93750	0.99915	0.95606	-0.04309
0.97917	1.04167	0.64713	-0.39453

For N = 24
L = 0.16538E-01
Solution vector:

x(i)	T(i)	T_exact	Diff
0.02000	1.00000	1.00000	0.00000
0.06000	1.00000	1.00000	0.00000
0.10000	1.00000	1.00000	0.00000
0.14000	1.00000	1.00000	0.00000
0.18000	1.00000	1.00000	0.00000
0.22000	1.00000	1.00000	0.00000
0.26000	1.00000	1.00000	0.00000
0.30000	1.00000	1.00000	0.00000
0.34000	1.00000	1.00000	0.00000
0.38000	1.00000	1.00000	0.00000
0.42000	1.00000	1.00000	0.00000
0.46000	1.00000	1.00000	0.00000
0.50000	1.00000	1.00000	0.00000
0.54000	1.00000	1.00000	0.00000
0.58000	1.00000	1.00000	0.00000
0.62000	1.00000	1.00000	0.00000
0.66000	1.00000	1.00000	0.00000
0.70000	1.00000	1.00000	0.00000

```
 0.74000    1.00000    1.00000      0.00000
 0.78000    1.00000    0.99998     -0.00002
 0.82000    1.00000    0.99988     -0.00012
 0.86000    1.00000    0.99909     -0.00091
 0.90000    1.00000    0.99326     -0.00674
 0.94000    1.00000    0.95021     -0.04979
 0.98000    1.00000    0.63212     -0.36788

 For N =           25
 L =       0.14852E-01
 Pe_delta =    1.92307692307692
Press any key to continue
```

$$\rho C_p \frac{dT}{dt} = \lambda \nabla \cdot \nabla T + S$$

$$\frac{dT}{dt} = \frac{\lambda}{\rho C_p} \nabla \cdot \nabla T + \frac{S}{\rho C_p}$$

$$\int_t \int_{V} \frac{dT}{dt}\, dV = \alpha \int_t \int_A \nabla T \cdot \hat{n}\, dA\, dt + \int_t \int_{V} \frac{S}{\rho C_p}\, dV\, dt$$

since our CV is 2D

$$\int_t \int_A \frac{dT}{dt}\, dt\, dx\, dy = \alpha \int_t \int_S \nabla T \cdot \hat{n}\, ds\, dt + \int_t \int_A \frac{S}{\rho C_p}\, dA\, dt$$

$$\left(T^{n+1} - T^n\right) dx\, dy = \Delta t \alpha \left[\frac{dT}{dx}\bigg|_e \Delta y - \frac{dT}{dx}\bigg|_w \Delta y + \frac{dT}{dy}\bigg|_n \Delta x - \frac{dT}{dy} \Delta x \right]$$

$$+ \frac{\bar{S}}{\rho C_p} \Delta x \Delta y \Delta t$$

$\frac{\bar{S} \Delta x \Delta y \Delta t}{\rho C_p}$ can be written as $S_1 + S_2 T_p^{n+1}$

$$\frac{2 h_{side} \Delta x \Delta y T_\infty}{\Delta x \Delta y \text{ thickness}} \frac{\Delta x \Delta y \Delta t}{\rho C_p} = S_1 = \frac{2 h_{side} T_\infty \Delta x \Delta y \Delta t}{\text{thick } \rho C_p}$$

$$S_2 = -\frac{2 h_{side} \Delta x \Delta y \Delta t}{\text{thick } \rho C_p}$$

$$T^{n+1}\Delta x \Delta y - T^{n}\Delta x \Delta y = \alpha \Delta t \frac{T_E - T_P}{\Delta x}\Delta y - \alpha \Delta t \frac{T_P - T_W}{\Delta x}\Delta y$$

$$+ \alpha \Delta t \frac{T_N - T_P}{\Delta y}\Delta x - \alpha \Delta t \frac{T_P - T_S}{\Delta y}\Delta x$$

$$+ S_1 + S_2 T_P^{n+1}$$

$$\left[\Delta A + \frac{\alpha \Delta t \cancel{\Delta y}}{\cancel{\Delta x}} + \frac{\alpha \Delta t \cancel{\Delta y}}{\cancel{\Delta x}} + \frac{\alpha \Delta t \cancel{\Delta x}}{\cancel{\Delta y}} + \frac{\alpha \Delta t \cancel{\Delta x}}{\cancel{\Delta y}} - S_2\right] T_P^{n+1}$$

$$- \frac{\alpha \Delta t \cancel{\Delta y}}{\cancel{\Delta x}} T_E - \frac{\alpha \Delta t \cancel{\Delta y}}{\cancel{\Delta x}} T_W - \frac{\alpha \Delta t \cancel{\Delta x}}{\cancel{\Delta y}} T_N - \frac{\alpha \Delta t \cancel{\Delta x}}{\cancel{\Delta y}} T_S$$

$$= S_1 + T^{n}\Delta x \Delta y$$

PS3Q2 cont'd

rewrite the discretized eqn at interior pts as

(1) $$A_P T_P^{n+1} + A_E T_E^{n+1} + A_N T_N^{n+1} + A_W T_W^{n+1} + A_S T_S^{n+1} = b$$

at $x=0$, $$T_W = \frac{hT_\infty - \left[\frac{-\lambda}{\Delta x} + \frac{h}{2}\right] T_P}{\left[\frac{\lambda}{\Delta x} + \frac{h}{2}\right]} \quad (2)$$

at $x = 5cm$: $$\lambda\left[\frac{T_E - T_P}{\Delta x}\right] = h\left(T_\infty - \left(\frac{T_E + T_P}{2}\right)\right)$$

$$\left[\frac{\lambda}{\Delta x} + \frac{h}{2}\right] T_E + \left[-\frac{\lambda}{\Delta x} + \frac{h}{2}\right] T_P = hT_\infty$$

$$T_E = \frac{hT_\infty - \left[\frac{-\lambda}{\Delta x} + \frac{h}{2}\right] T_P}{\left[\frac{\lambda}{\Delta x} + \frac{h}{2}\right]} \quad (3)$$

at $y=0$: $$\frac{T_S + T_P}{2} = 150 + 273.15$$

$$T_S = 846.3 - T_P \quad (4)$$

at $y = 5cm$:

$$\lambda\left[\frac{T_N - T_P}{\Delta y}\right] = h\left(T_\infty - \left(\frac{T_N + T_P}{2}\right)\right)$$

$$\left[\frac{\lambda}{\Delta y} + \frac{h}{2}\right] T_N + \left[\frac{-\lambda}{\Delta y} + \frac{h}{2}\right] T_P = hT_\infty$$

PS3 Q2 cont'd

$$T_N = \frac{h T_\infty - \left[\frac{-\lambda}{\Delta y} + \frac{h}{2}\right] T_P}{\left[\frac{\lambda}{\Delta x} + \frac{h}{2}\right]} \quad (4)$$

Note that since $\Delta x = \Delta y$, (2), (3), and (5) can be written as

$$(2) \quad T_W = C_1 + C_2 T_P$$
$$(3) \quad T_E = C_1 + C_2 T_P$$
$$(5) \quad T_N = C_1 + C_2 T_P$$

where $C_1 = \dfrac{h T_\infty}{\left[\frac{\lambda}{\Delta x} + \frac{h}{2}\right]}$ and $C_2 = -\dfrac{\left[\frac{-\lambda}{\Delta y} + \frac{h}{2}\right]}{\left[\frac{\lambda}{\Delta x} + \frac{h}{2}\right]}$

For the source term

$$\bar{S} = \frac{2hA(T_\infty - T_P^{n+1})}{V} \qquad \frac{A}{V} = \frac{1}{\text{thickness}}$$

$$\bar{S} = \frac{2h(T_\infty - T_P^{n+1})}{0.0035}$$

$$\int s\,dV \approx \bar{S} \times (\text{thickness} \times \text{del}x \times \text{del}y)$$

$$\bar{S} = 2h\Delta x \Delta y T_\infty - 2h\Delta x \Delta y T_P^{n+1}$$

---

for ind = 1, sub (2) and (4) into (1)

$$A_P T_P + A_E T_E + A_N T_N + A_W(C_1 + C_2 T_P) + A_S(706.3 - T_P) = b$$

$$(A_P + A_W C_2 - A_S) T_P + A_E T_E + A_N T_N = b - A_W C_1 - A_S 706.3$$

PS3 Q2 cont'd

along the bottom sub (4) into (1)

$$A_P T_P + A_E T_E + A_N T_N + A_W T_W + A_S(706.3 - T_P) = b$$

$$(A_P - A_S)T_P + A_S T_E + A_N T_N + A_W T_W = b - A_S 706.3$$

---

at bottom right, sub (3) and (4) into (1)

$$A_P T_P + A_E(C_1 + C_2 T_P) + A_N T_N + A_W T_W + A_S(706.3 - T_P) = b$$

$$(A_P + A_E C_2 - A_S)T_P + A_N T_N + A_W T_W = b - A_E C_1 - A_S 706.3$$

---

left side: sub (2) into (1)

$$A_P T_P + A_E T_E + A_N T_N + A_W(C_1 + C_2 T_P) + A_S T_S = b$$

$$(A_P + A_W C_2)T_P + A_E T_E + A_N T_N + A_S T_S = b - A_W C_1$$

---

right side: sub (3) into (1)

$$(A_P + A_E C_2)T_P + A_W T_W + A_N T_N + A_S T_S = b - A_E C_1$$

---

upper left

$$(A_P + A_W C_2 + A_N C_2)T_P + A_S T_S + A_E T_E = b - A_W C_1 - A_N C_1$$

---

upper row

$$(A_P + A_N C_2)T_P + A_S T_S + A_E T_E + A_W T_W = b - A_n C_1$$

---

upper right

$$(A_P + A_E C_2 + A_N C_2)T_P + A_S T_S + A_W T_W = b - A_E C_1 - A_N C_1$$

```
program PS3_Q2
!***************Begin Header*****************************************************
!This program was written by Dr. Seth Dworkin on February 19, 2010.
!This program solves the problem of 2D diffusion in a heat sink
!using finite volumes
!***************************End Header*****************************************

!Variable declaration
implicit none
integer, parameter :: NX = 80
integer, parameter :: NY = 80
double precision, parameter :: pi = 3.14159265358979323846264338327950d0
double precision, parameter :: rho = 1716.d0 !Kg/m^3
double precision, parameter :: cp = 4817.d0 !J/kg*K
double precision, parameter :: lambda = 14.6d0 !W/m*K
double precision, parameter :: T_CPU = 423.15d0 !K
double precision, parameter :: h = 472.d0 !W/m^2*K
double precision, parameter :: hside = 36.4d0 !W/m^2*K
double precision, parameter :: Tinf = 298.15d0  !K
double precision, parameter :: thickness = 0.0035d0  !m
double precision, dimension(NX*NY) :: e,f,g,b,Tn,Tnp1,x,y
double precision, dimension(NX*NY) :: ld,ud  !lower and upper diagonal of the lin sys
integer :: i,j,ii,iii,numcvs,ind
double precision :: length,height,alpha,Ap,Ae,Aw,An,As,T_center_np1,T_center_n
double precision :: delx,dely,delA,delV,delt,S1,S2,C1,C2,soldiff

!open a file to write the computed solutions to
open(UNIT=21,STATUS='replace',FORM='FORMATTED',FILE='T_plate.dat')

!Variable initialization
Length = 0.05d0 !meters
height = 0.05d0 !meters
delt = 0.01d0    !seconds
numcvs = NX*NY
alpha = lambda/(rho*cp)     !in units of m^2/s

!calculate the length of each cv
delx = Length/NX

!calculate the height of each cv
dely = height/NY

!calculate the area of each cv
delA = delx*dely

!calculate the volume of each cv
delV = delA*thickness

!define the distance to the center of each c.v.
x(1) = delx/2.d0
do i=2,NX
    x(i) = x(i-1) + delx
end do

!define the distance to the center of each c.v.
y(1) = dely/2.d0
do i=2,NY
    y(i) = y(i-1) + dely
end do

!set an initial guess
do ind=1,numcvs
    Tn(ind) = 298.15d0
end do

!set the constant parameters of the problem
S1 = 2.d0*hside*Tinf*delA*delt/(rho*cp*thickness) !2.d0*hside*delA*Tinf/(rho*cp)
S2 = -2.d0*hside*delA*delt/(rho*cp*thickness)  !-2.d0*hside*delA/(rho*cp)
Ap = delA + 4.d0*alpha*delt - S2
Ae = -alpha*delt
Aw = -alpha*delt
An = -alpha*delt
As = -alpha*delt
C1 = h*Tinf/((lambda/delx) + (h/2.d0))
C2 = -((h/2.d0) - (lambda/delx))/((lambda/delx) + (h/2.d0))
```

```
!initialize a timestep number counter
ii=0

!enter the main timestepping loop
222 continue
ii=ii+1

!consider the bottom left hand point (ind = 1)
ind = 1
ld(ind) = 0.d0
e(ind) = 0.d0
f(ind) = Ap + Aw*C2 - As
g(ind) = Ae
ud(ind) = An
b(ind) = (S1 + Tn(ind)*delA) - Aw*C1 - As*2.d0*T_CPU

!consider the points along the bottom between the corners
do ind = 2,NX-1
    ld(ind) = 0.d0
    e(ind) = Aw
    f(ind) = Ap - As
    g(ind) = Ae
    ud(ind) = An
    b(ind) = (S1 + Tn(ind)*delA) - As*2.d0*T_CPU
end do

!consider the bottom left hand point (ind = 1)
ind = NX
ld(ind) = 0.d0
e(ind) = Aw
f(ind) = Ap + Ae*C2 - As
g(ind) = 0.d0
ud(ind) = An
b(ind) = (S1 + Tn(ind)*delA) - Ae*C1 - As*2.d0*T_CPU

!consider the points along left side between the corners
do j = 2,NY-1
    ind = (j-1)*NX+1
    ld(ind) = As
    e(ind) = 0.d0
    f(ind) = Ap + Aw*C2
    g(ind) = Ae
    ud(ind) = An
    b(ind) = (S1 + Tn(ind)*delA) - Aw*C1
end do

!consider the interior points
do j = 2,NY-1
    do i = 2,NX-1
        ind = (j-1)*NX+i
        ld(ind) = As
        e(ind) = Aw
        f(ind) = Ap
        g(ind) = Ae
        ud(ind) = An
        b(ind) = (S1 + Tn(ind)*delA)
    end do
end do

!consider the points along right side between the corners
do j = 2,NY-1
    ind = j*NX
    ld(ind) = As
    e(ind) = Aw
    f(ind) = Ap + Ae*C2
    g(ind) = 0.d0
    ud(ind) = An
    b(ind) = (S1 + Tn(ind)*delA) - Ae*C1
end do

!consider the upper left hand point
ind = (NY-1)*NX + 1
ld(ind) = As
e(ind) = 0.d0
f(ind) = Ap + An*C2 + Aw*C2
```

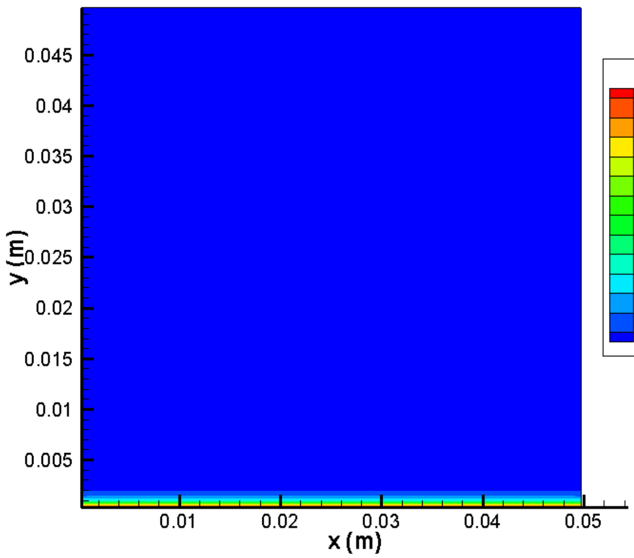
```
    g(ind) = Ae
    ud(ind) = 0.d0
    b(ind) = (S1 + Tn(ind)*delA) - An*C1 - Aw*C1

    !consider the upper points between the corners
    do i = 2,NX-1
        ind = (NY-1)*NX + i
        ld(ind) = As
        e(ind) = Aw
        f(ind) = Ap + An*C2
        g(ind) = Ae
        ud(ind) = 0.d0
        b(ind) = (S1 + Tn(ind)*delA) - An*C1
    end do

    !consider the upper right hand point
    ind = NY*NX
    ld(ind) = As
    e(ind) = Aw
    f(ind) = Ap + An*C2 + Ae*C2
    g(ind) = 0.d0
    ud(ind) = 0.d0
    b(ind) = (S1 + Tn(ind)*delA) - An*C1 - Ae*C1

    !solve the tridiagonal linear system using the Thomas algorithm
    call P_Bi_CGSTAB(NX,numcvs,e,f,g,b,ud,ld,Tnp1)

    !check the stop criteria
    soldiff=0.d0
    do j=1,NY
        do i=1,NX
        ind=(j-1)*NX+1
            soldiff = soldiff + (Tn(ind) - Tnp1(ind))**2
        end do
    end do
    soldiff = dsqrt(soldiff)/numcvs

    write(*,*)delt*ii,soldiff

    !the ss value in the centre of the plate is first calculated as 352K
    !since it starts at 298.15 K
    T_center_n = 0.d0
    T_center_np1 = 0.d0
    do j=40,41
        do i=40,41
        ind=(j-1)*NX+1
            T_center_n = T_center_n + Tn(ind)
            T_center_np1 = T_center_np1 + Tnp1(ind)
        end do
    end do
    T_center_n = T_center_n/4.d0
    T_center_np1 = T_center_np1/4.d0

    if (T_center_n.lt.((375.7+298.15)/2.d0) .and. T_center_np1.gt.((375.7+298.15)/2.d0))then
    write(*,*)'Centre half time = ',delt*ii
    endif

    !update the solution so that the next timestep can proceed
    do ind=1,numcvs
        Tn(ind) = Tnp1(ind)
    end do

    !move to the next time step
    if(soldiff.gt.5d-10)goto 222

write(*,*)'Solution Complete'

write(21,*)'TITLE ="Temp"'
write(21,34)'VARIABLES ="x (m)","y (m)","T"'
write(21,*) ' ZONE T=', '"' ,'Species', '"' , ',', ' i =', NX,',' , ' J =', NY,  ',' ,' F =POINT'
DO j=1,NY
    DO i=1,NX
        ind=(j-1)*NX+i
        write(21,33)x(i),y(j),Tnp1(ind)
    enddo
enddo
```

```
33  FORMAT(1X,1P120E13.4)
34  FORMAT(1X,A4000)
CLOSE(21)

write(*,*)
!write(*,*)'Temp profile at j=5'
!j=5
!do i=1,NX
!   ind=(j-1)*NX+i
!   write(*,*)x(i),Tnp1(ind)
!end do

10 format(F9.1,4x,F6.2,1x,F6.2,1x,F6.2,1x,F6.2,1x,F6.2,1x,F6.2,1x,F6.2,1x,F6.2)
11 format(F14.8,4x,F13.4)
end program PS3_Q2

!*****************************************************************
Subroutine P_Bi_CGSTAB(NX,numcvs,e,f,g,b,ud,ld,x)
integer :: numcvs,NX
double precision, dimension(numcvs) :: e,f,g,b,x,r,v,p,r_bar,vim1,pim1,rim1,xim1,ld,ud
double precision, dimension(numcvs) :: y,K,z,s,t,kinvs,kinvt,esym,fsym,gsym,bsym
integer :: ii,jj,kk
double precision :: errsum,rho,alpha,omega,CGTOL,rhoim1,omegaim1,beta
!This subroutine solves a tri-diagonal linear system using preconditioned Bi-CGSTAB
!The algorithm was taken from
!Van Der Vorst Siam J. Sci. Stat. Comput. 13 (1992) 631-644
!and was adapted for a nonsymmetric tridiagonal linear system

!Set the Bi-CGSTAB tolerance
CGTOL = 1.d-9

!Generate the initial guess
do ii=1,numcvs
    xim1(ii)=b(ii)/f(ii)
end do

!define the preconditioner as the diagonal of A
do ii=1,numcvs
    K(ii)=f(ii)
end do

!calculate r0
ii=1
rim1(ii)=b(ii)-ud(ii)*xim1(ii+NX)-f(ii)*xim1(ii)-g(ii)*xim1(ii+1)
do ii=2,NX
    rim1(ii)=b(ii)-ud(ii)*xim1(ii+NX)-e(ii)*xim1(ii-1)-f(ii)*xim1(ii)-g(ii)*xim1(ii+1)
end do
do ii=NX+1,numcvs-NX
    rim1(ii)=b(ii)-ld(ii)*xim1(ii-NX)-ud(ii)*xim1(ii+NX)-e(ii)*xim1(ii-1)-f(ii)*xim1(ii)-g(ii)*xim1(ii+1)
end do
do ii=numcvs-NX+1,numcvs-1
    rim1(ii)=b(ii)-ld(ii)*xim1(ii-NX)-e(ii)*xim1(ii-1)-f(ii)*xim1(ii)-g(ii)*xim1(ii+1)
end do
ii=numcvs
rim1(ii)=b(ii)-ld(ii)*xim1(ii-NX)-e(ii)*xim1(ii-1)-f(ii)*xim1(ii)

!set r0_bar equal to r0
do ii=1,numcvs
    r_bar(ii)=rim1(ii)
end do

!initialize the constants
kk=0
rhoim1 = 1.d0
alpha = 1.d0
omegaim1 = 1.d0
do ii=1,numcvs
    vim1(ii) = 0.d0
    pim1(ii) = 0.d0
end do

!begin main loop
errsum=1.d0
2001 continue
if(errsum.gt.CGTOL)then
```

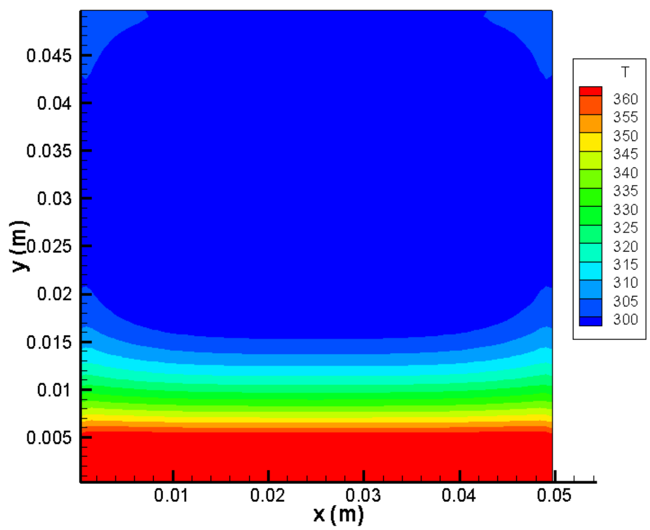
```
kk=kk+1
!check the progress
if(mod(kk,1000)==0)write(*,*)' BiCGSTAB itn = ',kk,' Resid = ',errsum

rho=dotprod(numcvs,rim1,r_bar)
beta=rho*alpha/(rhoim1*omegaim1)
do ii=1,numcvs
    p(ii)=rim1(ii)+beta*(pim1(ii)-omegaim1*vim1(ii))
end do

!solve for y from Ky=pi
do ii=1,numcvs
    y(ii)=p(ii)/K(ii)
end do

!perform the matrix vector multiplication vi = Ay
ii=1
v(ii) = f(ii)*y(ii) + g(ii)*y(ii+1) + ud(ii)*y(ii+NX)
do ii=2,NX
    v(ii) = e(ii)*y(ii-1) + f(ii)*y(ii) + g(ii)*y(ii+1) + ud(ii)*y(ii+NX)
end do
do ii=NX+1,numcvs-NX
    v(ii) = e(ii)*y(ii-1) + f(ii)*y(ii) + g(ii)*y(ii+1) + ld(ii)*y(ii-NX) + ud(ii)*y(ii+NX)
end do
do ii=numcvs-NX+1,numcvs-1
    v(ii) = e(ii)*y(ii-1) + f(ii)*y(ii) + g(ii)*y(ii+1) + ld(ii)*y(ii-NX)
end do
ii=numcvs
v(ii) = e(ii)*y(ii-1) + f(ii)*y(ii) + ld(ii)*y(ii-NX)

alpha=rho/dotprod(numcvs,r_bar,v)
do ii=1,numcvs
    s(ii)=rim1(ii)-alpha*v(ii)
end do
!solve for z from Kz=s
do ii=1,numcvs
    z(ii)=s(ii)/K(ii)
end do
!perform the matrix vector multiplication t = Az
ii=1
t(ii)=f(ii)*z(ii)+g(ii)*z(ii+1)+ud(ii)*z(ii+NX)
do ii=2,NX
    t(ii)=e(ii)*z(ii-1)+f(ii)*z(ii)+g(ii)*z(ii+1)+ud(ii)*z(ii+NX)
end do
do ii=NX+1,numcvs-NX
    t(ii)=e(ii)*z(ii-1)+f(ii)*z(ii)+g(ii)*z(ii+1)+ld(ii)*z(ii-NX)+ud(ii)*z(ii+NX)
end do
do ii=numcvs-NX+1,numcvs-1
    t(ii)=e(ii)*z(ii-1)+f(ii)*z(ii)+g(ii)*z(ii+1)+ld(ii)*z(ii-NX)
end do
ii=numcvs
t(ii)=e(ii)*z(ii-1)+f(ii)*z(ii)+ld(ii)*z(ii-NX)

do ii=1,numcvs
    kinvs(ii)=s(ii)/K(ii)
    kinvt(ii)=t(ii)/K(ii)
end do
omega = (dotprod(numcvs,kinvt,kinvs))/(dotprod(numcvs,kinvt,kinvt))
do ii=1,numcvs
    x(ii)=xim1(ii)+alpha*y(ii)+omega*z(ii)
end do
do ii=1,numcvs
    r(ii)=s(ii)-omega*t(ii)
end do

!calculate the remaining error
errsum=0
ii=1
errsum=errsum+(b(ii)-f(ii)*x(ii)-g(ii)*x(ii+1)-ud(ii)*x(ii+NX))**2
do ii=2,NX
    errsum=errsum+(b(ii)-e(ii)*x(ii-1)-f(ii)*x(ii)-g(ii)*x(ii+1)-ud(ii)*x(ii+NX))**2
end do
do ii=NX+1,numcvs-NX
    errsum=errsum+(b(ii)-e(ii)*x(ii-1)-f(ii)*x(ii)-g(ii)*x(ii+1)-ld(ii)*x(ii-NX)-ud(ii)*x(ii+NX))**2
end do
do ii=numcvs-NX+1,numcvs-1
```

```
        errsum=errsum+(b(ii)-e(ii)*x(ii-1)-f(ii)*x(ii)-g(ii)*x(ii+1)-ld(ii)*x(ii-NX))**2
    end do
    ii=numcvs
    errsum=errsum+(b(ii)-e(ii)*x(ii-1)-f(ii)*x(ii)-ld(ii)*x(ii-NX))**2
    errsum = dsqrt(errsum)/numcvs

    !update all of the variables to prepare for the next iteration
    rhoim1 = rho
    omegaim1 = omega
    do ii=1,numcvs
        rim1(ii)=r(ii)
        vim1(ii)=v(ii)
        pim1(ii)=p(ii)
        xim1(ii)=x(ii)
    end do

    goto 2001
else
    return
endif

12 format(1x,i5,3x,E17.5E3)
End Subroutine P_Bi_CGSTAB

!*****************************************************************
double precision function dotprod(n,a,b)
!this function calculates the dot product of two vectors a and b, of a given length n
integer :: n,i
double precision, dimension(n) :: a,b

dotprod = 0.d0
do i=1,n
    dotprod = dotprod + a(i)*b(i)
end do
end function dotprod
```

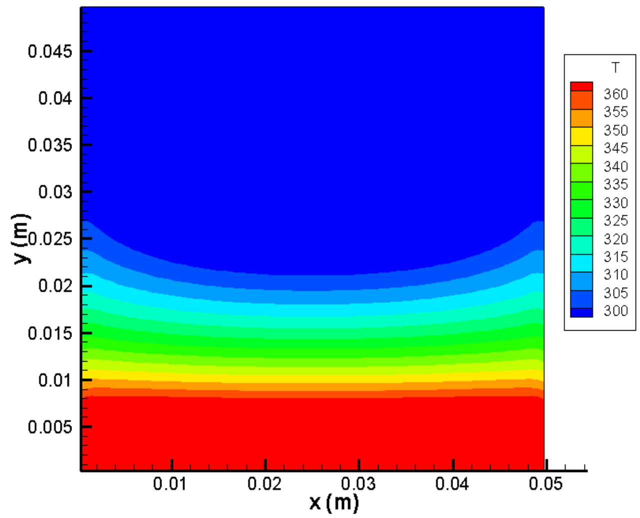
After the stop criteria has been met, the temp distribution in the plate at 0.12s looks like:

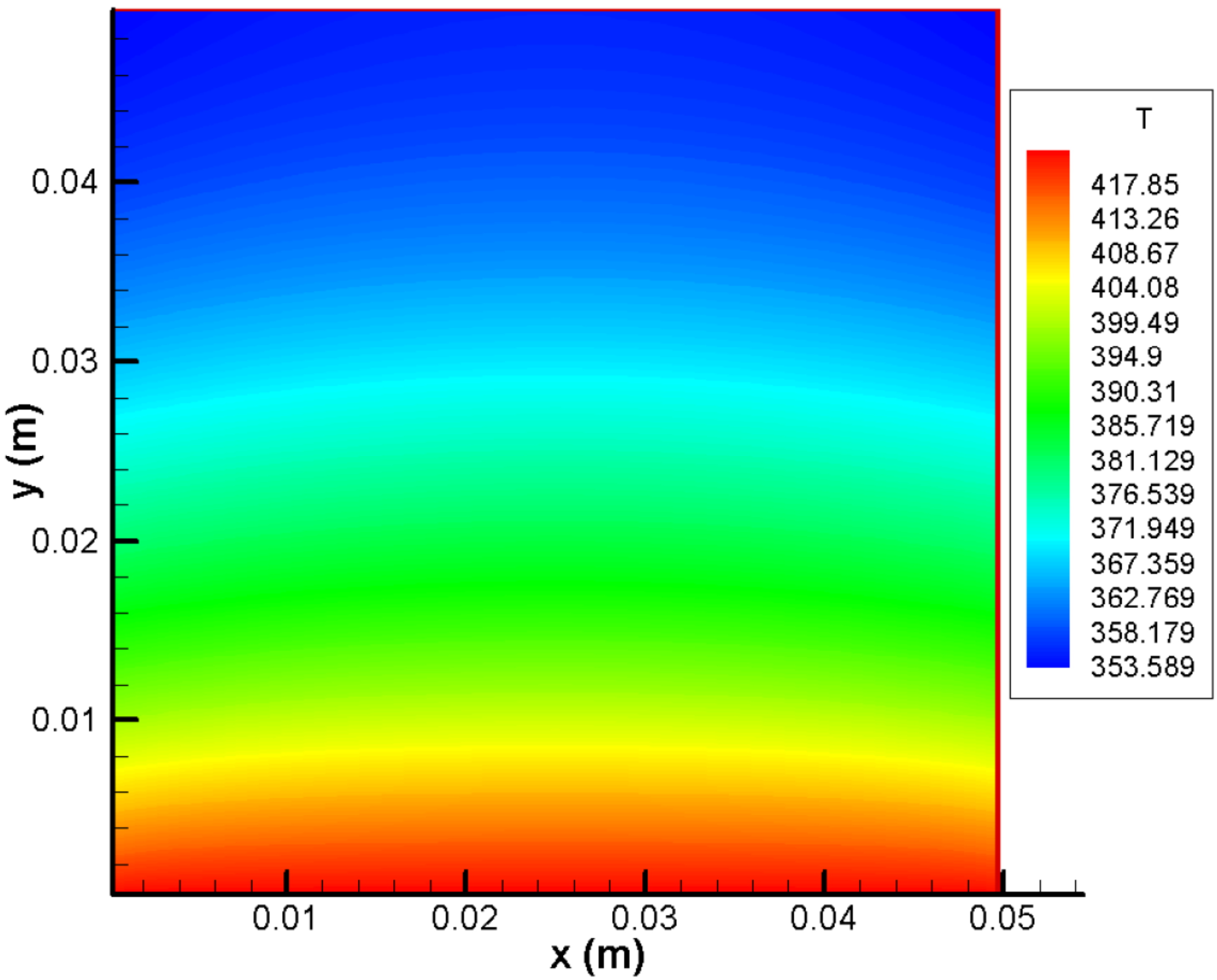


However, the distribution using a stop criteria of 5e-5 is



The distribution at steady state using a stop criteria of 5e-8 (the same with 5e-10) is:




T
417.85
413.26
408.67
404.08
399.49
394.9
390.31
385.719
381.129
376.539
371.949
367.359
362.769
358.179
353.589
0.04
0.03
0.02
0.01
y (m)
0.01
0.02
0.03
0.04
0.05
x (m)