Indian Institute of Technology, Madras

Foundations of Computational Fluid Dynamics - AM5630

Steady 2D Convection Diffusion

Kanak Agarwal

November 22, 2025

1 Introduction

The two dimensional convection diffusion equation i.e., the 2D transport equation for temperature is given by,

$$\frac{\partial}{\partial x}(\rho UT) + \frac{\partial}{\partial x}(\rho VT) = \frac{\partial}{\partial x}\left(\Gamma \frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(\Gamma \frac{\partial T}{\partial y}\right) + S \tag{1}$$

where $\Gamma = k/c_p$. Discretizing the above equation as implemented in [1] and taking a control volume as depicted in Fig. 1,

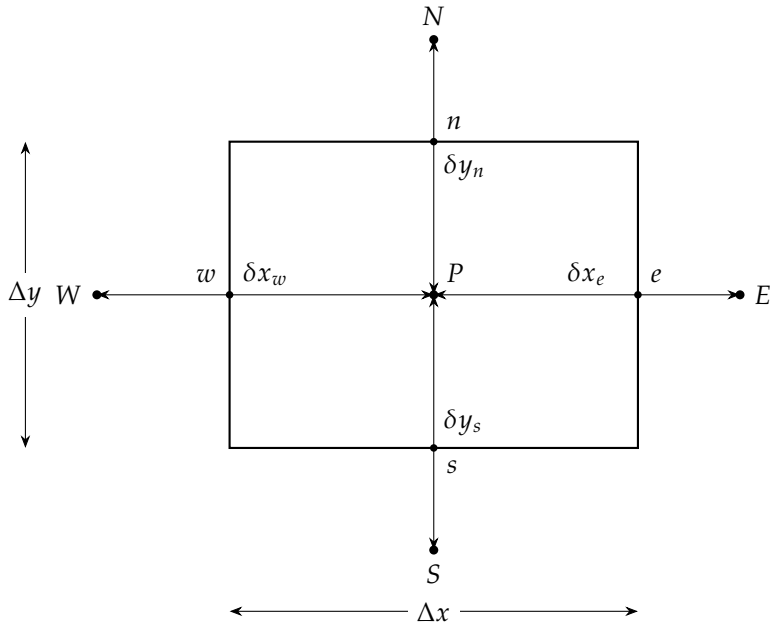


Figure 1: Typical Control Volume in Two-Dimensional Finite Volume Discretization

$$a_P T_P = a_E T_E + a_W T_W + a_N T_N + a_S T_S \tag{2}$$

where,

$$a_P = a_W + a_E + a_S + a_N + \Delta F \tag{3}$$

and,

$$a_W = max\left[F_w, \left(D_w + \frac{F_w}{2}\right), 0\right], \quad a_E = max\left[-F_e, \left(D_e - \frac{F_e}{2}\right), 0\right] \tag{4}$$

$$a_S = max\left[F_s, \left(D_s + \frac{F_s}{2}\right), 0\right], \quad a_N = max\left[-F_n, \left(D_n - \frac{F_n}{2}\right), 0\right] \tag{5}$$

$$\Delta F = F_e - F_w + F_n - F_s \tag{6}$$

$$F_w = (\rho u)_w \Delta y, \quad F_e = (\rho u)_e \Delta y, \quad F_n = (\rho u)_n \Delta x, \quad F_s = (\rho u)_s \Delta x \tag{7}$$

$$D_w = \frac{\Gamma_w}{\delta x_w}\Delta y, \quad D_e = \frac{\Gamma_e}{\delta x_e}\Delta y, \quad D_n = \frac{\Gamma_n}{\delta y_n}\Delta x, \quad D_s = \frac{\Gamma_s}{\delta y_s}\Delta x \tag{8}$$

2 Code Architecture

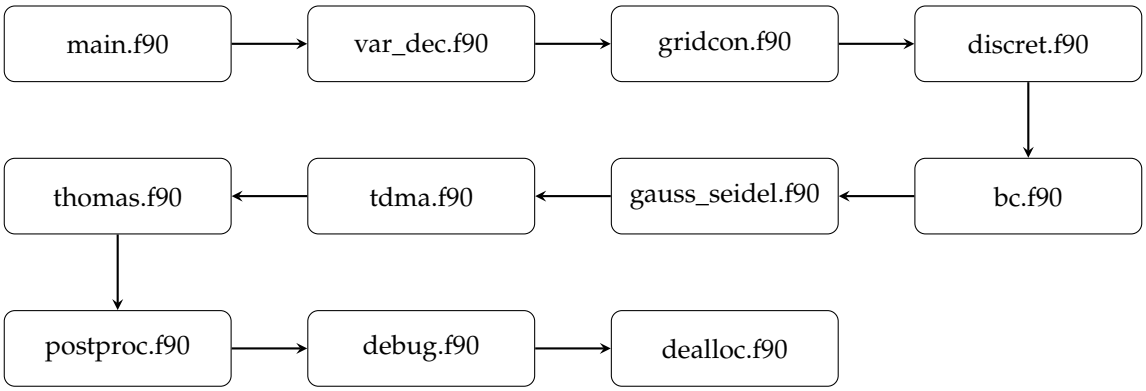


Figure 2: Code Architecture

The architecture of the code developed is as depicted in Fig. 2. The program is initiated by *main.f90*, this in turn calls the *var_dec.f90* module which declares all the variables, arrays and the precision to be used by the code. After this the meshing and initial conditions subroutine, *gridcon.f90* is called which reads the relevant domain and mesh parameters from the input files and allocates and initialises the mesh arrays. Post-meshing the boundary conditions are allocated and initialised by the *bc.f90* subroutine.

Further, the discretisation subroutine *discret.f90* is invoked which calculates the coefficents of the dicretised equation througout the domain based on the hybrid differencing scheme. This is followed then by the solver subroutine, *gauss_seidel.f90* which incorporates a Gauss-Seidel solver and checks for the residual at each iteration against the tolerance. The discretised equation is also solved using the triadiagonal matrix algorithm for which the appropriate scripts have been developed (*tdma.f90* and *thomas.f90*). Once the solution is obtained postprocessing and debugging is carried out by the respective subroutines. Finally, the *dealloc.f90* subroutine deallocates all the allocated arrays.

Few additional utilities have been also developed. A *makefile* is used to make the executable. Custom bash scripts to plot both the residuals and the mesh have been developed using *gnuplot*. Further the temperature and heat flux fields are written to *.tec* files and visualised using ParaView.

3 Computational Domain and Initial Conditions

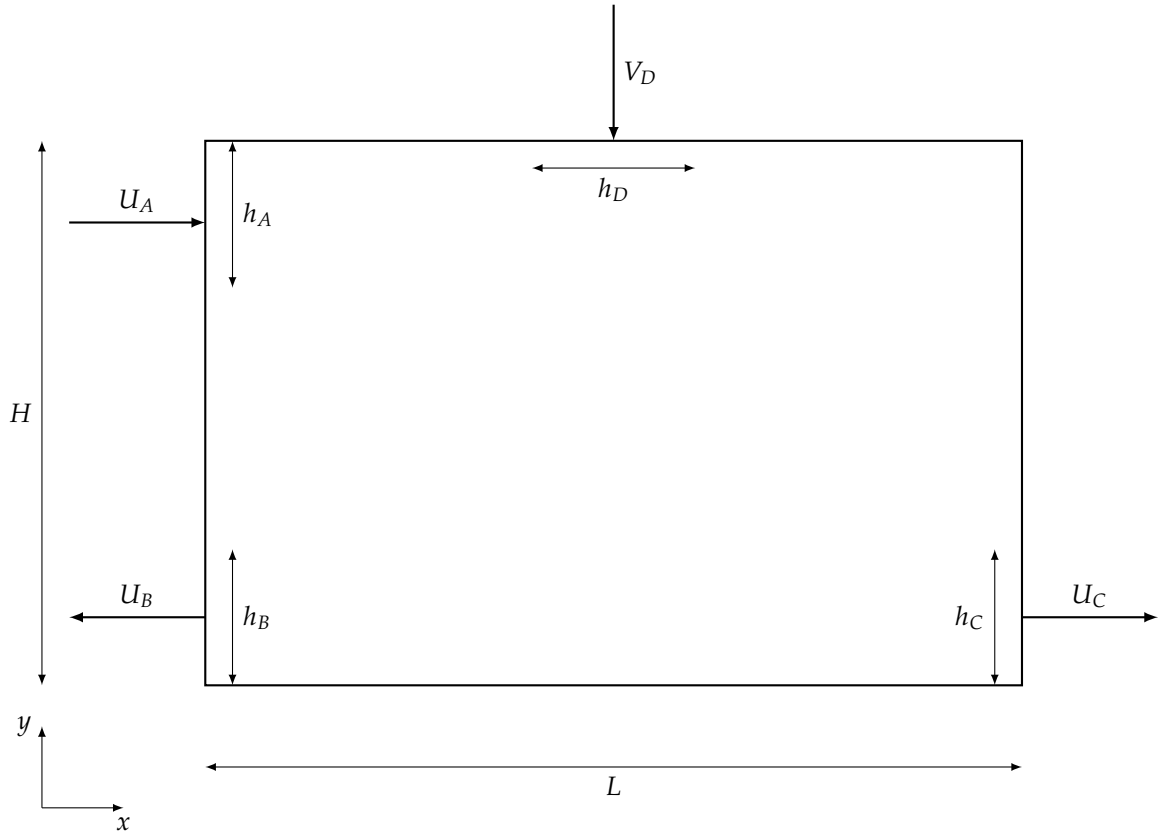


Figure 3: Computational Domain and Boundary Conditions, Physical parameters: $\rho = 1$, $k/c_p = 1/50$, and $h_A/H = h_C/H = 0.068$. Boundary velocities: $U_A = 1$, $U_B = 0$, $U_C = 1$, $V_D = 0$. Temperature boundary conditions: $T_A = 20°C$ and $T = 50°C$ at $x = L$ except over the outlet.

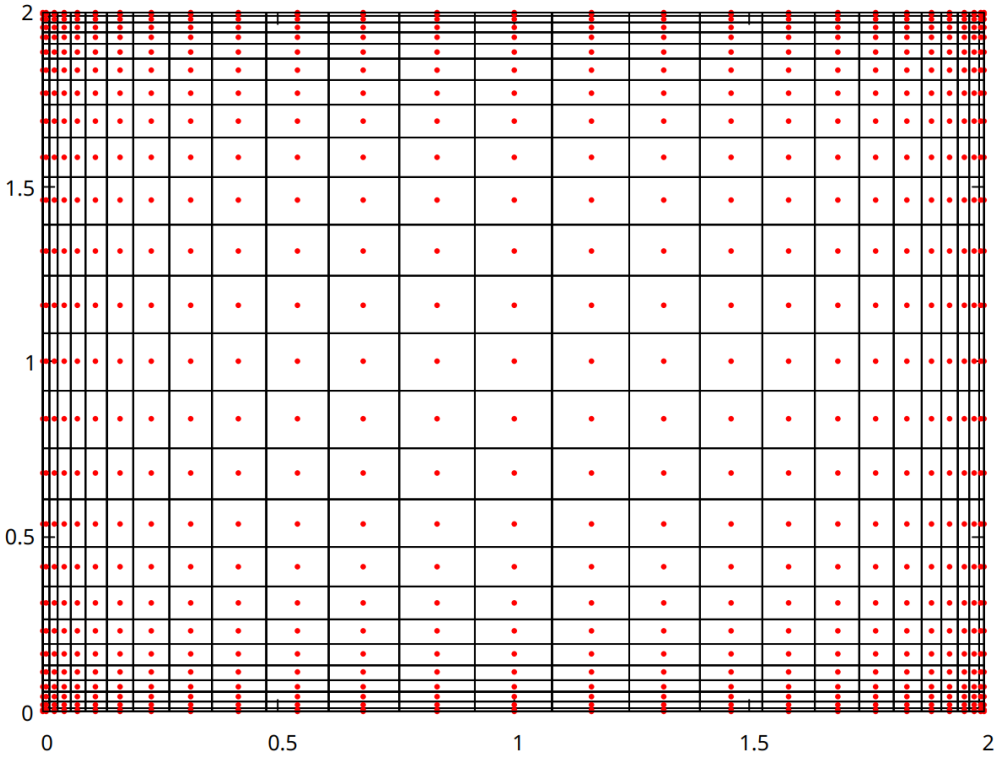


Figure 4: Mesh

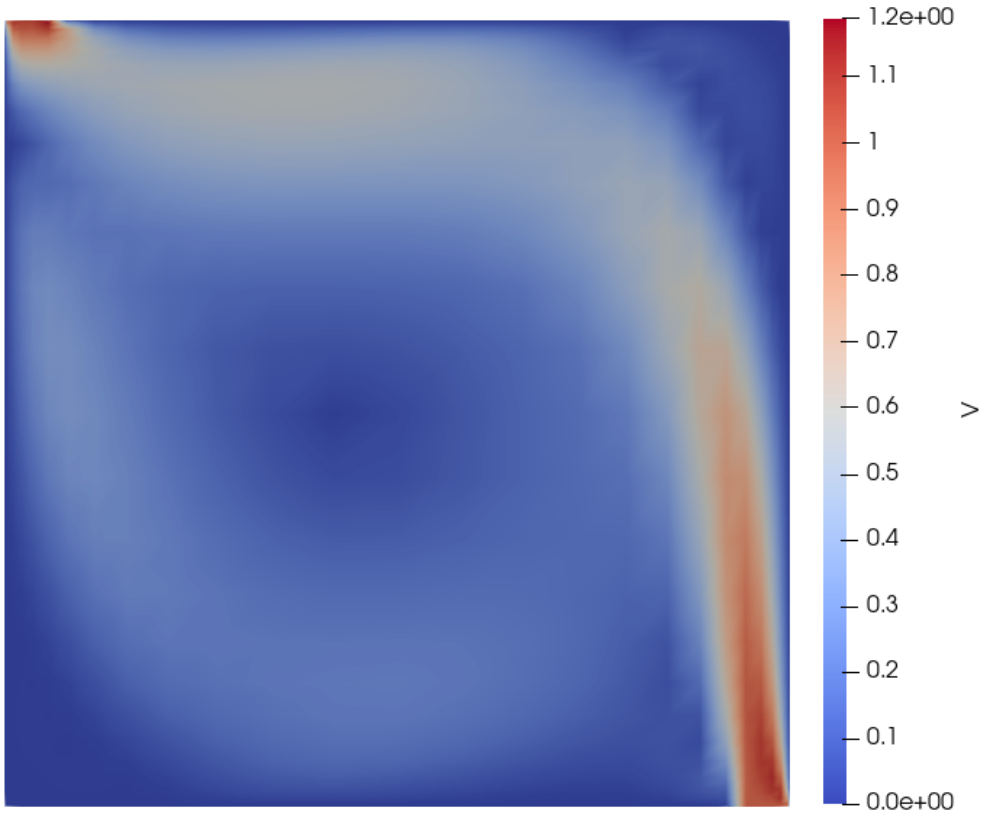


Figure 5: Initial Velocity Conditions - Velocity Magnitude

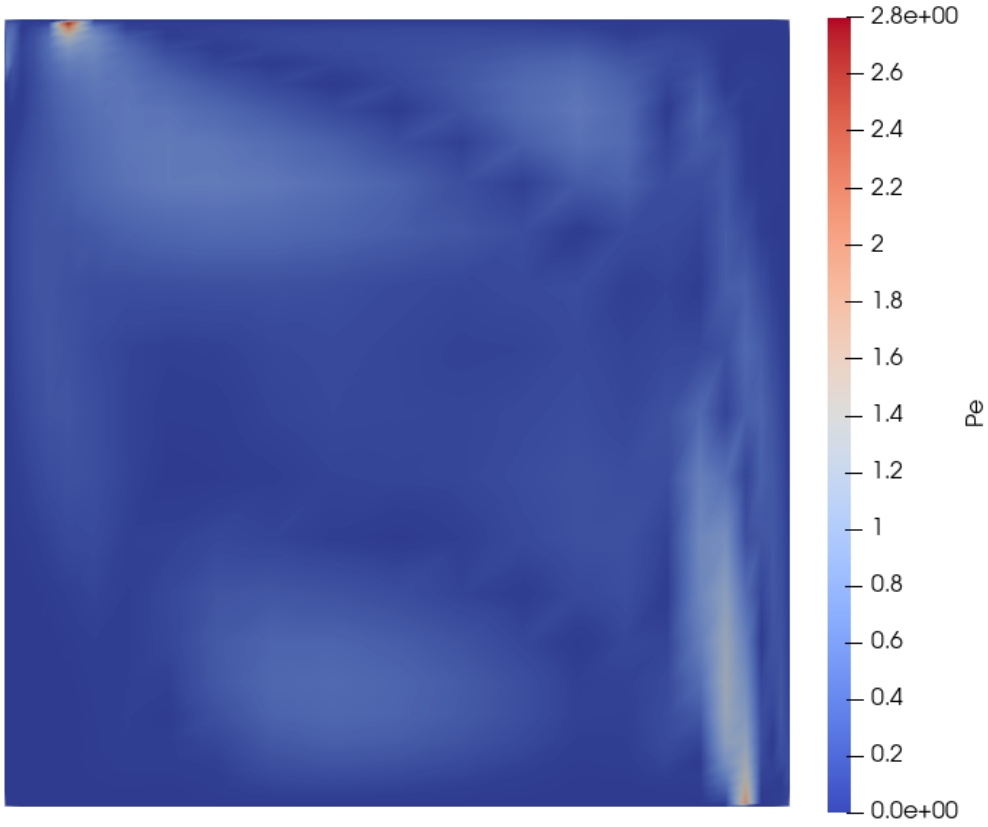


Figure 6: Peclet Number Distribution

4 Results

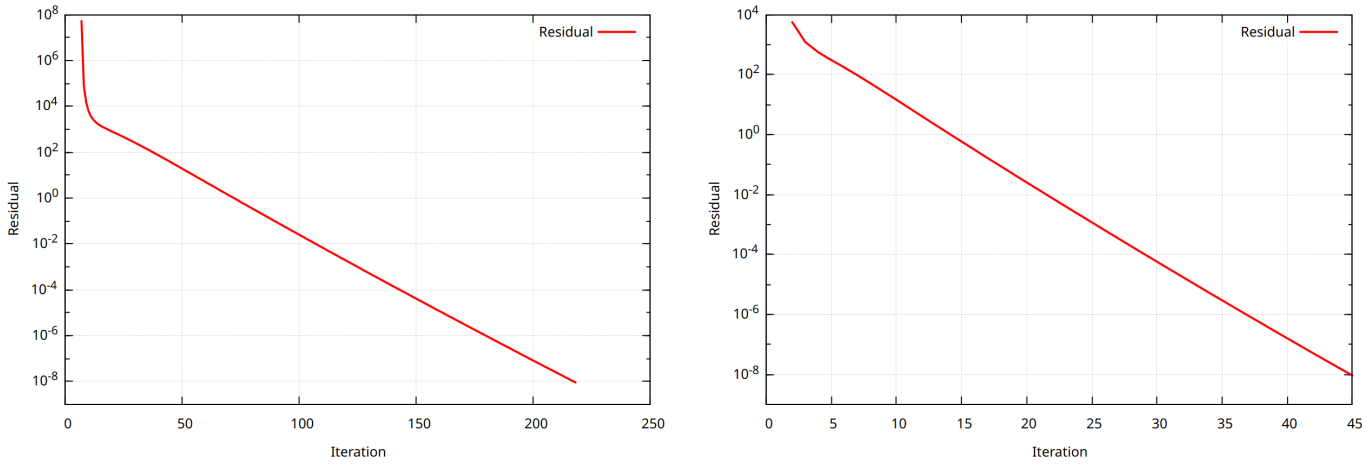
There are some physical inconsistencies in the results obtained. The initial velocity field does not respect global mass conservation and isn't divergence free. This is evident from the output of *debug.f90* as below,

```
mass fluxes (m_w, m_e, m_s, m_n) =  -1.9090950000000000       0.19308529999999999
 ↪  0.0000000000000000        0.0000000000000000
 net mass flux =  -1.7160097000000001

Divergence diagnostics:
  min div =  -119.67788538244828
  max div =   108.69283112341441
```

This net mass accumulation in turn causes heat flux to accumulate which is observed in the solution obtained by both Gauss Seidel and TDMA methods, as outlined below,

```
 max difference GS-TDMA =   2.9891898851996945E-011

 Fw (west)  =   1.6308760612730819
 Fe (east)  =  -6.5215481327922911
 Fs (south) =   0.0000000000000000
 Fn (north) =   0.0000000000000000
 Sum        =  -4.8906720715192087

 Net convective + conductive flux =   -4.8906720715192087       (Gauss Seidel)

 Fw (west)  =   1.6308760612701265
 Fe (east)  =  -6.5215481327903948
 Fs (south) =  -1.0714630787278226E-012
 Fn (north) =   4.9725017660951740E-013
 Sum        =  -4.8906720715208420

 Net convective + conductive flux =   -4.8906720715208420       (TDMA)
```

The solution and its convergence (at $\epsilon = 1e-08$) obtained is depicted below,



(a) (b)

Figure 7: Convergence of *(a)* Gauss Seidel and *(b)* TDMA

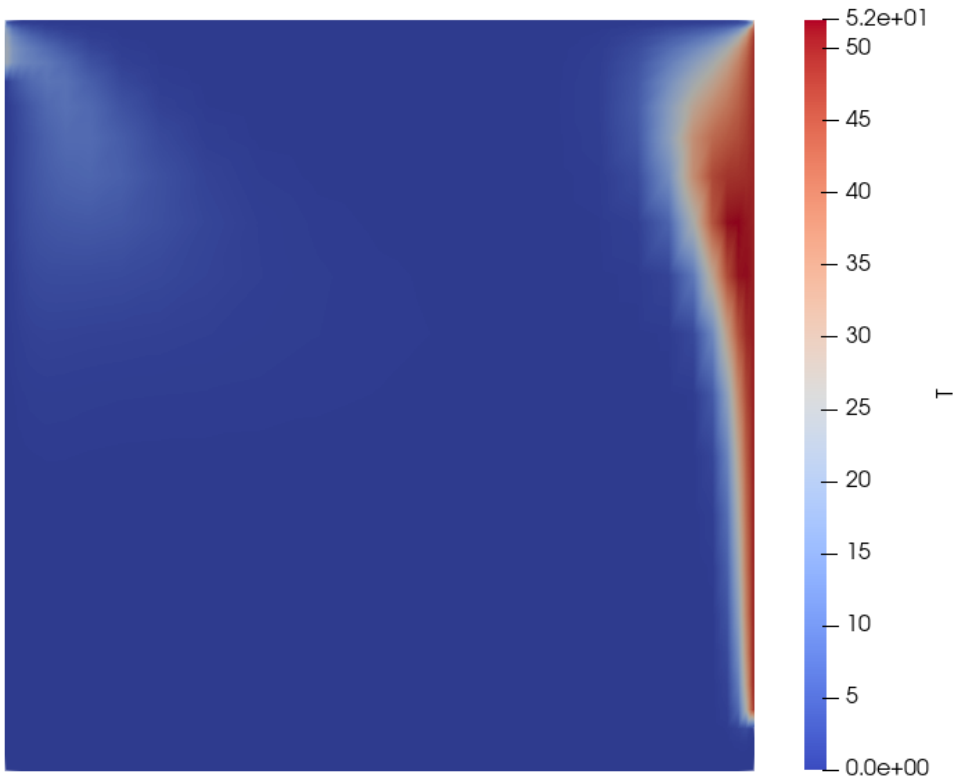


Figure 8: Temperature Distribution

The effect of the convergence criteria of TDMA and taking the Gauss Seidel solution at ($\epsilon = 1e-08$) as a benchmark is as follows,

```
max difference GS-TDMA (eps = 0.01) =   6.6836122435631751E-005
max difference GS-TDMA (eps = 0.0001) =   4.5701957418486927E-007
```

There is no visible visual change in the temperature distribution. When changing the outlet temperature to 20 instead of 50, the convective flux reduces (since Fe reduces do the reduced thermal gradient) and hence the net heat flux reduces,

```
 max difference GS-TDMA =   1.5526729858425514E-011

 Fw (west)  =   1.6309233161464849
 Fe (east)  =  -2.6019733135943635
 Fs (south) =   0.0000000000000000
 Fn (north) =   0.0000000000000000
 Sum        = -0.97104999744787857

 Net convective + conductive flux =  -0.97104999744787857       (Gauss Seidel)

 Fw (west)  =   1.6309233161448276
 Fe (east)  =  -2.6019733135938532
 Fs (south) =  -5.8683753855234988E-013
 Fn (north) =   1.0734978014683566E-013
 Sum        = -0.97104999744950515

 Net convective + conductive flux =  -0.97104999744950515       (TDMA)
```

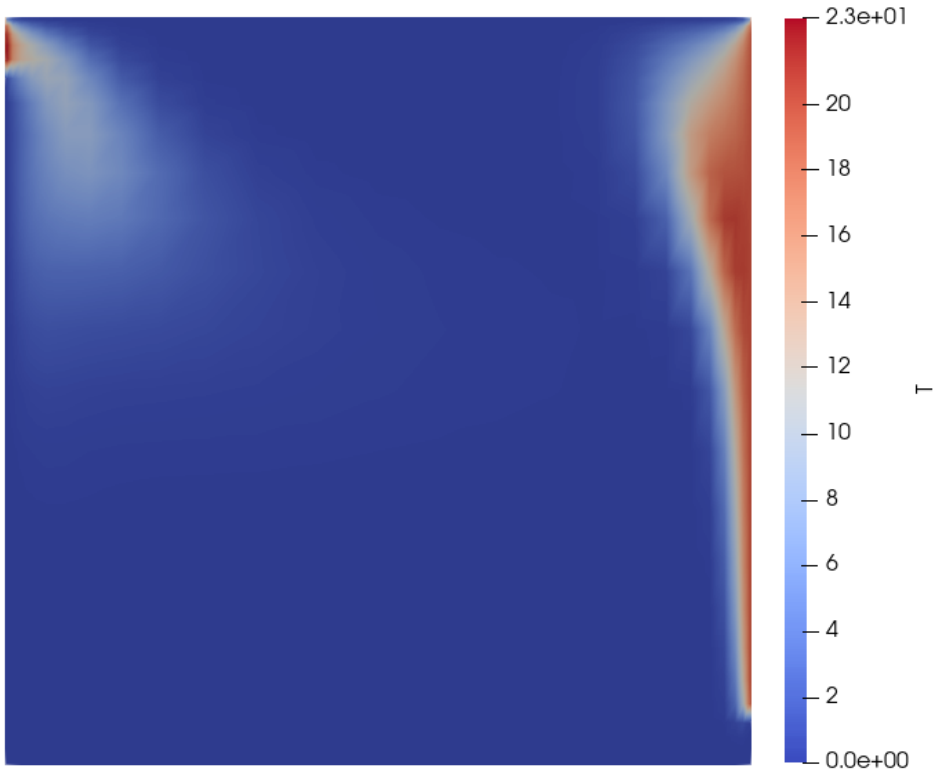


Figure 9: Temperature Distribution (modified BC)

5 References

- H. Versteeg and W. Malalasekera. *An Introduction to Computational Fluid Dynamics - The Finite Volume Method*, Longman Scientific & Technical, Harlow, England, 1st edition, 1995.
- Ferziger, J.H. and Peric, M. (2002) *Computational Methods for Fluid Dynamics*, 3rd Edition, Springer, Berlin.