# On Discretizing the Steady-State 2D Convection-Diffusion Equation Using Upwind and Central Difference Schemes

Jorge Luis Mares Zamora
*The University of Alabama in Huntsville*

**This project report presents the development of a solution to the 2D Steady-State Convection-Diffusion equation using Finite Difference techniques in MATLAB. The equation is solved in a square domain of side length $l = 1$, where the boundary conditions at each of the edges of the domain are given. The diffusion term is discretized using a central difference scheme for the second derivative. The discretization of the convection term is done using three separate techniques: first order UDS, second order UDS, and a central difference scheme; allowing us to do a qualitative comparison on the effects that each of these discretization techniques has on the approximated solution. The developed MATLAB program allows for a change in input values of the flow direction angle, diffusivity, and domain dimensions, $N_x$ and $N_y$.**

## I. Introduction

The steady-state convection-diffusion equation (Equation 1) describes how a given quantity, $\phi$, changes throughout a domain through the effects of convection (being carried by the velocity of a moving fluid), and by the effects of diffusion.

$$u\frac{\partial \phi}{\partial x} + v\frac{\partial \phi}{\partial y} = \alpha\left(\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2}\right) \tag{1}$$

The domain for this problem (Figure 1) is a unit square with a size of 1x1. $\phi$ is fixed at the East and South boundaries. At the South boundary, $\phi = 0$, and at the South boundary, $\phi = 0 \;\forall\; y < 0.25$, and $\phi = 1 \;\forall\; y > 0.25$. The West and North boundaries have a Linear Extrapolation boundary condition for $\phi$, meaning that the governing equation at those boundaries will compute $\phi$ by linearly extrapolating from the two closest interior nodes in the direction normal to the boundary.

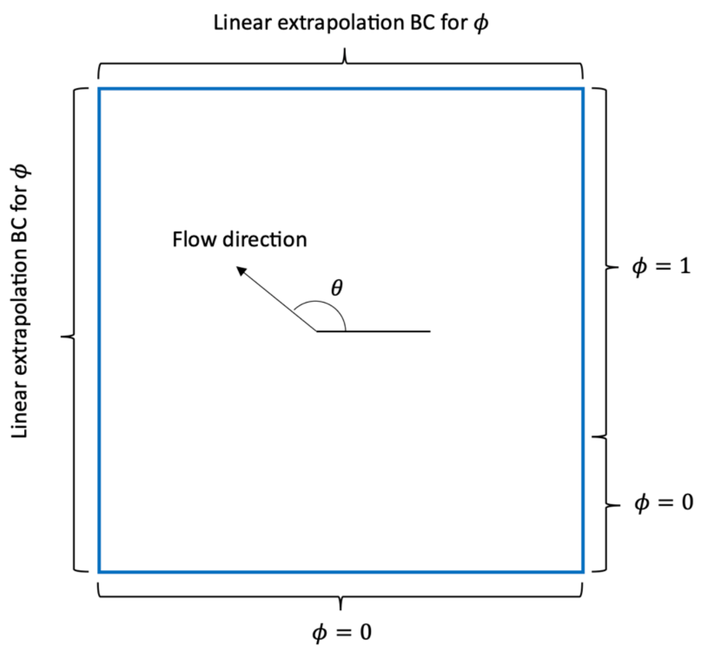
Linear extrapolation BC for $\phi$
Linear extrapolation BC for $\phi$
Flow direction
$\theta$
$\phi = 1$
$\phi = 0$
$\phi = 0$

**Figure 01: Schematic of the domain for the PDE.**

It is important to note that the convective-diffusion equation has no partial derivatives with respect to time, or any other time terms in the equation. This means that the solution is a steady-state one, where time marching will not be required to arrive to the desired solution. In this equation, there are two main "mechanisms" acting on $\phi$: the left hand side of the equation which represents the convection terms; and the right hand side of the equation which represents the diffusion terms. In the left hand side:

$$u = \cos(\theta) \tag{2}$$

$$v = \sin(\theta) \tag{3}$$

Here, the flow direction angle $\theta$ is given as an input parameter, and both $u$ and $v$ are constant throughout the entirety of the domain. To arrive to the solution of this problem, the diffusion term (R.H.S. of the PDE) will be discretized using a central difference scheme. On the other hand, the convective term (L.H.S. of the PDE) will be discretized using three separate techniques: 1st order upwind scheme (UD1), 2nd order upwind scheme (UD2), and a central difference scheme (CDS). The developed MATLAB program will then take the following input parameters: $\theta$, $\alpha$, $N_x$, $N_y$, and the desired scheme to be followed for the discretization of the convection terms.

## II. Methodology

The diffusion term of the P.D.E. is to be discretized using a central difference scheme for all the three separate discretization cases to be followed in this report. Then, the first step to arriving to the solution is to find this discretized form of the right hand side of the equation that the rest of the schemes will also use. The general formula for the central difference discretization (Equation 4), can then be applied to all the terms in the diffusion term. This discretization yields the following expression:

$$\left(\frac{\partial^2 \phi}{\partial x^2}\right)_i = \frac{\phi_{i+1} + \phi_{i-1} - 2\phi_i}{(\Delta x)^2} \tag{4}$$

$$u\frac{\partial \phi}{\partial x} + v\frac{\partial \phi}{\partial y} = \alpha\left(\frac{\phi_{i+1,j} + \phi_{i-1,j} - 2\phi_{i,j}}{(\Delta x)^2} + \frac{\phi_{i,j+1} + \phi_{i,j-1} - 2\phi_{i,j}}{(\Delta y)^2}\right) \tag{5}$$

It is important to note that the expression for the diffusion term in Equation 5 cannot be simplified any further as it is currently stated, since any further simplifications would require the assumption that $\Delta x = \Delta y$. This assumption is a valid one as long as $N_x = N_y$; however, since it is a requirement that our program runs for the case when $N_x \neq N_y$. The discretization of the boundary nodes at the West and South boundaries, the governing equation is just that of Equations 6 and 7.

$$\phi_{i,j} = 0 \ \forall \ South\ nodes \tag{6}$$

$$\phi_{i,j} = \begin{cases} 0 \ \forall \ West\ nodes\ where\ y < 0.25 \\ 1 \ \forall \ West\ nodes\ where\ y \geq 0.25 \end{cases} \tag{7}$$

For the East and North boundaries, the governing equation is obtained by linearly extrapolating from the two nearest interior node values in the direction normal to the boundary. This extrapolation is obtained by the following derivation, and the final boundary expression is presented in Equation 8.

$$\phi_i = \phi_{i+1} + \frac{-\Delta x}{\Delta x}(\phi_{i+2} - \phi_{i+1})$$

$$\phi_i = 2\phi_{i+1} - \phi_{i-2} \tag{8}$$

This equation can be used in both of the boundaries, changing the indexes for each of the interior nodes in the respective directions.

## A. First Order Upwind Scheme

The first approach taken to arrive to the discretization of the convective term is to follow a 1st order upwind scheme (UD1). The UD1 discretization says the following:

$$UD1 = \begin{cases} BDS\ when\ u > 0 \\ FDS\ when\ u < 0 \end{cases} \tag{9}$$

Where:

$$BDS\colon \left(\frac{\partial \phi}{\partial x}\right)_i = \frac{\phi_i - \phi_{i-1}}{\Delta x} \tag{10}$$

$$FDS\colon \left(\frac{\partial \phi}{\partial x}\right)_i = \frac{\phi_{i+1} - \phi_i}{\Delta x} \tag{11}$$

The problem statement made the following restriction on the value of $\theta_p$, where this new variable is the allowed theta in the problem): $\theta_p \in \{90° < \theta < 180°\}$. Then, since $\cos(\theta) < 0\ \forall\ \theta_p$, and $\sin(\theta) > 0\ \forall\ \theta_p$, it follows that we will always need to use FDS in the x-direction and BDS in the y-direction to successfully implement UD1 in our solver. Then the discretization of the P.D.E. using UD1 on the convective terms and CDS on the diffusion term will look as follows.

$$u\left(\frac{\phi_{i+1,j} - \phi_{i,j}}{\Delta x}\right) + v\left(\frac{\phi_{i,j} - \phi_{i,j-1}}{\Delta y}\right) = \alpha\left(\frac{\phi_{i+1,j} + \phi_{i-1,j} - 2\phi_{i,j}}{(\Delta x)^2} + \frac{\phi_{i,j+1} + \phi_{i,j-1} - 2\phi_{i,j}}{(\Delta y)^2}\right) \tag{12}$$

Since the solution to the P.D.E. is steady-state, and thus does not require marching in time, solving this problem just requires solving a system of linear equations. Each one of these equations is the governing equation for the interior nodes (Equation 12), as well as each one of the respective boundary equations at the boundary nodes. The system can be solved using either a direct method or an iterative one, and due to computational efficiency and ease of setup, I chose to use an iterative technique – Gauss Seidel. Setting up Gauss Seidel for this problem involves solving for $\phi_{i,j}$ in Equation 12, resulting in Equation 13:

$$\phi_{i,j} = \frac{-\left(\frac{u}{\Delta x} - \frac{\alpha}{(\Delta x)^2}\right)\phi_{i+1,j} - \left(-\frac{v}{\Delta y} - \frac{\alpha}{(\Delta y)^2}\right)\phi_{i,j-1} + \left(\frac{\alpha}{(\Delta x)^2}\right)\phi_{i-1,j} + \left(\frac{\alpha}{(\Delta y)^2}\right)\phi_{i,j+1}}{\left(\frac{v}{\Delta y} - \frac{u}{\Delta x} + \frac{2\alpha}{(\Delta x)^2} + \frac{2\alpha}{(\Delta y)^2}\right)} \tag{13}$$

Then iteratively solving for each of these equations at all of the interior nodes. The values are stored in an $N_x \times N_y$ matrix, and two for loops are used to update at each of the nodes, each iteration using the newest updated value for each of the nodes. After each of the interior nodes is updated, we use the Equations 6, 7 and 8, to update the values at the boundaries. The Gauss Seidel steps was set to stop when the magnitude approximate error, $\mathcal{E}$, was less than a tolerance of $10^{-10}$.

$$\mathcal{E} = \max\big(abs(\phi_{new} - \phi_{old})\big) \tag{14}$$

## B. Second Order Upwind Scheme

The second scheme used to discretize the convective terms is a second order upwind scheme, UD2. Following the same philosophy as that in part A, since $u < 0$ and $v > 0$, the following discretization formulas were used.

$$\left(\frac{\partial \phi}{\partial y}\right)_{i,j} = \frac{\phi_{i,j-2} - 4\phi_{i,j-1} + 3\phi_{i,j}}{2\Delta y} \tag{15}$$

$$\left(\frac{\partial \phi}{\partial x}\right)_{i,j} = -\frac{3\phi_{i,j} - 4\phi_{i+1,j} + \phi_{i+2,j}}{2\Delta x} \tag{16}$$

Then, the final discretization for all the interior points is presented in Equation 17.

$$\phi_{i,j} = \frac{\left(\frac{2u}{\Delta x} - \frac{\alpha}{(\Delta x)^2}\right)\phi_{i+1,j} + \left(-\frac{u}{2\Delta x}\right)\phi_{i+2,j} + \left(\frac{v}{2\Delta y}\right)\phi_{i,j-2} + \left(-\frac{2v}{\Delta y} - \frac{\alpha}{(\Delta y)^2}\right)\phi_{i,j-1} + \left(-\frac{\alpha}{(\Delta x)^2}\right)\phi_{i-1,j} + \left(-\frac{\alpha}{(\Delta y)^2}\right)\phi_{i,j+1}}{\left(\frac{3u}{2\Delta x} - \frac{3v}{2\Delta y} - \frac{2\alpha}{(\Delta x)^2} - \frac{2\alpha}{(\Delta y)^2}\right)} \tag{17}$$

This discretization method was also solved using the Gauss Seidel iterative technique presented above in Part A of this section, the only difference being the update equation which was used at each of the interior points in the domain. The interior points right next to the South and East boundaries, where there is no place to index two neighboring nodes ahead, used a UD1 discretization instead. For more details on the conditional logic used to handle these cases, please refer to the attached code.

## C. Central Difference Scheme

The third scheme used to discretize the convective terms was CDS. The general formula for CDS is presented below in Equation 18, and the discretized equation using CDS is presented in Equation 19.

$$\left(\frac{\partial \phi}{\partial x}\right)_{i} = \frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x} \tag{18}$$

$$\phi_{i,j} = \frac{\left(\frac{u}{2\Delta x} - \frac{\alpha}{(\Delta x)^2}\right)\phi_{i+1,j} + \left(-\frac{u}{2\Delta x} - \frac{\alpha}{(\Delta x)^2}\right)\phi_{i-1,j} + \left(\frac{v}{2\Delta y} - \frac{\alpha}{(\Delta y)^2}\right)\phi_{i,j+1} + \left(\frac{-v}{2\Delta y} - \frac{\alpha}{(\Delta y)^2}\right)\phi_{i,j-1}}{-\left(\frac{2\alpha}{(\Delta x)^2} + \frac{2\alpha}{(\Delta y)^2}\right)} \tag{19}$$

This scheme was initially attempted to be solved using Gauss Seidel, however, due to the unstable nature of CDS for the convective terms (as well as the A matrix of the system of equations not being diagonally dominant), the solution did not converge. Then, in order to get a solution, a direct method had to be attempted. The resulting system is of size $N_x \times N_y$. This will thus also be the size of the coefficient matrix A. Since all of the values of $\phi$ have to be stored in a column matrix, the matrix to column matrix conversion followed the index convention presented in Table 3.2 of the Computational Methods for Fluid Dynamics Textbook, where the index of each of the points in the column matrix is defined as follows:

$$l = (m-1)N_x + j \tag{20}$$

Then, a column matrix with all the initial values of $\phi$ was created, and the column coefficients were calculated. The coefficient matrix, $A$, is a sparse matrix, which according to the textbook is better off stored as just the diagonals rather than a full matrix. Given MATLAB's ability to work with matrices though, I decided to store my coefficient matrix as a sparse matrix, which managed all the storage issues whilst providing the flexibility to use the same workflow as with normal matrices (i.e. solving matrix equations and indexing using regular notation). With the equation coefficients matrix created (using the MATLAB function *spdiag*), the coefficients of the boundary conditions equations needed to be modified for the rows of the matrix that corresponded to the nodes in the boundary of our $T$ column matrix. The index of the nodes in the boundary change depending on the size of our square domain, so they were calculated using the following logic:

$$p = 1,2,\ldots,N_y \;\; where \;\; p \in \{West\ Boundary\} \tag{21}$$

$$p = N_y^2 - Ny, N_y^2 - N_y + 1, \ldots, Ny^2 \; where \; p \in \{East\ Boundary\} \tag{22}$$

$$p = N_x * n + 1 \; \forall \; n \in (1,2,\ldots,N_x) \; where \; p \in \{South\ Boundary\} \tag{23}$$

$$p = N_x * n \; \forall \; n \in \{1,2,\ldots,N_x) \; where \; p \in \{North\ Boundary\} \tag{24}$$

Then, the system was solved without running into convergence issues related to the iterative Gauss Seidel technique.

## III. Results

The developed program was run to solve the problems using all three of the discretization approaches presented in the methodology section of this report. The input parameters used to obtain the results in this report are the following:

$$\theta = 150^\circ$$

$$\alpha = 10^{-6}$$

$$N_x = N_y = 11, 21, and\ 41$$

The results for the coarse grid ($N_x = N_y = 11$), medium grid ($N_x = N_y = 21$), and fine grid ($N_x = N_y = 41$), are presented in Figure 02, Figure 03, Figure 04, respectively.

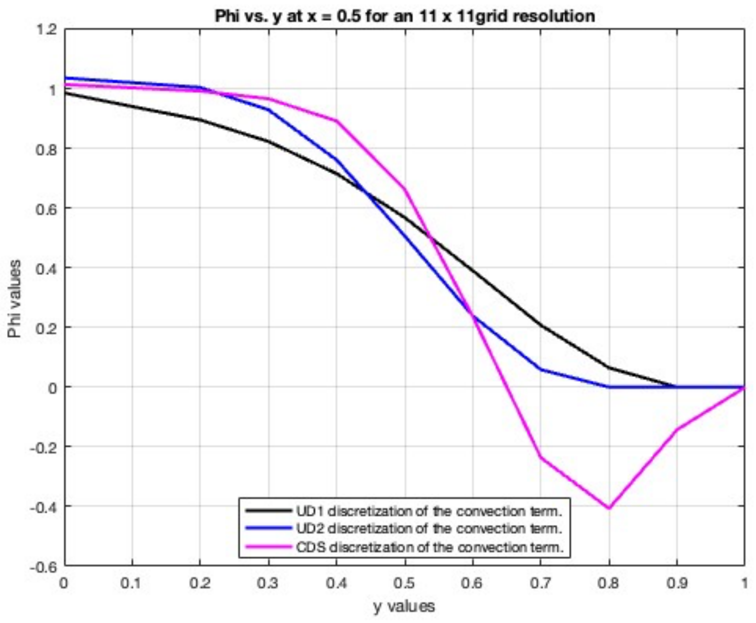

**Figure 02: Results for the 11x11 grid resolution.**

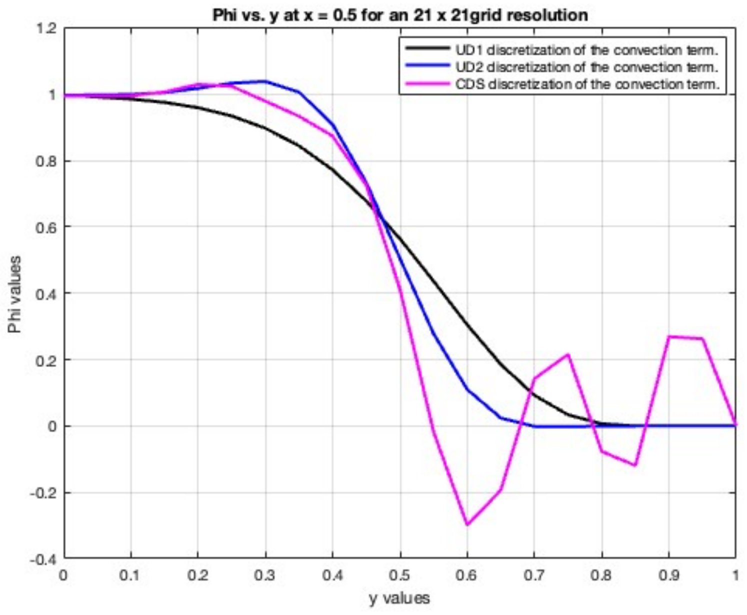

**Figure 03: Results for the 21x21 grid resolution.**

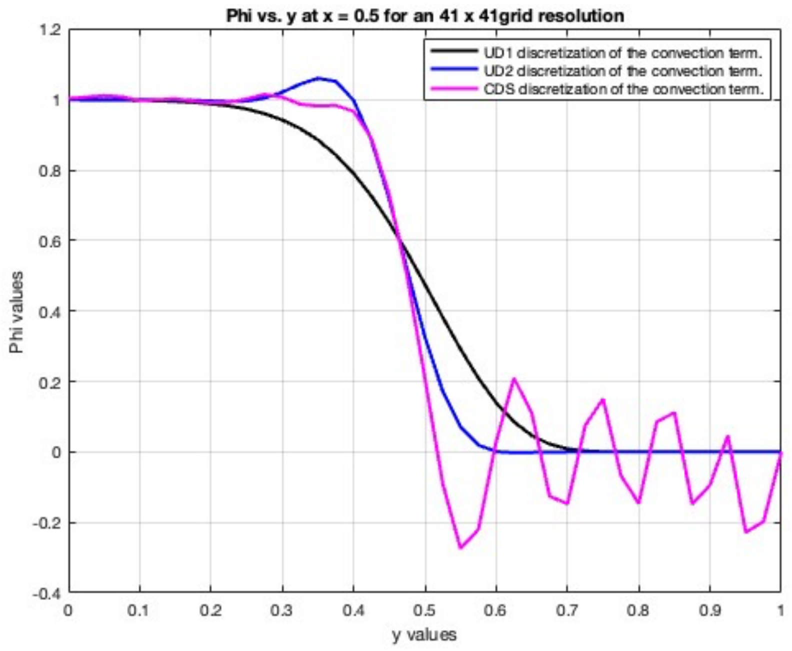

**Figure 04: Results for the 41x41 grid resolution.**

Looking at these results, it is possible to make a few observations about each of these discretization schemes. The most relevant feature is that CDS is completely unstable in its approximations and shows an oscillatory behavior. The oscillations are expected and show how CDS is not great at discretizing the convective terms of the PDE. These oscillations are also presented and discussed in Figure 3.8 of the textbook and agree with the discussion that we had in class when reviewing this material. There are also some key differences between the results produced by UD1 and UD2, mainly that, UD1 shows less sharp of a gradient than UD2 does. This also is what is expected according to the class discussion as well as the textbook, as the UD1 scheme trades off stability for added numerical diffusivity.

## IV. Conclusions

The 2D steady-state convection-diffusion equation was successfully solved in a unit square domain during this project. The presented results highlight the differences between using upwind schemes and central differencing when discretizing convection terms. Although in theory, central differencing should be more accurate, it is possible to see that there is a trade off to be made regarding its numerical stability, and how depending on the problem it may not converge nicely to an accurate approximation of the solution. On the other hand, it is possible to see how the first order upwind scheme is numerically stable but has a very prevalent added numerical diffusion to the approximation of the solution, which is especially prevalent at lower grid resolutions. UD2 posed a balance between these two schemes, providing a higher order approximation whilst also providing numerical stability to the solution of the problem.

## References

[1] Ferziger. J. H., Peric M., Street R. L., *Computational Methods for Fluid Dynamics,* 4th Ed., Springer, Chap. 3.

[2] Shotorban B, Lecture Notes MAE 623, Fall 2025