# Progress Report

Syed Ahmad Raza

2018.05.09

# 1 2D Finite Volume solver Navier-Stokes using two layers of ghost cells

## 1.1 Ghost cells

Two layers of ghost cells were implemented in C++ using Finite Volume Method to solve the Navier-Stokes equations, as shown in the following figure for a reduced grid size.

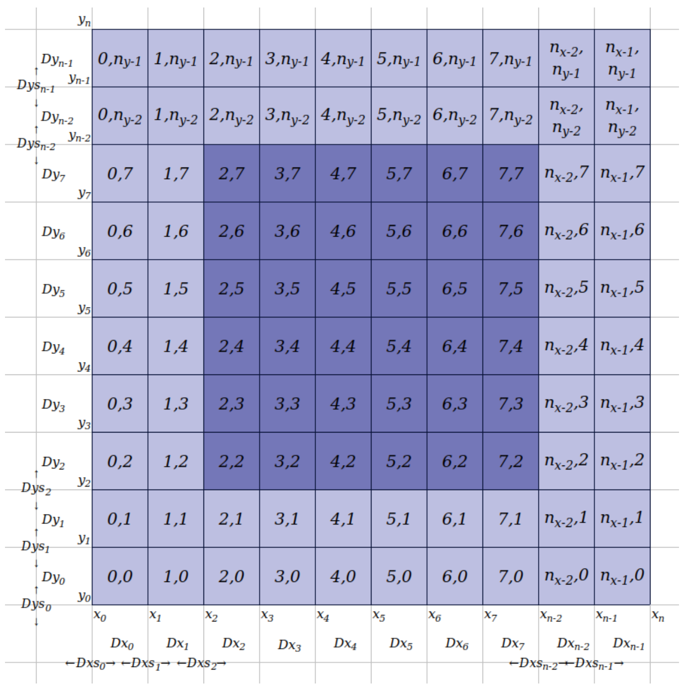

Figure 1: Figure representing the utilization of two ghost cells around the grid, for grid of reduced size

## 1.2 Debugging

Earlier code had a major bug, which in that it implemented the far-side velocity boundary conditions incorrectly. The corrected velocity boundary conditions in the x and y directions for the far side are given below.

$u$-velocity for the east boundary:

$$u_{n_x-3,j} = 0 \qquad u_{n_x-2,j} = u_{n_x-3,j} \tag{1}$$

$v$-velocity for the north boundary:

$$v_{i,n_y-3} = 0 \qquad v_{i,n_y-2} = v_{i,n_y-3} \tag{2}$$

This can be understood with the help of the following figure.

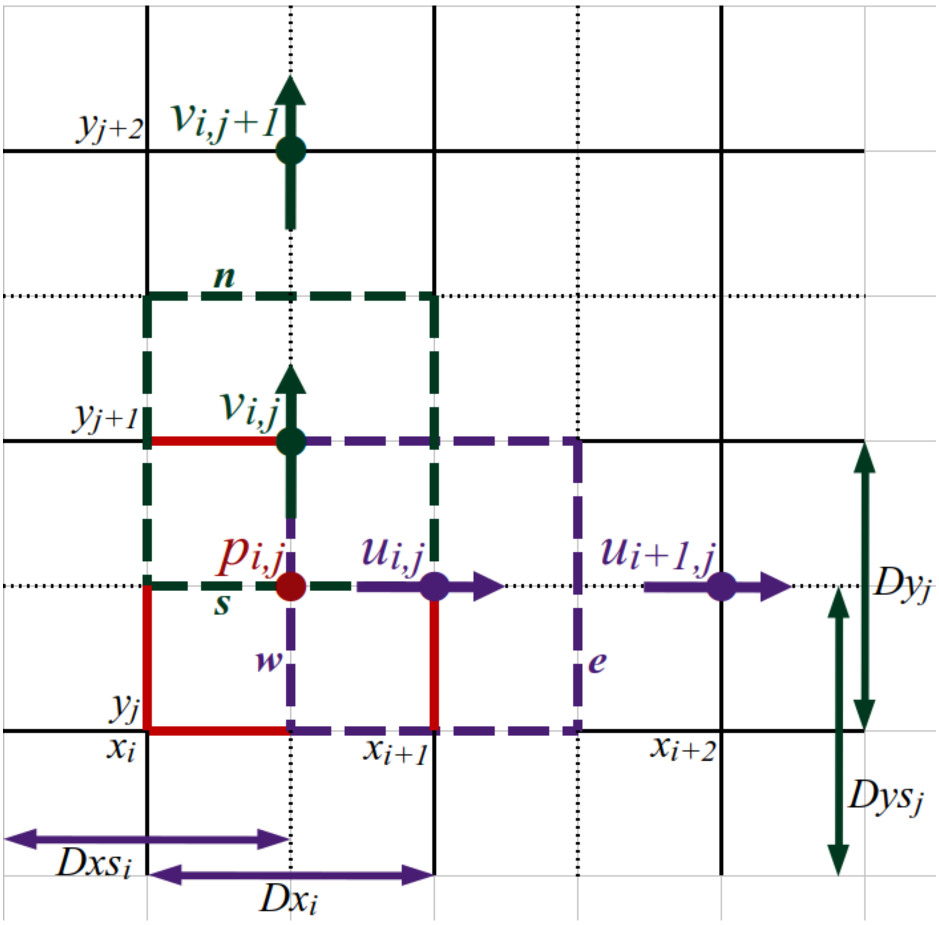


Figure 2: Representation of the utilization of two ghost cells around the grid, for grid of reduced size

## 1.3 Results

The code was solved for the case of a lid-driven cavity flow in a two-dimensional square box. The results are shown in the following figures.

Figure 3: Comparing the solution provided by Ghia et. al versus the numerical solution for $u$-velocity values for all $y$ at the center of $x$ -axis, for a 121×121 grid.

Figure 4: Comparing the solution provided by Ghia et. al versus the numerical solution for $v$-velocity values for all $x$ at the center of $x$ -axis, for a 121×121 grid.

## 1.4 Ongoing tasks

1. Calculating the virtual force for a cylinder inside a lid-driven cavity
2. Employ parallel computing by using ultraMPP C++ library to execute the code on parallel cores
3. Solve cavity-driven flow for 3D