2 D Poiseuille Flow

```
Clear[vx, vy, Fx, Fy, p, Nx, Ny, U, ix, iy, t, dt, T, s, A];
ρ = 1.2041; (*density of the fluid (air at 293.15K in this case)*)
L = 10; (*Physical length of the grid*)
a = 1; (*Physical height of the grid*)
Nx = 40; (*Number of grid points over the length of the grid*)
Ny = 20; (*Number of grid points over the height of the grid*)
U = 1; (*Entry velocity of the fluid*)
maxtime = 400; (*The number of time-iterations*)
Rey = 100; (*Reynolds number for the fluid*)
nu = (U * a)/Rey; (*The dynamics viscosity of the fluid*)
dt = 0.02; (*The physical time between time stamps*)
Δx = L/Nx; (*The physical distance between grid points in the length direction*)
Δy = a/Ny; (*The physical distance between grid points in the height direction*)
(*The relevant series of matrices are created*)
vx = Table[Table[0, {Ny + 2}, {Nx + 1}], {maxtime}];
vy = Table[Table[0, {Ny + 1}, {Nx + 2}], {maxtime}];
Fx = Table[Table[0, {Ny + 2}, {Nx + 1}], {maxtime}];
Fy = Table[Table[0, {Ny + 1}, {Nx + 2}], {maxtime}];
p = Table[Table[0, {Ny + 2}, {Nx + 2}], {maxtime}];
s = Table[Table[0, {Ny + 2}, {Nx + 2}], {maxtime}];
A = Table[Table[0, {Ny + 2}, {Nx + 2}], {maxtime}];

(*The velocity in the length direction is initialized*)
For[i = Nx + 1, i ≥ 1, i--,
  For[j = Ny + 1, j > 1, j--,
    vx[[1, j, i]] = U;
   ];
 ];

(*The A-matrix used in the pressure-subroutine with incorporated BC's (A.p=s).*)
Clear[n, A];
A = Table[0, {(Ny) * (Nx)}, {(Ny) * (Nx)}];
n = 0;
For[i = 1, i ≤ (Ny) * (Nx), i++,
  n++;
  If[n == Nx + 1, n = 1, 0];
  For[j = 1, j ≤ (Ny) * (Nx), j++,
   A[[i, j]] =
    (-2 (1/(Δy) ^2 + 1/(Δx) ^2) + If[1 ≤ i ≤ Nx ∨ Ny * Nx - Nx + 1 ≤ i ≤ Ny * Nx, 1/(Δy) ^2, 0] +
        If[n == 1 ∨ n == Nx, 1/(Δx) ^2, 0]) * KroneckerDelta[i, j] + 1/(Δx) ^2 *
      (If[n ≠ Nx, KroneckerDelta[i, j - 1], 0] + If[n ≠ 1, KroneckerDelta[i, j + 1], 0]) +
     1/(Δy) ^2 * (KroneckerDelta[i, j - (Nx)] + KroneckerDelta[i, j + Nx])
```

```
   ];
  ];
 (*The correct solution is only obtained
  when the rank of A is equal to the number of rows*)
 Clear[n];
 n = 0;
 For[i = 1, i ≤ (Ny) * (Nx), i++,
   n++;
   If[Dimensions[A][[1]] ≠ MatrixRank[A],
    A = A[[1 ;; (Ny) * (Nx) - n, 1 ;; (Ny) * (Nx) - n]];
    , Indeterminate];
  ];

 (*Begin the time loop untill maxtime-1*)
 For[t = 1, t < maxtime, t++,
   (*calculate the next F_x-vlaue for internal points*)
   For[iy = 2, iy ≤ Ny + 1, iy++,
    (*iy runs over the vertical physical space + dummy boundary*)
    For[ix = 2, ix ≤ Nx, ix++, (*ix runs over the horizontal physical space+
       dummy boundary*)
```

$$Fx[[t, iy, ix]] = \rho * \left(-vx[[t, iy, ix]] * \frac{vx[[t, iy, ix+1]] - vx[[t, iy, ix-1]]}{2 * \Delta x} - \frac{1}{4}\left(vy[[t, iy, ix]] + vy[[t, iy, ix+1]] + vy[[t, iy-1, ix]] + vy[[t, iy-1, ix+1]]\right) * \frac{vx[[t, iy+1, ix]] - vx[[t, iy-1, ix]]}{2 * \Delta y}\right) + nu * \left(\frac{1}{(\Delta x)^\wedge 2}\left(vx[[t, iy, ix+1]] + vx[[t, iy, ix-1]] - 2\, vx[[t, iy, ix]]\right) + \frac{1}{(\Delta y)^\wedge 2}\left(vx[[t, iy+1, ix]] + vx[[t, iy-1, ix]] - 2\, vx[[t, iy, ix]]\right)\right);$$

```
     ];
   ];
   (*calculate the next F_y-vlaue for internal points*)
   For[iy = 2, iy ≤ Ny, iy++,
    For[ix = 2, ix ≤ Nx + 1, ix++,
```

$$Fy[[t, iy, ix]] = \rho * \left(-vy[[t, iy, ix]] * \frac{vy[[t, iy+1, ix]] - vy[[t, iy-1, ix]]}{2 * \Delta y} - \frac{1}{4}\left(vx[[t, iy, ix]] + vx[[t, iy, ix-1]] + vx[[t, iy+1, ix]] + vx[[t, iy+1, ix-1]]\right) * \frac{vy[[t, iy, ix+1]] - vy[[t, iy, ix-1]]}{2 * \Delta x}\right) + nu * \left(\frac{1}{(\Delta y)^\wedge 2}\left(vy[[t, iy+1, ix]] + vy[[t, iy-1, ix]] - 2\, vy[[t, iy, ix]]\right) + \frac{1}{(\Delta x)^\wedge 2}\left(vy[[t, iy, ix+1]] + vy[[t, iy, ix-1]] - 2\, vy[[t, iy, ix]]\right)\right);$$

```
     ];
   ];
   (*calculate the next source-vlaue for internal points*)
   For[ix = 2, ix ≤ Nx + 1, ix++,
    For[iy = 2, iy ≤ Ny + 1, iy++,
      s[[t, iy, ix]] =
```

```
        (Fx[[t, iy, ix]] - Fx[[t, iy, ix - 1]])/Δx + (Fy[[t, iy, ix]] - Fy[[t, iy - 1, ix]])/Δy + ρ/dt *
         ((vx[[t, iy, ix]] - vx[[t, iy, ix - 1]])/Δx + (vy[[t, iy, ix]] - vy[[t, iy - 1, ix]])/Δy);
   ];
];
(*Start the subroutine that determine the pressure corresponding to the source*)
(*First, arrange the s-matrix indices in a vector, S*)
Clear[S, n];
S = Table[0, {Nx * Ny}];
n = 0;
For[i = 2, i ≤ Ny + 1, i++,
 For[j = 2, j ≤ Nx + 1, j++,
   n++;
   S[[n]] = s[[t, i, j]];
  ];
];
(*Next, remove entries from the end of S corresponding
  to the rows removed in A in order to obtain rank(A)=rows(A)*)
Clear[temp];
S = S[[1 ;; Dimensions[A][[1]]]];
(*Solve A.x=S. Inverting such large matrices is computationally too heavy,
so approximate methods must be applied. This can be done via SOR,
but the inbuilt function is much, much faster.*)
temp = LinearSolve[A, S];
(*"temp" is a vector corresponding to S. This is rearranged
 into a matrix with pressure measurements, corresponding to s*)
n = 0;
For[i = 1, i ≤ Ny, i++,
 For[j = 1, j ≤ Nx, j++,
   n++;
   If[n ≤ Dimensions[temp][[1]] ⋀ temp[[j + (i - 1) * Nx]] > 10^-5,
    p[[t, i + 1, j + 1]] = temp[[j + (i - 1) * Nx]];
    , Indeterminate];
  ];
];
(*The boundary terms are set equal to the outer interla points. Without
  incorporating the BC's in the A-matrix this would eb wrong*)
For[ix = 2, ix ≤ Nx + 1, ix++,
 p[[t, 1, ix]] = p[[t, 2, ix]];
 p[[t, Ny + 2, ix]] = p[[t, Ny + 1, ix]];
];
For[iy = 2, iy ≤ Ny + 1, iy++,
 p[[t, iy, 1]] = p[[t, iy, 2]];
 p[[t, iy, Nx + 2]] = p[[t, iy, Nx + 1]];
];
(*calculate the next v_x-vlaue for internal points*)
For[ix = 1, ix ≤ Nx + 2, ix++,
 For[iy = 1, iy ≤ Ny + 2, iy++,
   If[2 ≤ iy ≤ Ny + 1,
    If[2 ≤ ix ≤ Nx,
      vx[[t + 1, iy, ix]] =
```

```
            vx[[t, iy, ix]] + dt/ρ * (Fx[[t, iy, ix]] - (p[[t, iy, ix + 1]] - p[[t, iy, ix]])/Δx);
         , Indeterminate];
      , Indeterminate];
     (*calculate the next v_y-vlaue for internal points*)
     If[2 ≤ iy ≤ Ny,
      If[2 ≤ ix ≤ Nx + 1,
         vy[[t + 1, iy, ix]] =
            vy[[t, iy, ix]] + dt/ρ * (Fy[[t, iy, ix]] - (p[[t, iy + 1, ix]] - p[[t, iy, ix]])/Δy);
         , Indeterminate];
      , Indeterminate];
     (*Applty BC's for V_x and v_y*)
     If[2 ≤ iy ≤ Ny + 1,
      vx[[t + 1, iy, 1]] = U;
      vx[[t + 1, iy, Nx + 1]] = vx[[t + 1, iy, Nx]];
      , Indeterminate];

     If[2 ≤ iy ≤ Ny,
      vy[[t + 1, iy, 1]] = -vy[[t + 1, iy, 2]];
      vy[[t + 1, iy, Nx + 2]] = vy[[t + 1, iy, Nx + 1]];
      , Indeterminate];

     If[1 ≤ ix ≤ Nx + 1,
      vx[[t + 1, 1, ix]] = -vx[[t + 1, 2, ix]];
      vx[[t + 1, Ny + 2, ix]] = -vx[[t + 1, Ny + 1, ix]];
      , Indeterminate];

     If[1 ≤ ix ≤ Nx + 2,
      vy[[t + 1, 1, ix]] = 0;
      vy[[t + 1, Ny + 1, ix]] = 0;
      , Indeterminate];
    ];
  ];
];
```