Assignment#5: Computation of Lid-Driven Cavity Flow Using Vorticity-Stream Function Formulation

Problem Statement

The goal of this assignment is to numerically solve the steady-state lid-driven cavity (as shown in Figure 1) flow using the Vorticity-Stream Function formulation. The flow is incompressible and two-dimensional. The velocity field is derived from the stream function, and the vorticity transport equation governs the evolution of vorticity in the domain.

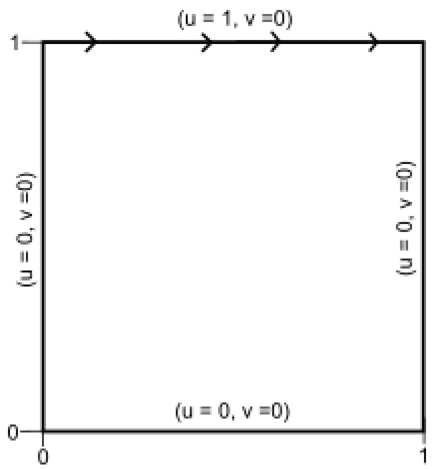

Figure 1: Lid driven cavity

Compute the steady-state lid-driven cavity flow for Reynolds number 100.

1 Governing Equations

The Navier-Stokes equations in terms of vorticity (ω) and stream function (ψ) are given as:

1.1 Vorticity Transport Equation

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

where:

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \quad \text{(vorticity)} \tag{2}$$

$$u = \frac{\partial \psi}{\partial y}, \quad v = -\frac{\partial \psi}{\partial x} \quad \text{(velocity components)} \tag{3}$$

where ν is the kinematic viscosity.

1.2 Stream Function Equation (Poisson Equation)

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega \tag{4}$$

This equation ensures that the velocity field satisfies the incompressibility condition.

2 Boundary Conditions

Lid (Top Wall, $y = 1\ m$):

- $u = 1\ m/s,\ v = 0\ m/s$ (Driven by the lid)
- Imposed via vorticity: $\omega_{i,J} = -\frac{2(\psi_{i,J-1} - \psi_{i,J})}{\Delta y^2} - \frac{2u_{i,J}}{\Delta y}$;
  where $I \times J$ grid is used; $i = 1, 2, \ldots, I$ and $j = 1, 2, \ldots, J$.

Bottom and Side Walls ($y = 0\ m,\ x = 0\ m,\ x = 1\ m$):

- No-slip condition: $u = 0\ m/s, v = 0\ m/s$
- Imposed via vorticity:

$$\omega_{1,j} = -\frac{2(\psi_{2,j} - \psi_{1,j})}{\Delta x^2} \quad \text{(left wall)}$$
$$\omega_{1,j} = -\frac{2(\psi_{I-1,j} - \psi_{I,j})}{\Delta x^2} \quad \text{(right wall)}$$
$$\omega_{1,j} = -\frac{2(\psi_{i,2} - \psi_{i,1})}{\Delta y^2} \quad \text{(bottom wall)}$$

3 Computational Methodology

3.1 Discretization

- Use finite difference method (FDM) on a uniform grid.
- Approximate all spatial derivatives using second-order central difference schemes, except for the spacial derivatives for the cenvective terms in vorticity equation, which should be discretized using second order upwind differences.
- Time marching using explicit Euler schemes.

3.2 Algorithm Outline

1. Initialize the velocity and vorticity fields.
2. Apply the boundary conditions for vorticity at the grid points along the walls.
3. Solve the vorticity transport equation (1) to update the vorticity at new time step.
4. Solve the stream-function equation (4) to compute the stream-function field at new time step.
5. Compute the velocity field at new time step using the definition of stream-function.
6. Return to step 2 and repeat the computation for another time step.

7. Continue marching in time as suggested above till the Root Mean Square (RMS) Residual of u and v values fall below 10^{-8}.

3.3 Convergence Criteria

- Measure the residual of the Poisson (stream-function) equation. The residual $R_{i,j}$ at an internal grid point is the difference between the LHS and RHS of the discretized stream-function equation (4):

$$R_{i,j} = \left(\frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{\Delta x^2} + \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{\Delta y^2} + \omega_{i,j} \right) \quad (5)$$

  A global residual norm is used to check for convergence. The most common choice is the Root Mean Square (RMS) Residual, which is defined as

$$R_2 = \sqrt{\frac{1}{N} \sum_{i,j}^{I-2,J-2} R_{i,j}^2}, \quad \text{where } N = (I-2) \times (J-2)$$

  Terminate the Jacobi or Gauss-Seidel iteration when $R_2 \leq 10^{-2}$, as this level of residual reduction is sufficient for obtaining steady-state results.

- Terminate the time iteration when the Root Mean Square (RMS) residuals of the velocity components u and v fall below a threshold (i.e., 10^{-8}). Root Mean Square (RMS) Residual of f (where, f represents either u or v) is defined as

$$\text{RMS}_f = \sqrt{\frac{1}{N} \sum_{i,j}^{I-2,J-2} \left(f_{i,j}^{(n+1)} - f_{i,j}^{(n)} \right)^2}$$

  Here, superscript n indicates time level.

3.4 Time-step Δt Calculation

1. Compute time step dictated by the convective part (Δt_c) in the vorticity equation based on stability condition:

$$\Delta t_c = \sigma_c \frac{\Delta x \Delta y}{|u_{max}| \Delta y + |v_{max}| \Delta x}$$

where, $|u_{max}| = maximum(|u_{i,j}|); \quad |v_{max}| = maximum(|v_{i,j}|)$
$\forall\ i = 1, .., I \quad j = 1, .., J$
σ_c is Courant number.

2. Compute time step dictated by diffusion part (Δt_d) in the vorticity equation based on stability condition:

$$\Delta t_d = \sigma_d \frac{1}{2\nu} \left(\frac{\Delta x^2 \Delta y^2}{\Delta x^2 + \Delta y^2} \right)$$

σ_c is Diffusion number.

3. Time step (Δt) for the vorticity equation:

$$\Delta t = minimum(\Delta t_c, \Delta t_d)$$

4 Expected Results (Plots)

1. Stream-function contours.
2. Streamlines plot.
3. Distribution of the x-component of velocity vector (u) along the mid vertical line.
4. Distribution of the y-component of velocity vector (v) along the mid horizontal line.
5. Compare u and v velocity components along mid lines with the benchmark solutions of Ghia and Ghia [1]. This data have been uploaded separately in moodle along with assignment.
6. Plot the convergence history for $(RMS)_u$ and $(RMS)_v$ with iterations.

5 Implementation Details

- Initialization:
  - You may provide constant value of ψ (say, $\psi = 100$) along all the walls for all the time steps.

  - Initialize the internal grid point values with the same ψ (i.e. $\psi = 100$) at the starting. Mind that, these internal values will change with time steps.
  - Initialize the vorticities as zero at all point at the starting. Mind that, the voticity magnitude everywhere will change with time steps.

- **Programming Language**: Python, C or C++.
- **Grid Size**: Start with $I \times J$ grid (e.g., 31×31).
- Suggested values for Courant numbers, while time step calculation, are $\sigma_c = 0.4$ and $\sigma_d = 0.6$
- Consider $Re = 100$.

6 Assessment Criteria

- See Rubric for details.

References

[1] Ghia, U., Ghia, K. N., & Shin, C. T., *High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid Method*, Journal of Computational Physics, 48(3), 387–411.

[2] Hoffmann, K. A. and Chiang, S. T., *Computational Fluid Dynamics for Engineers*, Vol. I, 4th ed., Engineering Education Systems (2000).

[3] Pletcher, R. H., Tannehill, J. C., and Anderson, D. A., *Computational Fluid Dynamics and Heat Transfer*,3rd ed., Taylor & Francis (2011).

[4] Roache, P. J. *Fundamentals of Computational Fluid Dynamics*, 2nd ed., Hermosa Pub. (1998).

- **Checklist for submission:**
  1. Submit only the original source code written in C, C++, or Python, with proper inline documentation for each function. Use meaningful variable names to enhance readability.
     Important Note:
     - Submitting code in any other programming language or as a .pdf, .txt, or .docx file will result in zero marks for the entire assignment, as these formats cannot be compiled.
     - Online compiler should not be used for your code.
  2. Include a README.txt file with clear instructions on how to run your code and generate the required plots.
  3. Report: Submit a brief report in PDF format, including:
     - Provide a flowchart representing the logic of your program.
     - Include all the results outlined under the 'Expected Results' section described earlier.
     - The plots you submit must be independently reproducible from your code by the teaching assistants (TAs).
- **Instruction for submission:**
  - Rename your program file using your roll number (e.g., 204010006.c).
  - Rename your report file using your roll number (e.g., 204010006.pdf).
  - Upload all three required documents separately, as specified in the checklist above.
  - Don't submit a zip file.
- **Important Notes:**
  - Refer to the Rubric on Moodle before submission.
  - Marks will be awarded only if the program runs correctly and produces the expected results as per the rubric.
  - Submissions not following the instructions will not be evaluated.
- **Strict Warning: Plagiarism is strictly prohibited!** Copying code from others, online sources, or previous submissions will result in **severe penalties**.

– E N D –