Computational Physics - Final Assignment

# LID-DRIVEN CAVITY FLOW

USING FINITE DIFFERENCES

## Contents

# Practical information

**Supervision and receiving assistance on your assignment:**

- Supervision for the final problems is available. For this assignment, you will be supervised by Muhammad Waleed Iqbal (e-mail: m.w.iqbal@rug.nl).
- In the final weeks of the course (from tutorial 12 on, up to the exam week), the computer rooms are available to work in during the tutorial hours (see the schedule). The tutorial assistants will be present during some of these slots (how this is organized will be communicated later) to give you feedback on the pseudocode/code associated with your final assignments.
- I'm stuck: what do I do?
  - If you find that you get stuck on the assignments, then the teaching assistants are there to help! Please reach out timely if you find that you get stuck.
  - Before reaching out, please make sure to follow the debugging tips provided at the end of the notebook for tutorial B. Try to make the bug/error visible and reproducible, such that your TA can help you most efficiently.
  - When reaching out for assistance, you can share the notebook directly through Google Colab or you may send via email the executable python file/notebook together with the steps (again: see tutorial B) to solve the problem.
  - If you want to drop by to discuss your code, please send an email beforehand: this might save you disappointment in case the TA does not have time or is not present.

**Handing in the reports**

- Please hand in your reports (.pdf format) + your code (i.e. a working notebook file, or a separate .py file). The deadline for handing in the report can be found on Nestor. Prepare your files such that we can quickly regenerate your results (preferably with a single click).
- If there is a serious problem or exceptional circumstance that prevents you from meeting this deadline, please contact the course coordinator Prof.dr.ir. Patrick R. Onck at p.r.onck@rug.nl .
- Working together is allowed, but you have to write and hand in your own code in the end, and have to write the report yourself. Fraud will be taken seriously and appropriate actions will follow.
- If you do not hand in anything before this deadline, we will not register a grade. If you do hand in a report + files, a grade will be registered (also if it is an insufficient grade).
- If you do not meet the deadline, there is the possibility to do a 'resit'. This means that you get an extended deadline for your assignments. However, this will come at the cost of an additional, second, assignment.

## Introduction

In this assignment, you will apply the finite differences method to a well-known fluid dynamics problem, namely lid-driven cavity flow. The solution to this problem is usually used to validate new codes and simulation methods for incompressible viscous flows.

The computational domain for the lid-driven cavity problem is depicted in Fig. 1. We consider an incompressible flow in a two-dimensional rectangular domain with side lengths $l_x = 1$ and $l_y = 1$ (all parameters in this problem are considered non-dimensionalized). The top wall (i.e., the lid) is moved with a constant velocity $V_x = 1$ in the positive $x$-direction. Due to the 'no-slip condition' in fluid dynamics, this is also the fluid velocity along the domain's top boundary. As indicated in the figure, the $y$-component of the velocity at the top boundary is zero, and the velocity is zero in both $x$ and $y$-directions at the other three boundaries.

In this assignment, the aim is to find the velocity and pressure profiles in this rectangular domain by solving for the *stream function* and *vorticity field* from the vorticity form of the Navier-Stokes equations. We will solve these equations using the finite difference (FD) method.

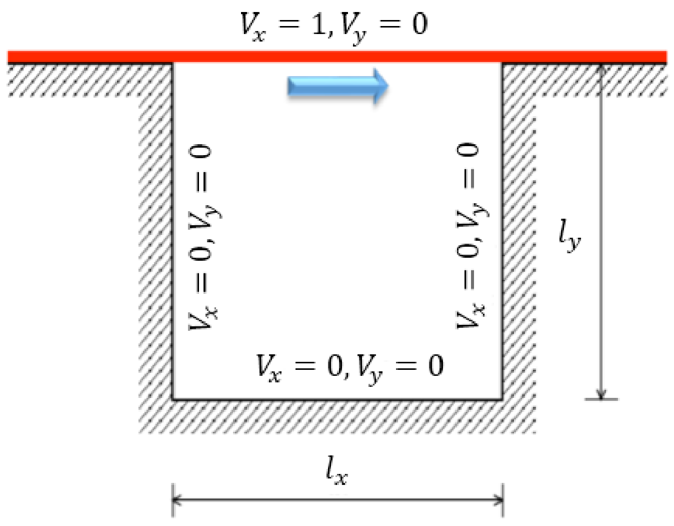

**Figure 1.** The computational domain for the lid-driven cavity problem, with indicated velocity boundary conditions. In particular, the moving top wall is indicated in red.

## Problem Description

This section comprises a brief summary of the equations and boundary conditions that describe lid-driven cavity flow. In order to solve this problem, we are going to discretize these equations using the FD method, as we will see in the following section.

### The vorticity form of the Navier-Stokes equations

The non-dimensionalized vorticity form of the Navier-Stokes equations can be written as

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -w \quad , \tag{1a}$$

$$\frac{\partial w}{\partial t} = -\frac{\partial u}{\partial y}\frac{\partial w}{\partial x} + \frac{\partial u}{\partial x}\frac{\partial w}{\partial y} + \frac{1}{Re}\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2}\right), \tag{1b}$$

where $u$ is the stream function and $w$ is the vorticity. Re is the 'Reynolds number', an important dimensionless number in the field of fluid dynamics which describes the relative contributions of viscous effects versus inertial effects in the fluid flow. The Reynolds number will be one of the input parameters for our simulations.

## Velocity and pressure from stream function and vorticity

The velocity $V$ is related to the stream function $u$ and vorticity $w$ as follows:

$$V_x = \frac{\partial u}{\partial y}, V_y = -\frac{\partial u}{\partial x}, \tag{2}$$

and

$$w = \frac{\partial V_y}{\partial x} - \frac{\partial V_x}{\partial y}. \tag{3}$$

The pressure field (for non-boundary points) can be obtained by solving the following Poisson equation:

$$\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial y^2} = 2\left(\frac{\partial^2 u}{\partial x^2}\frac{\partial^2 u}{\partial y^2} + \left(\frac{\partial^2 u}{\partial x \partial y}\right)^2\right), \tag{4}$$

subject to the boundary conditions described below.

## Boundary conditions

### Velocity field and stream function

The velocity boundary conditions for this problem are a result of no-slip and no-penetration conditions:

$$\begin{aligned} x = 0: &\quad V_x = V_y = 0 \\ x = l_x: &\quad V_x = V_y = 0 \\ y = 0: &\quad V_x = V_y = 0 \\ y = l_y: &\quad V_x = 1,\ V_y = 0. \end{aligned} \tag{5}$$

Equation (5) gives the boundary conditions in terms of the velocity field, but here we are after the stream function and vorticity. These boundary conditions are related as will be explained here. Combining Equation (2) with Equation (5) we can see that for the top boundary,

$$\frac{\partial u}{\partial y} = V_{wall}, \frac{\partial u}{\partial x} = 0, \tag{6a}$$

while for the other three boundaries

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial y} = 0. \tag{6b}$$

Therefore for the top wall $u$ can be a function of $y$, but since $y$ is constant for the top wall, we must have that $u$ is constant as well. Since on the other boundaries $u$ can *not* be a function of $x$ or $y$, it must follow that the stream function is one and the same constant along the entire boundary of the domain. In fact, the exact value does not matter, since we are only interested in relative differences. We can therefore set its value on the entire boundary to zero.

## Vorticity field

Finding the boundary values for the vorticity field is a slightly different story. Since the stream function is constant on the boundaries, on the left and right boundaries Equation (1a) becomes

$$\frac{\partial^2 u}{\partial x^2} = -w_{wall}, \tag{7a}$$

while on the top and bottom boundaries,

$$\frac{\partial^2 u}{\partial y^2} = -w_{wall}. \tag{7b}$$

The boundary conditions should also be discretized. Here, we explain how to find the discretized version of the boundary condition for the horizontally-moving wall. For the other walls, you can follow the same procedure.

First, we expand the stream function using a Taylor series expansion (denoting j=0 for the wall, j=1 for the first interior cell):

$$u_{i,j=1} = u_{i,j=0} + h\frac{\partial u_{i,j=0}}{\partial y} + \frac{h^2}{2}\frac{\partial^2 u_{i,j=0}}{\partial y^2} + O(h^3). \tag{8}$$

Equations (2) and (3) allow us to rewrite Equation (8) into

$$u_{i,j=1} = u_{i,j=0} + hV_{wall} - \frac{h^2}{2} w_{wall} + O(h^3). \tag{9}$$

Finally, the vorticity on the wall becomes

$$w_{wall} = \frac{2}{h^2}\left(u_{i,j=0} - u_{i,j=1}\right) + \frac{2}{h} V_{wall} + O(h). \tag{10}$$

## Pressure field

The interior points of the pressure field can be computed using Equation (4). The boundary conditions for the pressure field can be derived from the velocity-pressure form of the Navier-Stokes equations and gives, for the left and right boundaries,

$$\frac{\partial P}{\partial y} = \frac{1}{Re}\frac{\partial w}{\partial x}, \tag{11a}$$

and for the top and bottom boundaries,

$$\frac{\partial P}{\partial x} = -\frac{1}{Re}\frac{\partial w}{\partial y}. \tag{11b}$$

## Exercises

In this section, you can find the exercises for this final assignment. Some additional necessary information is provided along the way. Answers and solutions should be provided in the report.

Please note that, in contrast to the fluid dynamics exercises in tutorial classes, this problem is *time-dependent*! This means that the stream function, vorticity, velocity and pressure have to be solved for each time step, meaning they depend on $x$, $y$ and $t$.

### 1. Discretization of the Navier-Stokes equations

Let us start by discretizing the main equations that are going to be solved, i.e., Equations (1a) and (1b). To carry out the discretization step, we divide the computational domain in $N$ by $N$ grid cells with discrete centers $(x_i = i\,h, y_j = j\,h)$, where $h$ is the grid cell length.

**TASKS.**

1. Write down the discretized form of Eq. (1a) using the central difference approximation. Note that all variables are evaluated at the same time step $t_n = n\,\Delta t$. Use superscript notation to indicate the time step, and subscripts for the position on the grid, e.g., $w^n{}_{i,j}$.

2. Write down the discretized form of Eq. (1b). Use a forward difference approximation for the time-derivative. Note that all variables on the right-hand side are again evaluated at time step $t_n$, but $t_{n+1}$ will appear on the left-hand side.

3. Write down the discretized boundary conditions for the vorticity for each of the four boundaries.

4. When in doubt, have your equations checked by a TA before building the simulations!

## 2. Building the Navier-Stokes solver

Now, we are going to write the basic code to solve the vorticity form of the Navier-Stokes equations. An example of the program structure is shown in Listing 1.

**Tasks.** Build a Navier-Stokes solver for the lid-driven cavity problem. It should perform the following main steps:

1. Update the stream function using discretized Equation (1a). HINT: This is a Poisson equation.
2. Update the vorticity on the wall using Equation (10) and its equivalents on all four boundaries.
3. Update the vorticity on the interior using Equation (1b).
4. Check your results visually before implementing the velocity/pressure calculations.
5. Next, compute for the final state the velocity field using Equations (2) and plot the results.
6. BONUS: Compute the pressure for the final state using Equation (4). In every iteration, use Equation (11) to update the boundary values, then use (4) to solve for the interior points. HINT: Equation (4) is *also* a Poisson equation!

Note: While building the code, it is advised to use a small number of grid cells so that you can quickly compute and interpret the results. The larger simulations below may easily take an hour to complete!

```
# Example structure of the Navier-Stokes solver

Initialize variables vorticity, streamfunction, N, dt, etc.

# Main loop
for each timestep:
    Update streamfunction
    Update vorticity on boundaries
    Update vorticity

# Postprocessing
Calculate velocity from streamfunction, vorticity
Calculate pressure from streamfunction, vorticity

# Visualization
Plot streamfunction, vorticity, velocity, pressure
```

**Listing 1.** Example structure of the FD Navier-Stokes solver for the lid-driven cavity problem.

## 3. Solving three flow cases

With your program fully functioning, let us put it to the test by considering Reynolds numbers of (A) Re=10, (B) Re=100 and (C) Re=200.

**TASK.** For each of these cases, answer the following questions:

1. Place a probe at the domain center to compare the time history of the horizontal velocity. Explain whether the flow reaches steady state conditions (with little transient behavior) or not and try to estimate the time it takes to establish such a stationary solution. How does this time correlate with the Reynolds number?

   Note: Set a maximum number of time steps to keep your simulations within practical amounts of computation time.

2. Make plots of the stream function, vorticity, velocity and pressure fields. Give a physical interpretation of the velocity field (i.e., relate your results to the underlying physics, and in particular, to the Reynolds number). Discuss the validity and accuracy of your results.

3. Implement also a moving lower wall using appropriate boundary conditions. Try (at least) two cases with the lower wall moving (i) with the same velocity and direction as the upper one, (ii) with the opposite velocity. Report plots of your results, and in particular discuss the symmetry properties of the resulting flow patterns.

4. Make a comparison between the three cases, both from the computational and physics points of view.

NOTE: Given the computational domain has length 1 in both directions, you must choose the grid spacing $h$ and the time step $\Delta t$ such that the following stability criterion is fulfilled:

$$\frac{\Delta t}{Re\, h^2} \leq \frac{1}{4}. \tag{13}$$

In doing so, take note of the accuracy of your solution but also the computational effort.

## Grading criteria/requirements for the report

- In the table below you will find the required (sub)sections of the report, and the general grading criteria for each section.
- Each assignment/problem comes with its own specific sub-questions. Make sure to process the different sub-questions of the assignments in the relevant sections in the report (for example: no derivations in the results sections).
- The grading criteria are in line with the learning objectives, see the [ocasys page of the course]().
- **Please make sure that all of the below sections are actually present in your report, and that you try to follow the guidelines for these sections.**

1. **Introduction section:** This part of the report should at least contain (not specifically in the order as presented below):
    1. A description of the physical problem to be analyzed. You are encouraged to provide an image, or to consult literature sources to provide some additional background. Also indicate what properties, parameters, research questions etc. you will explore in your report.
    2. Announce which computational methods you will use, and explain the general purpose/context of these methods.
    3. The relevant formulae to solve (i.e. PDE's, ODE's, the general algorithm, etc.) and a brief discussion of any analytical solutions (if they exist) that you can compare your numerical result with. Make sure to quickly browse the literature (the references provided in your assignment are a good start) for some background on your assignment problem.

Weight: **1 pt**
Learning objective: **1**

2. **A pseudocode/discretization section.** This part of the report should contain a description of the kind of algorithms that you used, and is the perfect place to showcase your understanding of the simulation method. This section should contain:
    1. An explanation of the algorithms that you used to solve the physical problem at hand.
    2. **An explanation, including pseudocode** that details how you implemented the Algorithms that you yourself wrote for the assignment.
    3. Explain also in this section how the equations are discretized (i.e. what step sizes to take, implementation of boundary conditions, etc.).
    4. Discuss what kind of errors (size, sources) you would expect, based on the algorithms that you used. Discuss also the tolerance values, grid size and time step that you use and how they may affect the accuracy of your solution.

    * Tasks 1.1, 1.2, 1.3 should be provided in this section

Weight: **2 pts**
Learning objective: **2**

3. **A results section (possibly combined with discussion section).** In this section, provide the output of your code (i.e. graphs, tables, visualizations, etc.) and provide your observations. This section should contain:
    1. Graphs of a decent quality. Make sure that the graphs are legible, that it is clear for the reader which variables are plotted (use axis labels) and that you include a legend in case you plot multiple types of data/multiple lines in the same graph. You are encouraged to make animations, you can send these separately.
    2. Your observations, based on the results. You should clearly indicate in the main text what you see in each figure (however, please interpret the results in the discussion section).
    3. Also included in the grade of this part is the 'correctness' and 'quality' of your results: are the results correct? Is your simulation converged?

Weight: **3 pts** (together with discussion section)
Learning objectives: **1,4**

4. **Discussion section (possibly combine with the results section).** In this section, you should form an interpretation of your results, and possibly draw comparisons to theory or expectations. This section should at least contain:
    1. An interpretation of the results in light of the questions of the assignment. How do your results compare to the theoretical or experimental results? Relate your findings as much as possible to the underlying physics.
    2. A discussion on the applicability/accuracy/feasibility of the computational method to the problem that you studied
    3. If the method allows, incorporate a discussion on the quality of the results, i.e. assess the error sources and convergence of your program, and highlight possible alternatives or improvements.

Weight: **3 pts** (together with results section)
Learning objectives: **1,4**

5. **Conclusion section**. In this section, you should concisely but concretely summarize your report. This section should contain:
    1. A mention of the research problem/motivation
    2. A re-iteration of the used methods and developed algorithms.
    3. The main results obtained and the main conclusions inferred from these results (and the discussion section)
    4. A reflection/perspective statement: how did the computational method work out for this problem, and are there better ways to study your problem?

Weight: **1 pt**
Learning objective: **5**

6. **The (quality of the) complete code.** You do not need to list your complete code in the appendix of the report but are free to do so. What we **do** want to receive is the program as a notebook (.ipynb) or a separate executable .py file. We will grade the quality of your code based on some of the aspects that you practiced throughout the course, and reward initiative to improve the working of your code. Points that we consider in judging the quality of your code are:
    1. Overall code structure: are your loops, algorithms etc., written efficiently? Do you use unnecessary loops, or initialize unnecessary variables? Do you use functions for repeating parts, and do you use the appropriate data structures?
    2. Legibility of your code for a third person. Did you use descriptive variable names or the proper amount of comments/docstrings? A little commenting goes a long way to help yourself and others to understand the code better.
    3. Features added by yourself that optimize the working of the code. Examples are the use of additional python packages for checkpointing/storing data (think of pickle, for example).

Weight: **2 pts**
Learning objectives: **1,3,4**