B.Tech Project - CP303

Analyzing flow past an anisotropic particle

Submitted by

Akshat Chauhan (2021meb1265)

Kayitha Saran Yadav (2021meb1290)

Korupolu Deekshitha (2021meb1292)

Patel Mahant (2021meb1306)

Ujjawal Kumar (2021meb1331)

Under the supervision of

Dr. Navaneeth K. Marath

Department of Mechanical Engineering

Indian Institute of Technology, Ropar

Year of Submission: 2025

Abstract

This study investigates fluid-structure interaction within microfluidic systems using dynamic meshing, with a focus on drag behavior of rotating elliptical bodies in viscous flow. Our initial goal was to simulate the motion of elastic particles and compute hydrodynamic forces using IBM. While open-source IBM codes were available, we encountered significant challenges in adapting them for accurate force prediction and particle movement. As a result, we redirected our efforts toward a more structured investigation: analyzing the effect of elliptical body orientation on drag force in a controlled flow field.

We first validated our simulation framework under uniform flow conditions, conducting a grid independence study to ensure numerical accuracy. The results matched well with existing literature, establishing confidence in our approach. Using this verified setup, we then applied Poiseuille flow conditions to the elliptical geometry, simulating various orientations and Reynolds numbers. We plotted the variation of drag with respect to angle and performed a detailed analysis of mesh dependence for the Poiseuille flow case as well. These results serve as a reference for understanding the impact of orientation in confined shear-driven flows.

Overall, our work highlights a robust computational methodology for analyzing flow past immersed geometries and contributes new insights into the influence of shape orientation on drag in both uniform and Poiseuille flow regimes. These findings have potential applications in microfluidic design, bio-separation, and lab-on-a-chip systems.

Contents

List of Figures

Nomenclature

β	Blockage ratio
μ	Dynamic Viscosity
ρ	Density
a	Semi major axis of ellipse
b	Semi minor axis of ellipse
C_D	Drag coefficient
C_L	Lift coefficient
F_D	Drag force
F_L	Lift force
P	Pressure
Re	Reynolds Number
U	Velocity

1 Introduction

Microfluidic devices are essential tools in a variety of fields, especially in biological and medical diagnostics. They work by controlling and manipulating fluids on a very small scale, making it possible to separate, mix, or sort tiny particles like cells and proteins. Microfluidics represents 'the science and technology of systems that process or manipulate small amounts ($10^{-}9$ to $10^{-}18$ L) of fluids, using channels with dimensions ten to hundreds of micrometers'1.

One of their key uses is in separating healthy cells from diseased ones, such as in cancer diagnostics or isolating pathogens in a blood sample. As these cells or particles move through the tiny channels in these devices, they experience different hydrodynamic forces, which play a crucial role in how effectively they can be sorted. To optimize these devices, it is important to understand and quantify these forces. Microfluidic devices use different techniques to separate particles, including methods that leverage properties like particle size, shape, and electrical charge.

Accurate modeling and quantification of these fluid-structure interactions are vital for optimizing microfluidic device performance. Several particle separation techniques have been developed over the years, such as deterministic lateral displacement, inertial focusing, dielectrophoresis, and acoustic separation. Despite their varied operating principles, all of these methods depend critically on understanding how particles interact with the surrounding fluid under specific flow regimes. Some techniques include deterministic lateral displacement, inertial focus, dielectrophoresis, and acoustic separation. These methods are especially useful in medical diagnostics, where rapid and accurate sorting can make a significant difference[1]. When particles move through microchannels, they are subjected to several forces, such as drag, lift, and pressure forces. Understanding these forces and how they interact with the particles is essential for improving the design and efficiency of microfluidic devices. However, calculating these forces can be challenging due to the complex nature of fluid-structure interactions.

To address this complexity, we are using the Immersed Boundary Method (IBM), a computational technique originally developed by Dr. Charles Peskin in 1972 [2] to model how blood flows through heart valves.It has a unique ability to perform the entire simulation on a fixed cartesian grid, without explicitly creating a mesh. IBM has since evolved into a widely used tool for simulating the interactions between fluids and flexible structures. It has been applied in fields ranging from biology and bioengineering to materials science. IBM is particularly effective

for systems where particles or structures are immersed in moving fluid, such as in microfluidic devices. The way IBM works is by combining two types of grids: one fixed grid for the fluid (Eulerian) and a flexible, moving grid for the structures (Lagrangian). This combination allows for accurate simulations of how the fluid affects the particles and vice versa. One of the best things about IBM is that it can handle large deformations of structures without needing to constantly remesh the grid, making it ideal for microfluidic applications where particle shapes and behaviors can vary greatly.

For this project,we implemented the Immersed Boundary Method (IBM) using ANSYS (Dynamic Mesh) to simulate the movement of elastic particles in microchannels and calculate the hydrodynamic forces acting on them. IBM is particularly advantageous for such simulations, as it eliminates the need for body-fitted meshing, making it highly efficient for modeling fluid-structure interactions. The purpose of this study was to investigate the influence of these forces on particle behavior, which is essential for optimizing microfluidic device designs. By improving our understanding of particle transport and deformation in confined flows, we can enhance the efficiency of microfluidic systems used in biomedical and industrial applications.

We extend our research to analyze the crossflow of fluids past cylinders with circular cross-section in confined channels. This problem is of both fundamental and practical importance, as it provides insights into flow separation, wake formation, and drag forces in bounded environments. Our focus is on steady-state flow around a circular cylinder, examining the impact of the blockage ratio (β=H/D) on fluid dynamics, drag forces, and wake development. Using IBM within ANSYS, we efficiently simulate fluid-structure interactions without requiring complex meshing techniques. This study further explores the contributions of pressure drag and friction drag to the total drag coefficient, as well as the influence of Reynolds number on flow characteristics. The findings will provide valuable insights into the effect of confinement on flow behavior, which has implications for microfluidic design, biomedical applications, and engineering systems involving confined flows.

Section 2 provides a review of the literature on the development of the Immersed Boundary Method. Section 3 discusses analytical and simulation details. The results and discussion related to the observations are presented in 4.

2 Literature Review

Fluid-structure interaction (FSI) problems are crucial in both engineering and biological systems, where the interaction between fluids and elastic structures leads to complex, fully coupled dynamics. The Immersed boundary method, pioneered by Dr.Charles Peskin, is particularly well-suited to model such systems, with applications ranging from cardiovascular dynamics to aquatic locomotion and insect flight.The method employs a fixed Eulerian grid for the fluid domain and a moving Lagrangian framework for the structure. Forces exerted by the structure on the fluid are interpolated, enabling the simulation of interactions between flexible or rigid bodies and the surrounding fluid with high accuracy. Subsequent developments have adapted IBM to a wide range of problems, including particle-laden flows and microfluidic applications.

There are several high-performance implementations of the IB method, such as IBAMR (which

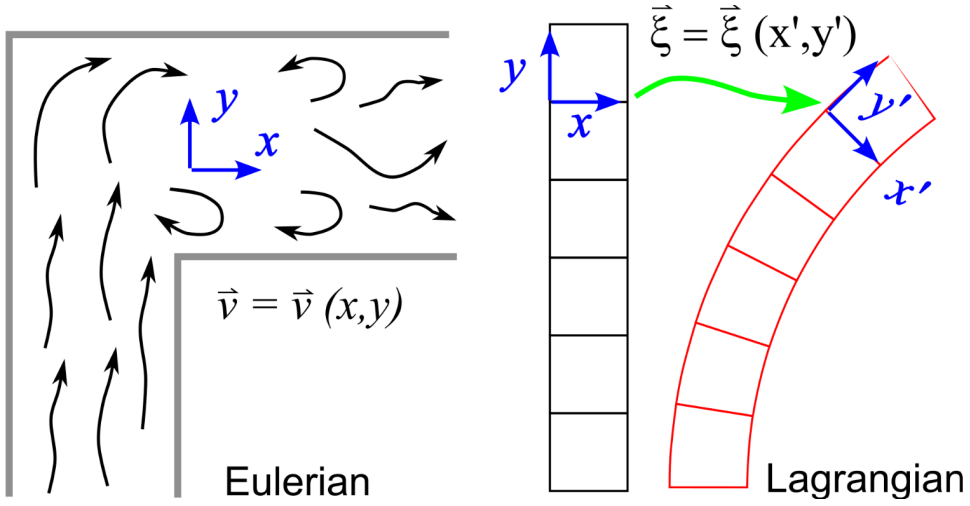


Figure 1: Difference between Lagrangian and Eulerian description

supports adaptive mesh refinement and parallelization in C++) and IBIS (implemented in FORTRAN). However, these require a deep understanding of lower-level programming languages and are computationally intensive, making them inaccessible to many researchers. Moreover, the installation and usage of these software packages can be complex due to dependencies on various libraries and high-performance computing requirements.

While there are other open-source IB codes, like matIB and pyIBM, they do not offer the range of fiber models, examples, or efficiency found in IB2d. Additionally, these earlier tools were often confined to specific problem domains or lacked broad applicability in biomechanics.

The IB2d package is positioned as a highly accessible tool for both experienced and novice users in the scientific community, providing full implementations in Python and MATLAB. IB2d simplifies the workflow for solving FSI problems and allows users to model a wide range of biomechanics problems, such as plant biomechanics, insect flight, muscle-fluid interactions, and physiological processes.

One of the key contributions of IB2d is its diverse array of fiber models, which provide significant flexibility in simulating the material properties of immersed structures. These models include Hookean and non-Hookean springs for elastic stretching, torsional springs for resistance to bending, and target points for enforcing motion or fixation at specific locations. Additionally, mass points allow the incorporation of inertia and gravity effects, while porous models simulate the permeability characteristics of biological tissues. Muscle models further enhance the framework by representing muscle dynamics through force-velocity and length-tension relationships, enabling detailed and realistic simulations of physiological behaviors.

The implementation of IBM ideology with ANSYS (Dynamic Mesh) represents a significant advancement in CFD simulations, offering improved efficiency and accuracy in modeling deformable structures in complex flow environments. By leveraging the versatile material representations of IBM and the adaptive meshing capabilities of ANSYS, researchers can achieve high-fidelity simulations applicable to a broad range of engineering and biomedical problems. Dynamic meshing is a widely used computational technique in Computational Fluid Dynamics (CFD) to simulate problems involving moving boundaries, deforming geometries, and fluid-structure interactions (FSI). Unlike static meshing, where the computational grid remains fixed, dynamic meshing enables the adaptation of the grid in response to changes in geometry, improving the accuracy of simulations without excessive computational costs.

Several studies have explored dynamic mesh techniques to handle complex simulations efficiently. Zhang et al. (2010) highlighted that dynamic meshing is essential for FSI problems, particularly in biomedical applications, aerodynamics, and microfluidics. The Spring-Based Smoothing method, where mesh nodes behave like interconnected springs, and the Laplacian Smoothing method, which redistributes node positions based on neighbors, are commonly used to improve mesh quality during deformation (Blom, 2000). In addition, local remeshing techniques are employed to refine or coarsen the mesh in highly deformed regions, ensuring numerical stability (Löhner, 2008).

In CFD software like ANSYS Fluent, dynamic meshing is implemented using three primary

methods:

Smoothing: Adjusts node positions to maintain mesh quality. Local Remeshing: Replaces distorted elements with refined or coarser mesh elements. Layering: Adds or removes mesh layers in structured grids to accommodate boundary motion. Studies by Mittal Iaccarino (2005) demonstrated the effectiveness of dynamic meshing for immersed boundary simulations, particularly in handling moving objects within a fluid domain without the need for body-fitted meshes. This has applications in microfluidic particle transport, oscillating structures, and biological tissue simulations.

Because of the compatibility issues with our software,we are working in Ansys using Dynamic meshing and UDF. Dynamic meshing and User-Defined Functions (UDFs) are crucial in computational fluid dynamics (CFD) for handling simulations involving moving boundaries and deforming structures. Dynamic meshing techniques allow the computational grid to adapt as the geometry moves, making it suitable for applications such as aerodynamics, fluid-structure interaction (FSI), and combustion systems. Various approaches to dynamic meshing include smoothing methods like Laplacian smoothing and the spring-based method, remeshing techniques that reconstruct the mesh when excessive distortion occurs, and the layering method used for structured meshes. Additionally, the Arbitrary Lagrangian-Eulerian (ALE) method provides a hybrid approach that accounts for fluid-structure interactions effectively. These methods have been widely applied in simulations of airfoil motion, arterial blood flow, and moving piston engines.

User-Defined Functions (UDFs) in CFD enable customization of solvers by allowing users to define complex physics beyond built-in solver capabilities. Written in the C programming language, UDFs help define boundary conditions, specify custom material properties, introduce source terms for transport equations, and control dynamic meshing. In the context of dynamic meshing, UDFs can be used to prescribe motion for rigid bodies, implement deforming mesh techniques, and enforce mesh quality controls to prevent excessive skewness. They also play a key role in applications such as defining temperature-dependent viscosity in non-Newtonian fluid simulations and incorporating adaptive mesh refinement (AMR) based on flow gradients. While UDFs provide greater flexibility in simulations, they come with challenges such as increased computational costs, potential numerical instabilities, and the need for efficient coding practices to optimize performance.

Combining dynamic meshing with UDFs enhances the ability of CFD solvers to tackle

complex problems across aerospace, biomedical engineering, and energy systems. While commercial software like ANSYS Fluent, OpenFOAM, and STAR-CCM+ offer built-in dynamic meshing capabilities, UDFs allow further customization, making them essential for specialized research and engineering applications. Future advancements in AI-driven mesh adaptation and hybrid meshing techniques could further improve the efficiency and accuracy of CFD simulations involving moving boundaries.

3 Analytical and Simulation Details

We tried to simulate the trajectory of a rigid circular cylinder in a channel using the Immersed Boundary Method(IBM) in the previous semester. We had an open source code [3][4] for it and we were trying to modify it according to our need. We validated the results for rigid cylinder from the immersed boundary method with literature. But as we increased the elasticity of the particle, the code threw errors and after many attempts we were not able to identify the reason for it. Hence we continued our project using ANSYS Workbench.

We chose an elliptical particle because it can represent the structure of the cell, and we will be performing the simulations for different aspect ratios to account for the elasticity of the cell.

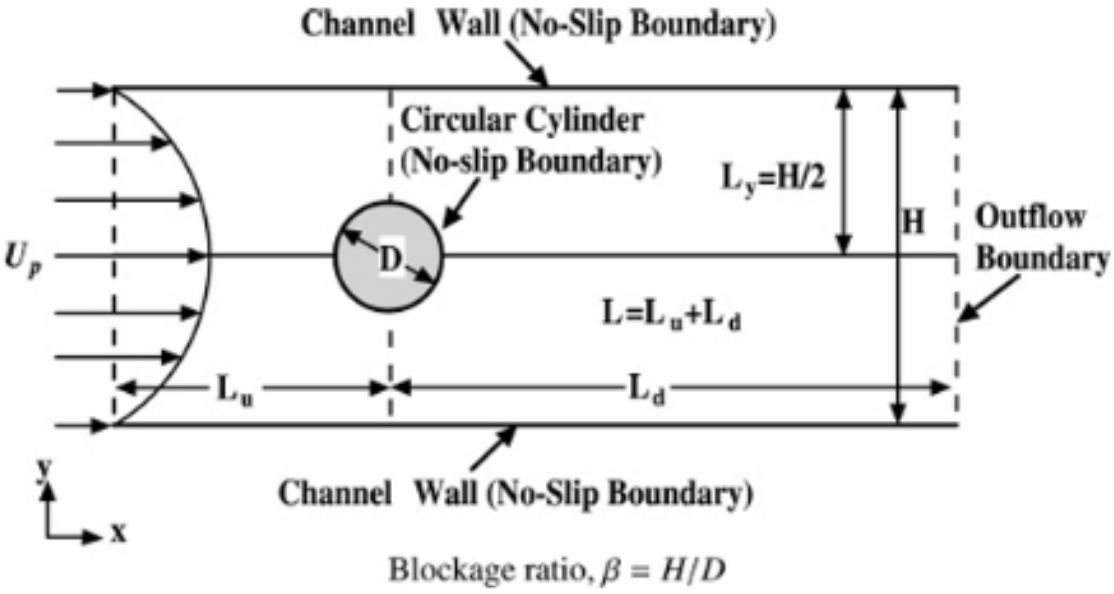


Figure 2: Geometry in consideration

3.1 Governing Equations

ANSYS Workbench solves the incompressible Navier-Stokes equations using the finite volume methods.

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{f} \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{2}$$

Using the solutions of the equation, we then calculate the drag and lift forces exerted by the fluid on the walls of the particle. Corresponding to the drag and the lift forces, the drag and the lift coefficients are calculated using:

$$C_D = \frac{F_D}{\frac{1}{2}\rho U^2 A} \tag{3}$$

$$C_L = \frac{F_L}{\frac{1}{2}\rho U^2 A} \tag{4}$$

where A is the area of projection.

3.2 Simulation Methodology

We analyzed the flow past an elliptical particle in uniform flow to validate the results with [5]. Grid independence tests were also performed for the simulation. Steady-state incompressible flow equations were solved in the given domain.

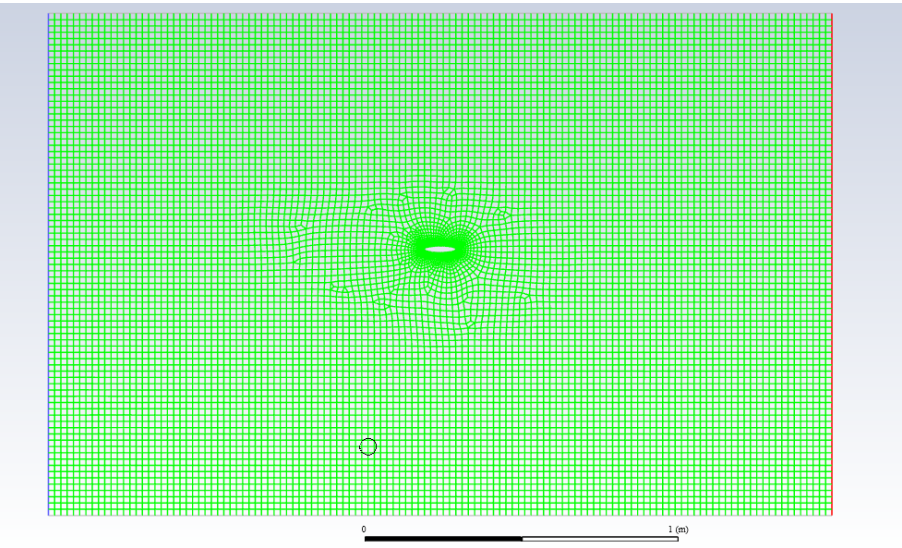


Figure 3: Geometry for validation

After validating the results, we moved on to the main focus of the project, which was to analyze the flow over an elliptical particle confined in a channel. Based on the forces on the particle, we can predict the trajectory of the particle in the channel

We have simulated flow over a fixed ellipse in a 2d channel at different Reynolds numbers. We considered a steady, Poisueille flow for all our simulations. The effect of orientation was also considered. The steady state incompressible equations were solve within the given domain by varying the Reynolds number, orientation and aspect ratio of the ellipse.

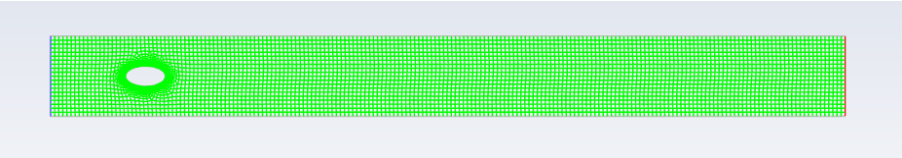


Figure 4: Geometry for simulations

3.3 Validation

In order to validate the results, we first simulated the flow for which the results were already known in from literature [5]. We simulated the uniform flow past an elliptic cylinder at 0 degree angle of attack and compared the drag coefficients at various Reynolds numbers.

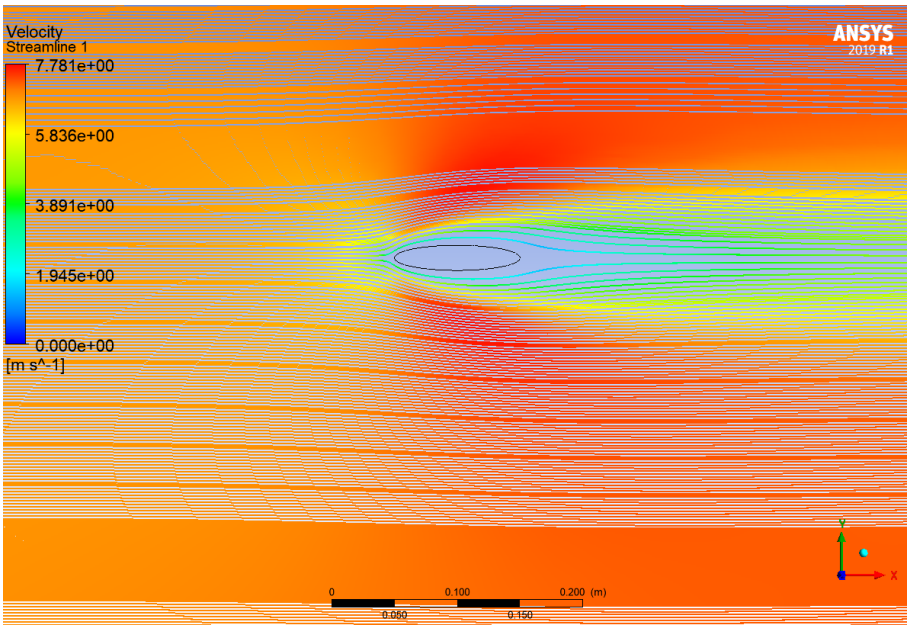


Figure 5: Streamlines at Re=20 and aspect ratio 0.5

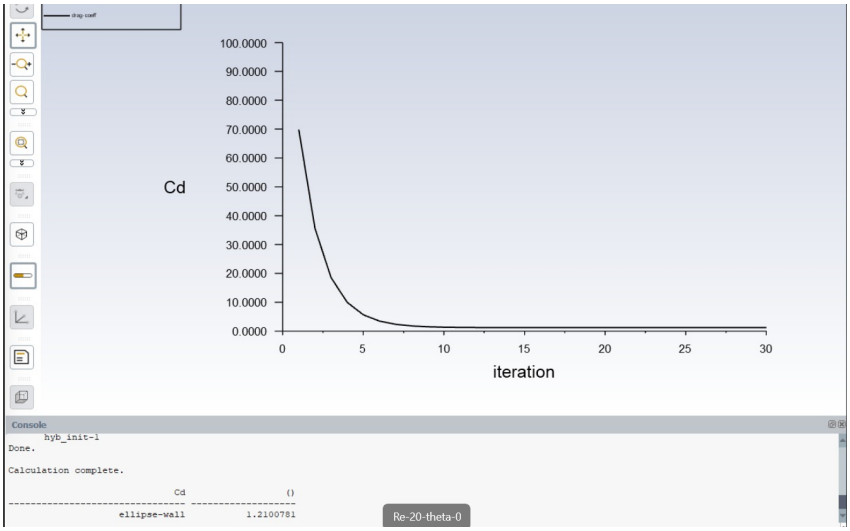


Figure 6: Drag coefficient at Re=20 and aspect ratio 0.5

The grid independence test was also performed . The drag coefficient was measured for a single Re at various mesh sizes. The value of the drag coefficient didn't change much as the grid size was lowered.

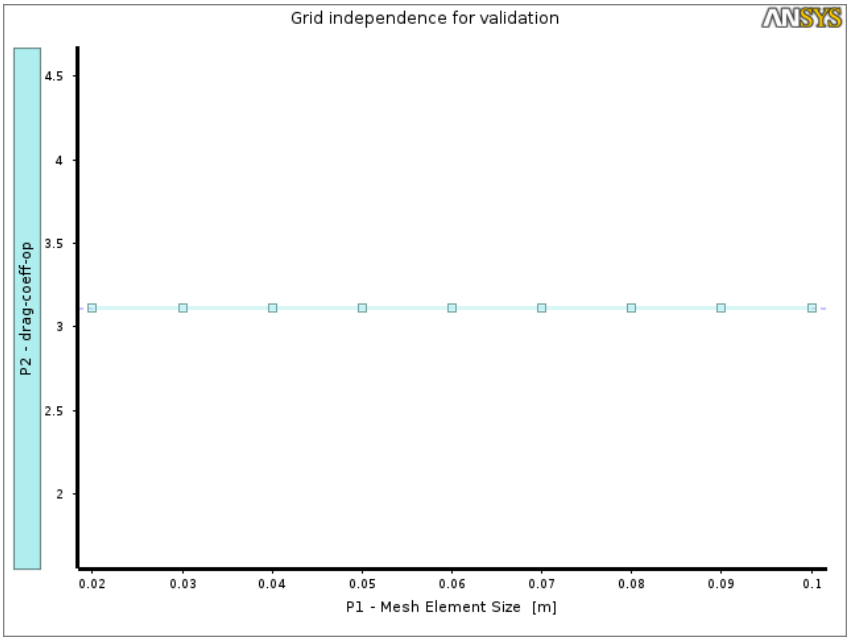


Figure 7: Grid independence for 0° AOA and aspect ratio 0.5

The drag coefficients were calculated for 3 Reynolds number of 5,20 and 40. The table below shows the comparisons of our results to the results in literature [5]

Drag coefficient	Dennis and Chang	Present
Re = 5	3.11	2.96
Re = 20	1.228	1.21
Re = 40	0.794	0.796

Table 1: Drag coefficient comparison

As observed, we got very close results to the literature and hence we can now move forward in producing our results. The only thing changed in our case is the boundary conditions and the orientations of the ellipse.

3.4 Details of our simulation

Our goal is to find out the forces acting on the elliptical particle to track the trajectory of the particle confined in a channel. Since, we were not able to simulate the effect of elasticity, we decided to do the simulations for different aspect ratios of the ellipse at different orientations. We simulated the elliptical particle for the aspect ratios(b/a) of 0.1,0.25,0.5 and 0.9. The angle of attack (AOA) was chosen to be 0,45 and 90 degrees.

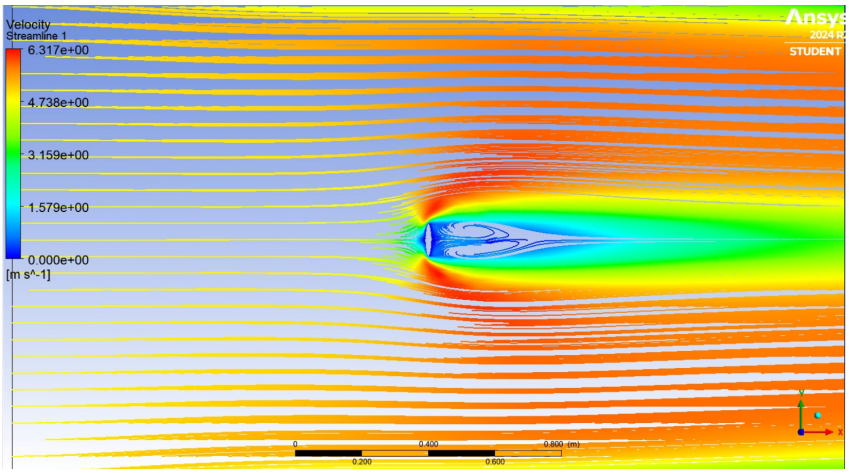


Figure 8: Streamlines at Re=40, aspect ratio 0.5 and 90° AOA

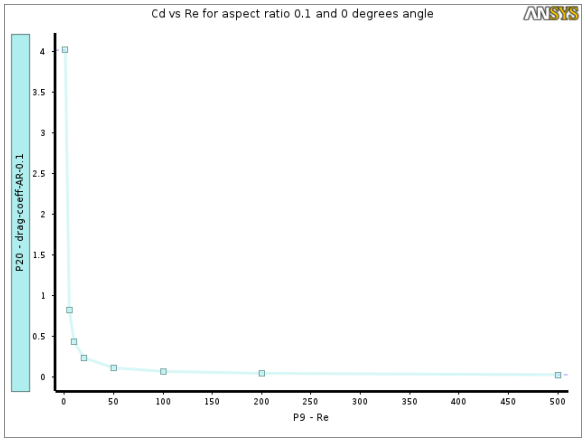
We can see the vortices forming behind the particle at reynolds number of around 40

4 Results and discussions

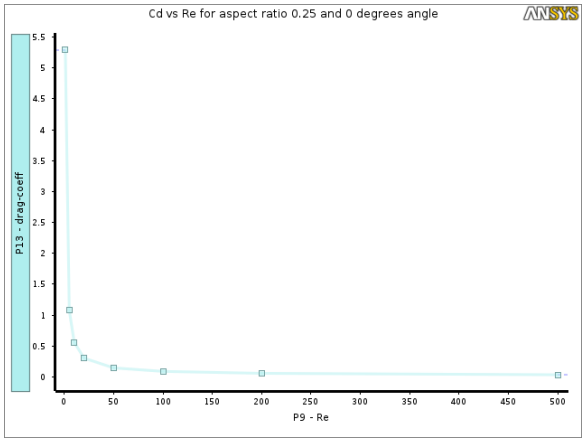
We simulated the elliptical particle for the aspect ratios(b/a) of 0.1,0.25,0.5 and 0.9. The angle of attack (AOA) was chosen to be 0,45 and 90 degrees. We plotted the drag coefficient vs Re for every aspect ratio considered and for different angles of attack.

4.1 0° AOA

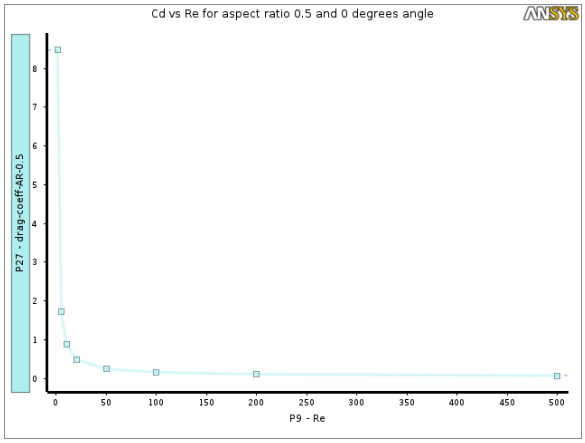
Drag coefficients measured for various Reynolds number:



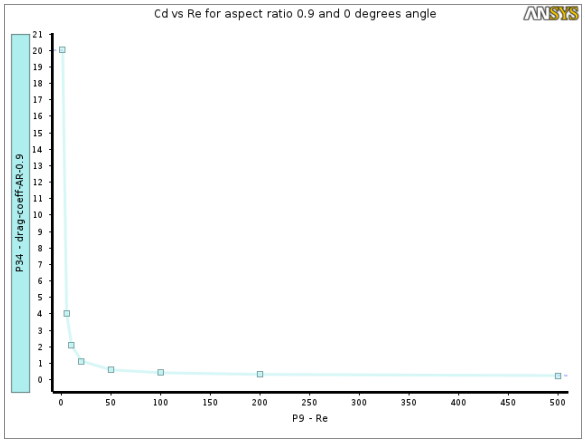
(a) Aspect ratio 0.1, 0° AOA



(b) Aspect ratio 0.25, 0° AOA



(c) Aspect ratio 0.5, 0° AOA



(d) Aspect ratio 0.9, 0° AOA

Figure 9: Drag coefficient vs Reynolds number for different aspect ratios at 0° angle of attack

All of these results were plotted together on a single plot. We can observe that as the aspect

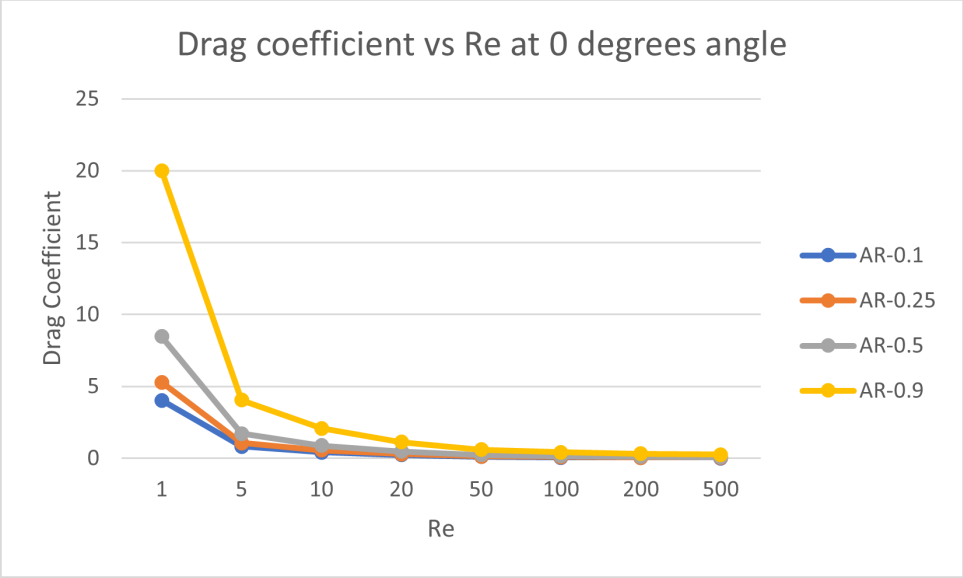


Figure 10: Drag-coefficients comparisons for 0° AOA

ratio increases, the drag coefficient also increases for a given Re, which is expected because as the body gets more bluff, the drag coefficient and the drag force on it increase for a given Re.

Grid independence test was conducted for Re=10 and aspect ratio 0.5 The value of the drag

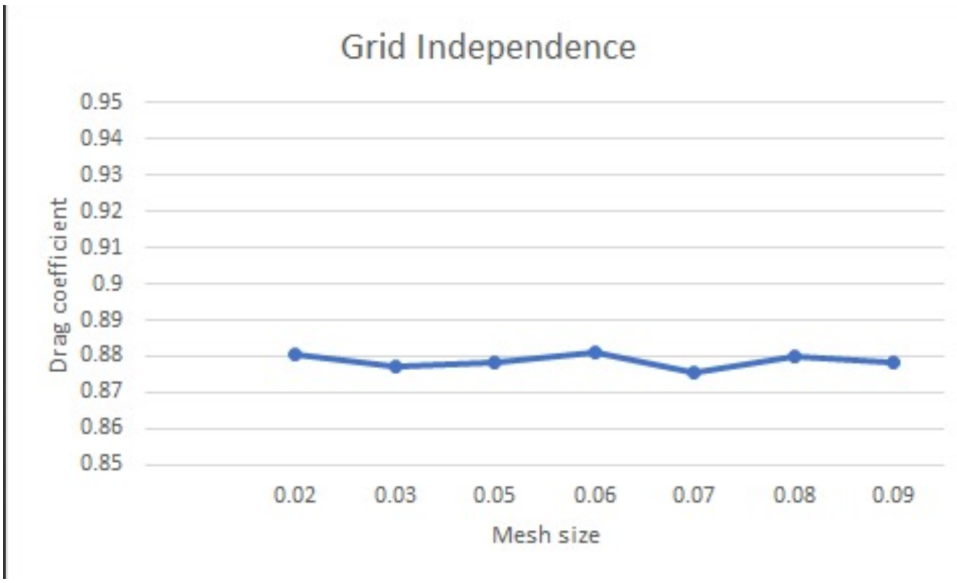


Figure 11: Grid independence for 0° AOA

coefficient didn't changed much as the meh size was decreased.

4.2 45° AOA

Similar exercise was done for 45° AOA and 90° AOA and following results were plotted in a separate plot for 45° AOA and 90° AOA.

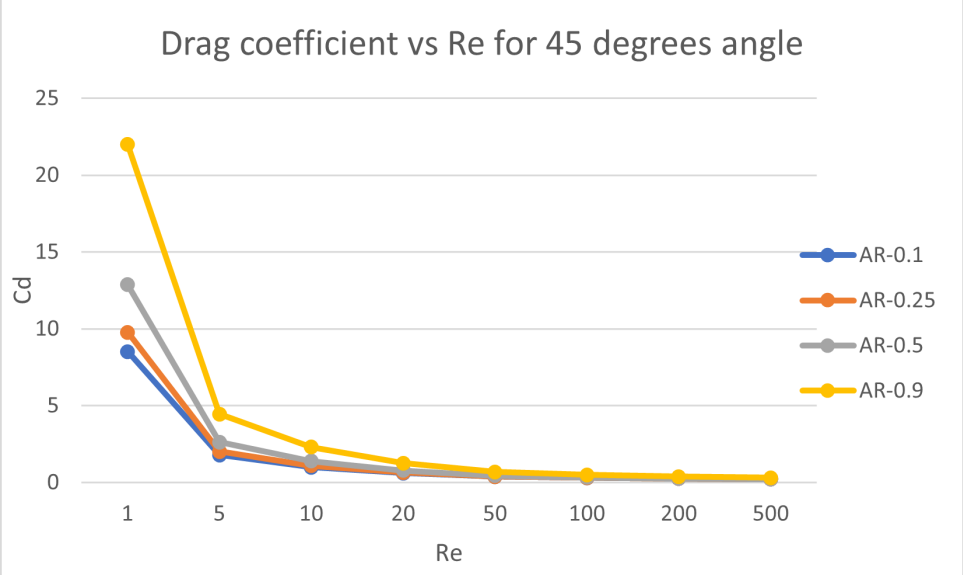


Figure 12: Drag-coefficients comparisons for 45° AOA

Since the elliptical particle is anisotropic, we will get a lift force also for 45° angle from the positive x-axis.

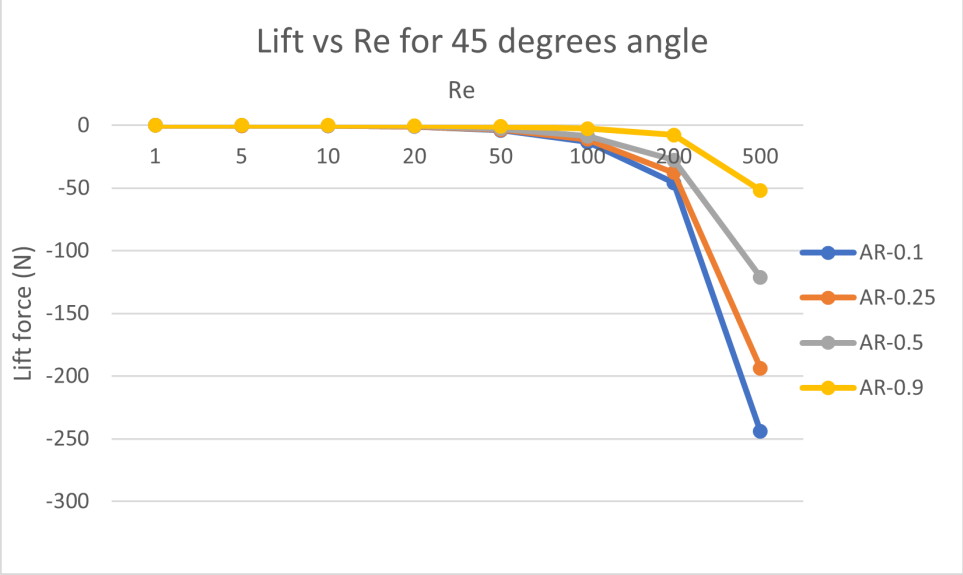


Figure 13: Lift force comparisons for 45° AOA

The magnitude of the lift force decreases as the aspect ratio increases.

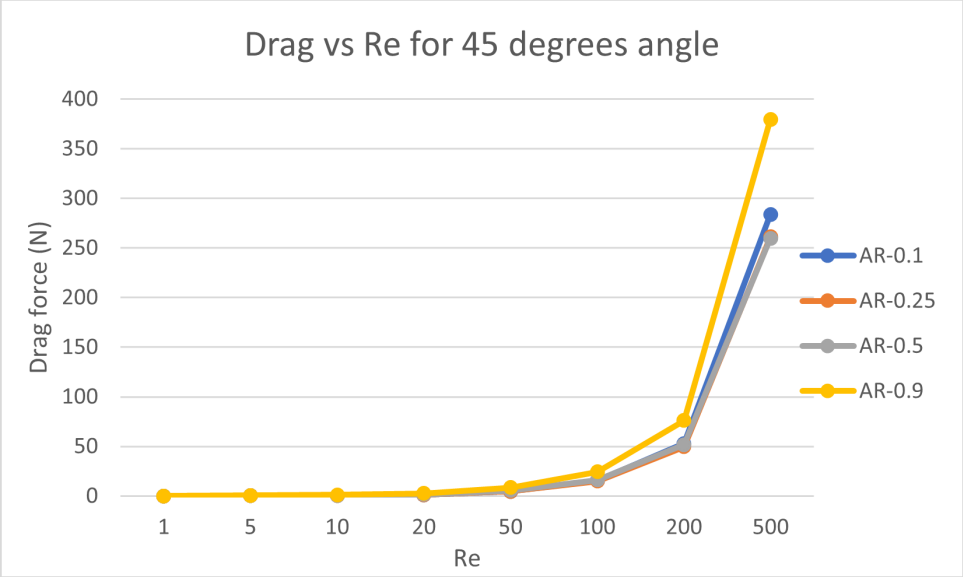


Figure 14: Drag force comparisons for 45° AOA

We can see that as the Re increases, the drag force on the particle increases while the drag coefficient decreases.

4.3 90° AOA

The same exercise was done for 90° AOA. We won't observe the lift force here because the particle will be symmetric, so the force in the upper half and the lower half will cancel each other out.

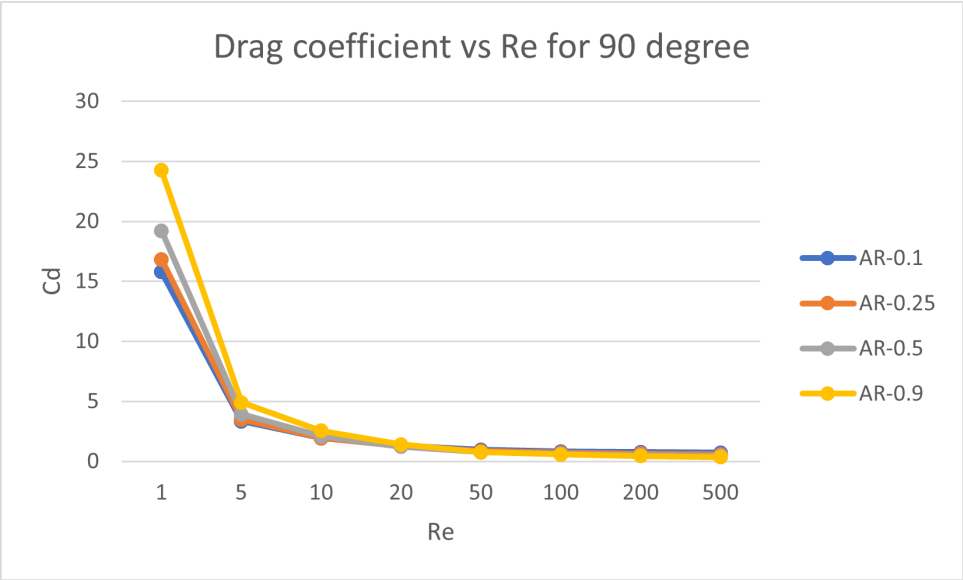


Figure 15: Drag coefficient comparisons for 90° AOA

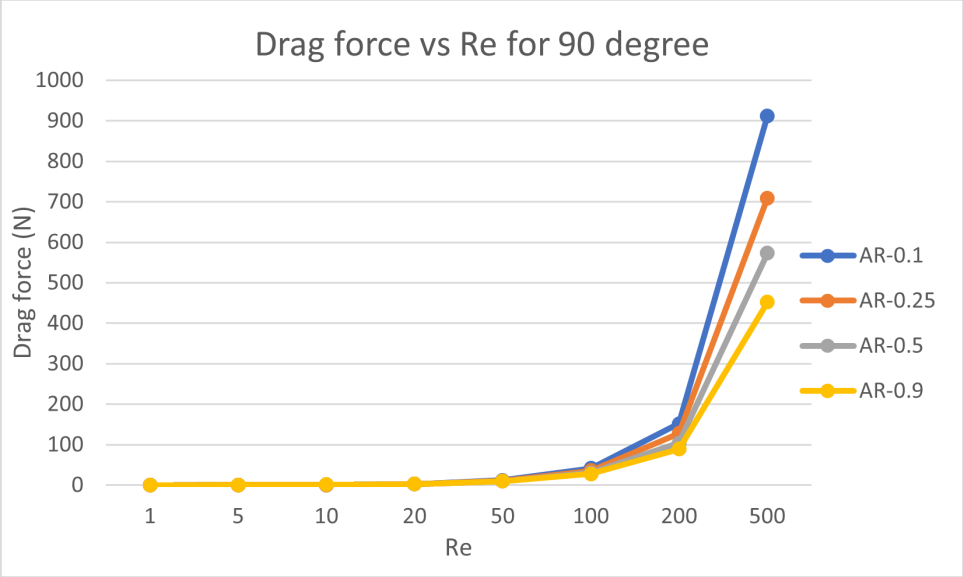


Figure 16: Drag force comparisons for 90° AOA

We can observe in the drag force comparisons that as the aspect ratio increases, the drag force decreases. This is due to the early separation of the boundary layer in the low aspect ratio ellipse at 90° AOA. The later the boundary layer separation, the lower the drag force.

5 Conclusion

We initially understood the Immersed Boundary Method (IBM) and implemented it in MATLAB to simulate the motion of a mass point within a confined channel. To validate the code, we attempted to modify it to compute the drag force on a stationary rigid 2D cylinder. However, the results obtained from the MATLAB simulation did not match the values reported in the literature.

To investigate the cause for the error, we performed a similar simulation in ANSYS, which produced highly accurate results. This comparison helped us identify the source of the error in the MATLAB implementation. The issue found out to be the usage of periodic boundary conditions in the MATLAB code. These boundary conditions likely prevented the flow from fully developing before reaching the cylinder, which is a critical requirement for obtaining accurate drag force values. In contrast, the ANSYS simulation successfully achieved a steady-state flow and ensured the flow was fully developed before interacting with the cylinder, thereby yielding reliable and accurate results.

In this semester, we started to simulate the fluid structure interaction in Ansys Fluent. And we found a research paper on drag coefficients and forces over an ellipse at different orientations. And then we tried to compare our Results with the paper. First we evaluated the drag force and coefficient of a stationary ellipse in a uniform flow and compared and validated with the literature. The values we got were very close to the paper. Then we simulated a moving ellipse inside a moving fluid confined in a rectangular channel and evaluated the drag coefficient and drag force.

The exercise of finding the forces on the elliptical cylinder can help evaluating the trajectory of the particle in a channel flow. By evaluating the trajectory of the particle, we can separate the particle with similar structural properties and can be applied in the development of the microfluidic devices to separate cells and other particles from a stream of blood or any fluid.

References

[1] Jun Zhang, Sheng Yan, Dan Yuan, Gursel Alici, Nam-Trung Nguyen, Majid Ebrahimi Warkiani, and Weihua Li. Fundamentals and applications of inertial microfluidics: A review. *Lab on a Chip*, 16(1):10–34, 2016.

[2] Charles S Peskin. Flow patterns around heart valves: a numerical method. *Journal of computational physics*, 10(2):252–271, 1972.

[3] Nicholas A Battista, W Christopher Strickland, and Laura A Miller. Ib2d: a python and matlab implementation of the immersed boundary method. *Bioinspiration & biomimetics*, 12(3):036003, 2017.

[4] Nicholas A Battista, W Christopher Strickland, Aaron Barrett, and Laura A Miller. Ib2d reloaded: A more powerful python and matlab implementation of the immersed boundary method. *Mathematical Methods in the Applied Sciences*, 41(18):8455–8480, 2018.

[5] SCR Dennis and Gau-Zu Chang. Numerical integration of the navier-stokes equations for steady two-dimensional flow. *The Physics of Fluids*, 12(12):II–88, 1969.